Oracle® Essbase

Database Administrator's Guide

リリース 11.1.2.3

ORACLE

ENTERPRISE PERFORMANCE
MANAGEMENT SYSTEM

Essbase Database Administrator's Guide, 11.1.2.3

著者: EPM 情報開発チーム

目次

ドキュメントのアクセシビリティについて

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

Access to Oracle Support

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

第 I 部

Essbaseの理解

Essbase の理解の内容：

- Essbase について
- Essbase の実装の開始
- 多次元データベースの理解
- ケース・スタディ: 単一サーバー、多次元データベースの設計
- Administration Services について

1 Essbaseについて

この章の内容

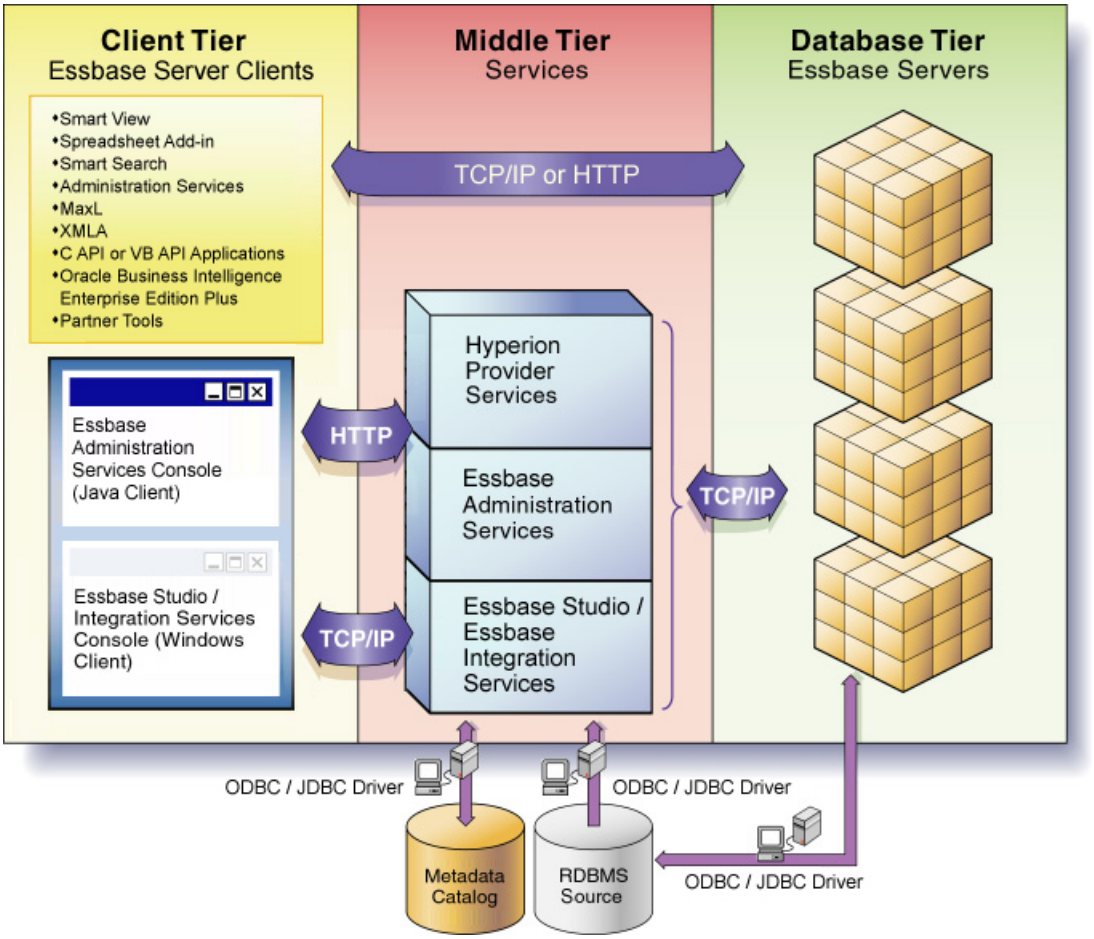
はじめに

Oracle Essbase 製品を使用することにより、企業は、重要なビジネス情報を必要なときに適切なユーザーに提供できます。企業は、Essbase を使用して、既存の複数のデータ・ソースのデータをすばやく利用および統合し、フィルタ済の情報をユーザーのニーズに合った最適なフォーマットでエンドユーザー・コミュニティに配布します。ユーザーはなじみのあるビジネス次元に沿って、データをリアル・タイムでやり取りし、直観的に調査します。こうすることで、思考するスピードで分析を実行できます。

Essbase 製品コンポーネント

Essbase 製品には、大規模なマルチユーザー環境で様々な分析アプリケーションを処理するための強力なアーキテクチャ機能が組み込まれています。図 1 に、Essbase アーキテクチャの 3 層間の情報の流れの概略を示します。クライアント層(左側)には、Oracle Hyperion Smart View for Office、管理サービス・コンソールなどの Essbase サーバー・クライアントが含まれます。中間層(中央)には、Oracle Hyperion Provider Services、Oracle Essbase Administration Services などのサービスが含まれます。データベース層(右側)は、Essbase サーバーで構成されます。クライアント層と中間層の通信、および中間層とデータベース層の通信は、HTTP を経由します。クライアント層とデータベース層の通信は、TCP/IP または HTTP を経由します。中間層とデータベース層でのデータ・ソースとメタデータ・カタログの通信は、ODBC ドライバと JDBC ドライバを経由します。

図 1　製品コンポーネント間の情報の流れの概略



Essbase

Essbase は、対称型マルチプロセッシング・ハードウェア・プラットフォームを利用するマルチスレッド OLAP データベースで、Web に配置可能なシンクライアント・アーキテクチャに基づいています。サーバーは、共有リソースとして動作し、すべてのデータ・ストレージ、キャッシュ、計算およびデータ・セキュリティを処理します。Essbase サーバーのクライアントに必要な処理は、サーバーに存在するデータの取得と表示のみです。

データベース・アウトラインと計算スクリプト、アプリケーション制御、多次元データベースの情報など、すべての Essbase アプリケーション・コンポーネントはサーバーに存在します。Essbase を使用すると、複数のディスク・ドライブにスパンするようにサーバー・ディスク・ストレージを構成して、大規模なデータベースを格納できます。Essbase を使用するには、同時要求を効率的に管理できるように、サーバーでマルチスレッド・オペレーティング・システムを実行する必要があります。また、サーバーでは、アプリケーションに対するすべてのユーザー要求のトラフィック調整機能として動作するサーバー・エージェント・プロセスを実行します。

集約ストレージ・データベースは、ストレージ・データベースをブロックし、データベースの次元性の大幅な増加を可能にする代替方法を提供します。Essbase では、集約ストレージを使用して、幅広い分析のニーズ(財務分析、プランニング、

予算作成、売上高分析、マーケティング分析、サプライチェーン分析、収益性分析)に、単一の分析インフラストラクチャで対応します。

MaxL は、Essbase サーバーの一部である多次元データベースのアクセス言語で、Essbase の管理タスクとメンテナンス・タスクを自動化する柔軟な方法を提供します。

システム要件の詳細は、Oracle Hyperion Enterprise Performance Management System 動作保証マトリックス(http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html)を参照してください。

Essbase をインストールして構成するには、Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

Essbase のサンプル・アプリケーション

Essbase には、Essbase の機能を学習するためのサンプル・アプリケーションのセットと関連データベースが用意されています。また、このドキュメントのほとんどの例は、このサンプル・アプリケーションと関連データベースに基づいています。

サンプル・アプリケーションは、Essbase サーバーとともにインストールされます。

サンプル・アプリケーションの使用方法は、付録 B「サンプル・アプリケーションの設定」を参照してください。

Administration Services

Administration Services は、データベース管理者とシステム管理者用の Essbase へのインタフェースで、複数の Essbase サーバーへの単一点アクセス・コンソールを提供します。管理サービス・コンソールを使用して、複数の Essbase サーバー、アプリケーションおよびデータベースの設計、開発、維持および管理を実行できます。Smart View などのクライアント・アプリケーションを開かずに、コンソールからデータをプレビューできます。また、カスタム Java プラグインを使用して、重要な機能を活用し、拡張できます。

Essbase Studio

Oracle Essbase Studio は、データ・モデリング、キューブの設計および分析アプリケーションの構築に関するタスクを実行するための単一の環境を提供することによって、キューブの構築を簡単にします。Essbase Studio では、ウィザード方式のユーザー・インタフェースを採用し、通常、Essbase アプリケーションを構築するための基となる様々なデータ・ソース・タイプのモデリングをサポートします。

単一の共通のメタデータ・リポジトリ(カタログ)では、企業に構築されたすべての Essbase アプリケーションに関するすべてのメタデータを取り込み、一番低い粒度でメタデータを再利用できます。このカタログによって、Essbase Studio では、企業全体に広がる様々なアプリケーションで共有されている共通のメタデータがあることを認識できます。

Essbase Studio では、複数のドリルスルー・オプションをサポートします: そのオプションは、リレーショナル・データベース、カスタム SQL、URL (Oracle Business Intelligence Enterprise Edition や Oracle Hyperion Financial Data Quality Management の URLなど)、および Java メソッドです。Essbase Studio では、メタデータの関係のグラフィカル表示を使用した系列追跡もサポートします。これにより、ユーザーは、アプリケーションの系列を追跡して、そのメタデータ・コンポーネント、最後にはその系列の元であるデータ・ソースにたどり着くことができます。

Integration Services

オプションの製品コンポーネントである Oracle Essbase Integration Services は、Essbase データベースに保管されているデータと、リレーショナル・データベースに保管されている詳細データの間のギャップを埋めるためのメタデータ駆動環境を提供します。ハイブリッド分析機能によって、ビジネス・ユーザーには意思決定のための詳細情報が、IT マネージャには大規模な分析アプリケーションの設計や維持におけるより高いモジュール性が提供されます。ハイブリッド分析では、Essbase データベースの一部をリレーショナル・データベースに保管することが可能になります。この保管されたリレーショナル・データは、対応する Essbase 階層にマッピングされます。

Provider Services

Provider Services は、Essbase for Java API、Smart View および XMLA クライアントに対する中間層のデータソース・プロバイダです。Provider Services は、同時性の高い分析シナリオをサポートするとともに、Web に対応した分散エンタープライズ環境における拡張性と信頼性を提供します。

Smart View

Smart View は、Essbase、Oracle Hyperion Financial Management、Oracle Hyperion Planning および Oracle Hyperion Enterprise Performance Management Workspace のデータに共通の Microsoft Office インタフェースを提供します。Smart View を使用すると、Microsoft Excel、Word および PowerPoint インタフェースで、データを表示、インポート、操作、配信および共有できます。

アプリケーション・プログラミング・インタフェース(API)

Essbase API (Essbase への開発者のインタフェース)を使用すると、カスタマイズされたアプリケーションを作成できます。『Oracle Essbase API リファレンス』は、API 関数の完全な一覧を提供しています。

開発者用製品

Essbase の開発者用製品を使用すると、ユーザーにプログラミングの知識があるかどうかにかかわらず、カスタマイズされたエンタープライズ分析アプリケーションを迅速に作成、管理および配置できます。

これらの製品(たとえば、Application Builder や Oracle Hyperion Application Builder for .NET)は、包括的な一連のアプリケーション・プログラミング・インタフェース、ドラッグ・アンド・ドロップ・コンポーネントおよびサービスを提供します。

ライフサイクル管理

Oracle Hyperion Enterprise Performance Management System のライフサイクル管理には、Oracle Enterprise Performance Management System 製品のアプリケーション、リポジトリ、または個別のアーティファクトを製品環境およびオペレーティング・システムの間で移行する一貫した方法があります。一般的には、Oracle Hyperion Shared Services Console のライフサイクル管理インタフェースは、ライフサイクル管理をサポートするすべての EPM System 製品に対して一貫しています。ただし、EPM System 製品では、ライフサイクル管理インタフェースでアーティファクトのリスト表示、およびエクスポートとインポートのオプションの表示が異なります。

ライフサイクル管理には、次の機能があります:

- アプリケーションおよびフォルダの表示
- アーティファクトの検索
- アプリケーションから他のアプリケーションへの直接的な移行
- ファイル・システムとの間の移行
- 移行定義ファイルの保存およびロード
- 選択されたアーティファクトの表示
- 移行の監査
- 移行のステータスの表示
- ファイル・システム上でのすばやい変更のための個々のアーチファクトのインポートおよびエクスポート

Shared Services Console のライフサイクル管理インタフェースに加え、ライフサイクル管理ユーティリティと呼ばれるコマンドライン・ユーティリティには、ソースから移行先にアーティファクトを移行させる代替方法を備えています。ライフサイクル管理ユーティリティは、Windows タスク・スケジューラや Oracle Enterprise マネージャなど、サードパーティ製スケジュール・システムともあわせて使用できます。

最後に、ユーザーがライフサイクル管理機能をカスタマイズして拡張できる、ライフサイクル管理アプリケーション・プログラミング・インタフェース(API)があります。

Oracle Hyperion Enterprise Performance Management System ライフサイクル管理の詳細は、『Oracle Hyperion Enterprise Performance Management System ライフサイクル管理ガイド』を参照してください。

主な機能

既存インフラストラクチャとの統合

Essbase 製品は、既存のビジネス・インテリジェンス・インフラストラクチャと統合できます。Essbase 製品を使用すると、情報技術(IT)のオーバーヘッドを最小限に抑えながら、重要なビジネス情報に対する企業ユーザーの分析要求を満足できるため、組織は既存の IT 投資から最大の収益を実現できるようになります:

- 拡張可能なアーキテクチャを提供
- データ・ソース、ハードウェアとオペレーティング・システムのプラットフォーム、アクセス・インタフェースおよび開発言語を幅広くサポート
- ローカル・エリア・ネットワークまたはワイド・エリア・ネットワーク全体、およびイントラネットまたはインターネットにわたって、分析アプリケーションを配置可能

データ統合

Essbase 製品を使用すると、所有しているデータ・ウェアハウス、レガシー・システム、Online Transaction Processing (OLTP)システム、Enterprise Resource Planning (ERP)システム、e-ビジネス・システム、Customer Relationship Management (CRM)アプリケーション、Web ログ・ファイル、その他の外部データ・ソース内のデータを組織で活用できるようになります。データベース統合のために、Integration Services には、グラフィカル・ツールのセット、データ統合サービス、およびリレーショナル・データベースまたはデータ・ウェアハウス環境に結合されたメタデータ・カタログが用意されています。

容易なサーバーおよびデータベース管理

Essbase 製品には、クロスプラットフォームの管理コンソールである管理サービス・コンソールが用意されています。このコンソールを使用すると、Essbase 環境を詳細に制御できます:

- 複数のサーバーとデータベースを管理できます。
- PERL 拡張モジュールを備えた構文言語コマンド・シェルである MaxL を使用して、バッチ・メンテナンスを自動化できます。

Web 環境でミッション・クリティカルなアプリケーション

中間層フレームワークによって、Essbase アプリケーションのための Web に対応した分散プラットフォームが作成されるため、Essbase 製品の機能が拡張されます。それにより、Web ベースの環境で多数のユーザーの分析ニーズに応えることが可能になります。Provider Services によって、クラスタリングおよびフェイルオーバー・サポートが提供されるため、プラットフォームの拡張性と信頼性が向

上するとともに、24 時間 x7 日間稼働環境におけるミッション・クリティカルなアプリケーションがサポートされます。

強力なクエリー

ビジネス・ユーザーの大規模なコミュニティでは、データをリアル・タイムに操作して、業績をすばやく分析できます。Essbase 製品を使用すると、使い慣れたビジネス次元に従ってデータを整理して提供することができ、ユーザーはデータを直感的に表示および調査して、そのデータを実用的な情報に変換することが可能です。

計算

Essbase には、要求の厳しい分析要件を満たすための強力な計算機能が含まれています。豊富な機能ライブラリを使用すると、高度で複雑なビジネス・ロジックや関係を簡単に定義できます。Essbase により、ユーザーはカスタム定義のマクロや関数、複数のデータベースにまたがる計算機能などを使用して、計算機を柔軟に構築、カスタマイズおよび拡張できるようになります。マルチプロセッサ・システムでは、DBA は複数スレッドで計算を実行するための単一の計算要求を構成して、計算速度を向上させることができます。

集約ストレージ・データベースは、ブロック・ストレージ・データベースに代わるデータベースであり、特定のタイプのアプリケーションについては、データベースの集約時間が大幅に短縮されます。

ライトバックとセキュリティ

Essbase には、データ更新とマルチユーザーの再計算を含む、独特のマルチユーザーの読取りおよび書込み機能があります。フロントエンド・ツールを使用しているビジネス・ユーザーは、データをサーバーに書き戻し、計算スクリプトを使用してサーバー上でデータを再計算できます。これは、高度なモデリングおよびプランニング・アプリケーションをサポートするための重要な機能です。

この堅牢なマルチレベルのセキュリティ・モデルにより、サーバーレベル、データベースレベルおよびセルレベルのセキュリティが提供されます。データのアクセス、表示および書込み機能の完全制御は、管理を通じて実現されます。EPM System のセキュリティにより、Oracle Internet Directory や Microsoft Active Directory などの LDAP 対応ディレクトリを含む様々な外部ユーザー・ディレクトリ、およびリレーショナル・データベースとの統合が可能になります。

容易な開発

Essbase には、ユーザーが効果的な多次元アプリケーションを開発するときに役立つ、多くの重要な利点があります。ユーザーには、次のことが可能です:

- グラフィカル・インタフェースを使用してアプリケーションを設計および管理することで、サーバーの機能の大部分を制御できます。

- 次元の追加、計算の変更および階層の変更を迅速に行うことで、新しいビジネスの展開を反映できます。さらに、動的次元ビルダーによって、スプレッドシート、フラット・ファイル、およびサポートされているリレーショナル・データベース・テーブルのデータなどの大量のデータを自動的に定義でき、データベース内に直接、動的にロードできます。
- プログラムを作成することなく、主要な計算を定義できます。
- 個人およびグループに対してセキュリティを定義し、各ユーザーに対する表示と取得手順をカスタマイズできます。その際、プログラミングを行う必要はありません。

2 Essbaseの実装の開始

この章の情報の一部は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 前のバージョンの Essbase からアップグレードする場合は、Oracle Enterprise Performance Management System Installation and Configuration Guide

次の表に、Essbase を実装するときのプロセス・マップを示します。このプロセスの各ステップには概略を記載しています。また、詳細情報を入手できる参照先も記載しています。

表 1　プロセス・マップ

プロセス手順	参照先
Essbase および分散 OLAP の基本について学習します。	• 第 1 章「Essbase について」 • 第 3 章「多次元データベースの理解」 • 第 6 章「アプリケーションとデータベースの作成」 • 第 10 章「属性の操作」 • 第 15 章「パーティション・アプリケーションの設計」 • Essbase のトレーニング・クラスへの参加。担当のソフトウェア・ベンダーにお問い合せください。
自社におけるニーズと要件を判断します。 データ分析のニーズおよび実行する計算とレポート処理を明確にします。	予算、予測およびその他の財務レポートと、それらの改善点。
データを多次元的な視点から分析します: • データ・ソースはどこにありますか? • データはどのようなタイプですか?そのデータは、詳細なリレーショナル・データですか、それとも、分析に利用できる概要的な階層型データですか? • そのデータのフォーマットは何ですか? • データにはどのようにアクセスしますか?リレーショナル・データにアクセスする必要がある場合は、Oracle Essbase SQL インタフェースまたは Integration Services が必要になることがあります。	• 第 3 章「多次元データベースの理解」 • 『Oracle Essbase SQL インタフェース・ガイド』 • ODBC ドライバのドキュメント • Integration Services のドキュメント

プロセス手順	参照先
Essbase をインストールします。	Oracle Enterprise Performance Management System Installation and Configuration Guide
アプリケーションおよびデータベースを設計します。 ● セキュリティなどのビジネス要件およびユーザー要件を明確にします。 ● ソース・データを識別し、Essbase データベースのスコープを決定します。 ● 最下位レベルのメンバー・データをリレーショナル・データベースに残して、ハイブリッド分析でアクセスするか、またはすべてのデータをロードするかを選択します。 ● 標準次元を定義し、疎および密のストレージを指定します。 ● 属性次元でのニーズを明確にします。 ● 異なる通貨のデータを追跡する、通貨換算アプリケーションでのニーズを明確にします。 ● アウトライン次元およびメンバーに必要な計算を定義します。 ● データベースでのデータの変更を監視するためのニーズを特定します。データの変更は、Essbase のトリガー機能を使用して監視します。	第 4 章「ケース・スタディ: 単一サーバー、多次元データベースの設計」
データベースのサイズを見積り、ディスク・スペースを確認して、メモリー内のインデックス・キャッシュ、データ・ファイル・キャッシュおよびデータ・キャッシュ・サイズが十分であることを確認します。	● 付録 A「制限」 ● 付録 D「ディスク要件およびメモリー要件の見積り」
アプリケーションおよびデータベースを作成します。	第 6 章「アプリケーションとデータベースの作成」
通貨アプリケーションを設計します。	第 14 章「通貨換算アプリケーションの設計および作成」
データベースのアウトラインを構築します。	第 7 章「データベース・アウトラインの作成および変更」
メンバーに別名を割当てます。	第 9 章「次元およびメンバーのプロパティの設定」
次元を構築します。データ・ロードによって新規メンバーをアウトラインに取り入れるかどうかを決定します。取り入れる場合は、ルール・ファイルとデータ・ソースを使用して次元を動的に構築することを検討してください。取り入れない場合は、標準のデータ・ロードを設定してください。	● 第 17 章「データ・ロードおよび次元構築の理解」 ● 第 18 章「ルール・ファイルの使用」

プロセス手順	参照先
データをロードします。データは次の方法でロードできます: ● フリーフォーム ● ルール・ファイルの使用 ● ハイブリッド分析の使用	● 第 17 章「データ・ロードおよび次元構築の理解」 ● 第 18 章「ルール・ファイルの使用」 ● 第 19 章「ルール・ファイルを使用したレコード、フィールドおよびデータへの操作の実行」 ● 第 20 章「データ・ロードまたは次元構築の実行およびデバッグ」 ● 第 21 章「高度な次元構築の概念の理解」
データベース計算を行います。 ● 計算のタイプ(アウトラインまたは計算スクリプト、あるいはその組合せ)を決定します ● データベース・アウトライン内のメンバーとメンバー集計の関係が正しいことを確認します。 ● 一部のメンバーを「動的計算」としてタグ付けするか高機能計算を使用することで、計算効率が向上するかどうかを検討します。 ● 計算結果を正しくするために、「2 パス計算」のタグを付ける必要があるメンバーはどれかを検討します。	● 第 22 章「Essbase データベースの計算」 ● 第 23 章「ブロック・ストレージ・データベース用の式の作成」 ● 第 24 章「ブロック・ストレージ・データベース用の式の例の確認」 ● 第 25 章「計算順序の定義」 ● 第 27 章「データ値の動的計算」 ● 第 28 章「時系列データの計算」 ● 第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」 ● 第 30 章「ブロック・ストレージ・データベース用の計算スクリプトの例の確認」 ● 第 31 章「カスタム定義計算マクロの作成」 ● 第 32 章「カスタム定義計算関数の作成」
動的計算、および動的計算を使用してパフォーマンスを向上させる方法について学習します。	第 27 章「データ値の動的計算」
Smart View、その他の Oracle ツールまたはサードパーティ・ツールでデータを表示します。	● Oracle Hyperion Smart View for Office User's Guide を参照してください ● サードパーティ・ツールについては、ベンダーのドキュメントを参照
パーティショニングについて学習します。相互接続された複数のデータベースに、データを分散することのメリットについて検討します。	● 第 15 章「パーティション・アプリケーションの設計」 ● 第 16 章「パーティションの作成および管理」
ファイルまたはセル・ノートをデータ・セルにリンクします。	第 11 章「Essbase データへのオブジェクトのリンク」
データのサブセットをコピーまたはエクスポートします。	第 36 章「データ・サブセットのコピーとデータのエクスポート」
データをバックアップおよび復元します。	『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』

プロセス手順	参照先
ストレージを割当てて、データベースに対する Essbase カーネル設定を指定します。 ● データ圧縮: ディスク上のデータ圧縮および圧縮方式を指定します。 ● キャッシュ・サイズ: インデックス、データ・ファイルおよびデータ・キャッシュのサイズを指定できます。オペレーティング・システムのシャットダウンを防ぐには、サーバー上のアクティブなすべてのデータベースのインデックスとデータ・キャッシュのサイズの合計が、システムの RAM の 3 分の 2 より大きくならないようにします。 ● キャッシュ・メモリーのロック: インデックス・キャッシュ、データ・ファイル・キャッシュおよびデータ・キャッシュで使用されるメモリーを物理メモリー内にロックできます。 ● ディスク・ボリューム: Essbase のインデックス・ファイルとデータ・ファイルのストレージの場所、適切なディスク・ボリューム名、および構成パラメータを指定できます。 ● 分離レベル: コミット・アクセスまたはアンコミット・アクセスを指定します。	● 第 49 章「データベース設定の管理」 ● 第 50 章「ストレージの割当てとデータの圧縮」
レポートを生成します。 ● レポートのタイプ(構造化したレポートまたはフリーフォーム・レポート)を選択します。 ● ページ・レイアウト、列番号、メンバー ID、データ値のフォーマットおよびタイトル・コンテンツなど、レポートの要素をプランします。 ● 構造化したレポートの場合は、ページ、列および行のヘッダーを作成します(フリーフォーム・レポートの場合は必要ありません)。 ● レポート・スクリプトを作成して、テストします。Administration Services のレポート・スクリプト・エディタまたは他のテキスト・エディタを使用します。 ● Essbase サーバー上またはクライアント・コンピュータ上にレポートを保存します。	● 第 33 章「レポート・スクリプトの基本の理解」 ● 第 34 章「レポート・スクリプトの作成」 ● 第 36 章「データ・サブセットのコピーとデータのエクスポート」
データベースのパフォーマンスおよびストレージ設定を調整します。	● 第 49 章「データベース設定の管理」 ● 第 50 章「ストレージの割当てとデータの圧縮」 ● 第 53 章「パフォーマンスの監視」
MaxL または ESSCMD を使用して、定期的な操作を自動化します。	● 第 52 章「MaxL データ定義言語の使用」 ● 付録 E「ESSCMD の使用」

プロセス手順	参照先
データベースに対するセキュリティを設計します。 ● データベース・セキュリティに対するニーズに基づいて、環境のセキュリティ・プランを作成します。 ● ユーザーおよびグループを作成し、必要に応じて、管理者またはデータ・アクセスの権限を割り当てます。 ● サーバー、アプリケーション、データベースまたはデータ・セル・レベルのスコープで、共通のデータ・アクセス権限を定義します。 ● グローバル・アプリケーション権限やデータベース権限を定義するには、関連するアプリケーションまたはアプリケーションとデータベースを選択して設定を調整します。	● 第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」 ● 第 41 章「セキュリティ・フィルタを使用したデータベース・セルへのアクセスの制御」 ● 第 42 章「ネイティブ・セキュリティ・モードでのセキュリティの例」
アプリケーションを維持します。	● 第 46 章「Essbase サーバー、アプリケーションおよびデータベースの実行」 ● 第 47 章「アプリケーションとデータベースの管理」 ● 800 ページの「Essbase ログの使用」 ● 第 49 章「データベース設定の管理」 ● 第 50 章「ストレージの割当てとデータの圧縮」 ● 第 51 章「データの整合性の確保」 ● 『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』
パフォーマンスを分析および改善し、エラー発生時にトラブルシューティングを行います。 ● ブロック・サイズが大きすぎないことを確認します。 ● インデックス・キャッシュ、データ・ファイル・キャッシュ、データ・キャッシュおよび計算機キャッシュに正しいサイズを設定します。 ● データベースを検証して、データの整合性を確保します。 ● 拡張性およびパフォーマンスを向上させるために、複数のキューブにわたってデータを分散させるパーティション化の使用を検討します。 ● アプリケーションを今後拡大できるように十分なディスク・スペースがあることを確認します。 ● Essbase サーバーからデータを定期的にアーカイブします。 ● アーカイブ後にログ・ファイルが更新されるように、スプレッドシート更新ロギングを使用可能にします。 ● 取得中にソートする場合は、取得ソート・バッファのサイズを増やします。	● 第 53 章「パフォーマンスの監視」 ● 第 54 章「Essbase のパフォーマンスの向上」 ● 第 55 章「Essbase キャッシュの最適化」 ● 第 56 章「データベースの再構築の最適化」 ● 第 57 章「データ・ロードの最適化」 ● 第 58 章「計算の最適化」 ● 第 26 章「高機能計算についての理解」 ● 第 59 章「レポートおよび他の取得機能の最適化」

3 多次元データベースの理解

この章の内容

この章の情報の一部は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

OLAP および多次元データベース

オンライン分析処理(OLAP)は、企業データを分析する必要のあるユーザーのための多次元のマルチユーザー対応のクライアント・サーバー型計算環境です。財務部門では、予算設定、アクティビティベースの原価計算(割当て)、財務実績分析および財務モデリングなどの用途に OLAP を使用します。販売部門では、売上の分析と予測に OLAP を使用します。マーケティング部門では、市場調査分析、販売予測、プロモーション分析、顧客分析および市場/顧客の区分に OLAP を使用します。一般的な製造部門での OLAP の用途には、製造のプランニングおよび不具合の分析が含まれます。

こうしたすべての用途で重要な点は、組織の戦略的方向に関する効果的な決定を下すために必要な情報を管理者に提供できるということです。適切に OLAP アプリケーションを使用することで、必要な情報が得られます。つまり、効果的な意思決定のために「ジャストインタイム」の情報が提供されます。

このような情報を提供するためには、基本レベル以上の詳細データが必要です。ジャストインタイムの情報は計算されたデータであり、通常は複雑な関係を反映し、たいていはすぐに計算されます。複雑な関係の分析とモデルングを行う場合は、応答時間が一貫して短くないと実用的であるとはいえません。また、データ関係の性質があらかじめわかっているとは限らないため、データ・モデルが柔軟である必要があります。真に柔軟なデータ・モデルでは、効果的な意思決定のために、OLAP システムは、変化するビジネス要件の必要に応じて対応できます。

OLAP アプリケーションは、多岐にわたる職務分野で広く利用されていますが、すべての分野で次の主要機能が必要です:

- データに対する多次元的な見方
- 計算集約機能

- 時間処理

OLAP システムで鍵となるのは、多次元データベースです。これは、データを集計および計算するのみでなく、様々なデータ・サブセットの取得と計算も行います。多次元データベースでは、複数のデータ・カテゴリ間の関係を分析する必要があるユーザーのため、データ・セットを複数のビューで表示する機能がサポートされています。たとえば、マーケティング・アナリストは、次のような疑問を持ちます:

- 製品 A の先月の販売数はどのくらいだったか?この数字は過去 5 年間の同月の売上と比べてどうか?この製品の支店別、地域別および区域別の販売数はどのようになっているか?
- この製品は、特定の地域でより多く売れたか?また地域によるトレンドはあるのか?
- 去年は顧客から製品 A の返品はあったのか?返品理由は製品の不具合なのか?その製品は特定の工場で製造されたものだったのか?
- 歩合制および価格設定は、販売員による製品販売数に影響を及ぼしたか?製品販売数の多い販売員はいたのか?

多次元データベースでは、データ表示の数は、データベース・アウトライン(データベースのすべての要素を定義する構造)によってのみ制限されます。ユーザーは、データをピボットして様々な視点から情報を表示したり、ドリル・ダウンしてより詳細な情報を探したり、ドリル・アップして概要を表示したりできます。

次元とメンバー

この項では、多次元データベース内のアウトライン、次元およびメンバーの概念を説明します。次元とメンバーについて理解することで、多次元データベースの機能について理解が深まります。

次元は、データベース・アウトライン内の最も高い集計レベルを表します。データベース・アウトラインでは、集計関係を示すツリー構造で次元とメンバーを表示します。たとえば、図 2 で、「Year」は次元(タイプ「Time」)であり、「Qtr1」はメンバーです。

Essbase には、標準次元と属性次元があります。

標準次元は、ビジネス・プランのコア・コンポーネントを表し、たいていは部門機能と関連しています。一般的な標準次元は、時間、勘定科目、製品ライン、市場および部門です。次元はメンバーほどには変更されません。

属性次元は標準次元に関連付けられています。属性次元を使用して、標準次元のメンバーをメンバー属性(特性)に基づいてグループ化して分析します。たとえば、パッケージがガラス瓶のカフェイン抜き製品の収益性と、パッケージが缶のカフェイン抜き製品の収益性を比較できます。

メンバーは、次元の個別のコンポーネントです。たとえば、製品 A、製品 B および製品 C は製品次元のメンバーです。各メンバーには一意の名前があります。Essbase では、メンバー(この章では、保管済メンバーと呼びます)に関連付けられ

ているデータを保管できます。または、ユーザーがデータを取得するときに、そのデータを動的に計算できます。

アウトラインの階層

すべての Essbase データベースの開発は、データベース・アウトラインを作成するところから始まります。この作業では、次のことを行います:

- Essbase データベース内のメンバー間の構造上の関係の定義
- データベース内のデータの配置
- アイテム間の集計および数学的な関係の定義

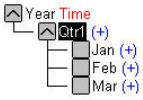
Essbase では、メンバーの概念を使用してデータ階層を表現しています。各次元は 1 つ以上のメンバーで構成されます。各メンバーは、さらに複数のメンバーで構成されている可能性があります。次元の作成時に、個々のメンバーの値の集計方法を Essbase に指定します。データベース・アウトラインのツリー構造では、集計はツリーの分岐内の複数のメンバーを示します。

たとえば、多くの業界では、月ごとにデータを要約し、この月次データをロールアップして四半期の数値を取得します。さらに、四半期のデータをロールアップして年間の数値を取得します。データを郵便番号別、都市別、州別、国別に要約する場合もあります。レポートを作成する目的で、あらゆる次元を使用してデータを集計できます。

たとえば、Essbase サーバーに付属している Sample.Basic データベースの年次元は、四半期のデータを保管する Qtr1、Qtr2、Qtr3、Qtr4 の各メンバーと、1 年の要約データを保管する Year メンバーの 5 つのメンバーで構成されています。Qtr1 は、月ごとのデータを保管する Jan、Feb、Mar の各メンバーと、四半期の要約データを保管する Qtr1 メンバーで構成されています。同様に、Qtr2、Qtr3 および Qtr4 も、月ごとのデータを保管するメンバーと四半期の合計を保管するメンバーで構成されています。

図 2 のデータベース・アウトラインでは、前の段落で説明したように、階層構造を使用して四半期内のデータの集計とその関係を表現しています。

図 2　階層構造



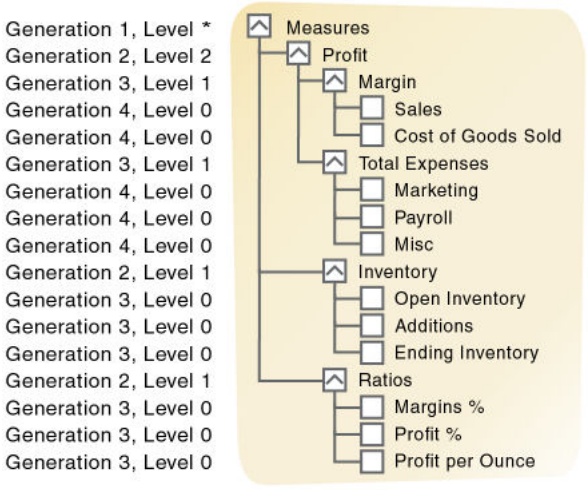
次元によって、比較的少数のメンバーで構成されるものと、非常に多くのメンバーで構成されるものがあります。Essbase では、次元内のメンバー数に制限がないので、必要に応じて新しいメンバーを追加できます。

次元とメンバーの関係

Essbase では、階層を表す用語(世代とレベル、ルートとリーフ)と家系を表す用語(親、子、兄弟、子孫と祖先)を使用して、データベース・アウトライン内のメン

バーの役割と関係を説明します。この項のサブトピックでは、図 3 に示すアウトラインを使用して、メンバーの位置を説明します。

図 3　メンバーの世代番号およびレベル番号



* The level of Measures depends on the branch

親、子および兄弟

図 3 は、次のような親、子および兄弟関係を示しています:

- 親は、下位に分岐を持つメンバーです。たとえば Margin は、Sales と Cost of Goods Sold の親メンバーです。
- 子は、上位に親を持つメンバーです。たとえば Sales と Cost of Goods Sold は、Margin という親メンバーの子です。
- 兄弟は、すぐ上に同じ親を持つ同じ世代の子メンバーです。たとえば、Sales と Cost of Goods Sold は兄弟です(共通の親メンバー Margin を持ちます)。一方、同じ分岐レベルの Marketing は、親が Total Expenses であるため、兄弟ではありません。

子孫および祖先

図 3 は、次のような子孫と祖先の関係を示しています:

- 子孫は、親の下位の分岐に含まれるメンバーです。たとえば、Profit、Inventory および Ratios は Measures の子孫です。また、Profit、Inventory および Ratios の子も、Measures の子孫です。
- 祖先は、あるメンバーの上位の分岐に含まれるメンバーです。たとえば、Margin、Profit および Measures は Sales の祖先です。

ルートおよびリーフ

図 3 は、次のようなルート・メンバーとリーフ・メンバーの関係を示しています:

- ルートは、分岐内の最上位のメンバーです。「Measures」は、「Profit」、「Inventory」、「Ratios」と、「Profit」、「Inventory」および「Ratios」の子のルートです。
- リーフは子を持たないメンバーです。レベル 0 メンバーと呼ぶ場合もあります。たとえば、「Opening Inventory」、「Additions」および「Ending Inventory」はリーフ・メンバーです。

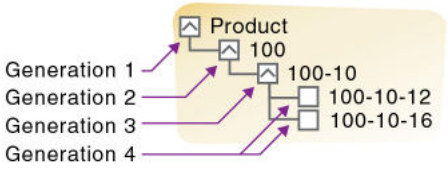
世代とレベル

図 3 は、次の世代レベルを示しています:

- 世代とは、次元内の集計レベルのことです。ツリーのルート分岐は世代 1 です。世代番号はルートからカウントを開始し、リーフ・メンバーに向けて大きくなります。図 3 では、「Measures」は世代 1、「Profit」は世代 2、「Margin」は世代 3 です。各レベルの兄弟はすべて同じ世代に所属します。たとえば、「Inventory」と「Ratios」はどちらも世代 2 です。

  図 4 は、製品次元の一部に世代番号を付けたものです。Product は世代 1、100 は世代 2、100-10 は世代 3、100-10-12 および 100-10-16 は世代 4 です。

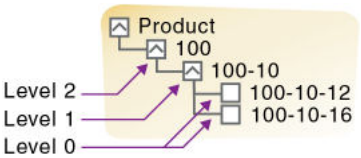
図 4　世代



- レベルも次元内の分岐を表しますが、世代とは使用する番号が逆になります。レベル番号はリーフ・メンバーからカウントし、ルートに向けて大きくなります。ルート・レベルの番号は、分岐の深さによって様々です。図 3 で、Sales と Cost of Goods Sold はレベル 0 です。その他すべてのリーフ・メンバーもレベル 0 です。Margin はレベル 1、Profit はレベル 2 です。Measures のレベル番号は、分岐によって異なります。Ratios 分岐の場合、Measures はレベル 2 です。Total Expenses 分岐の場合、Measures はレベル 3 です。

  図 5 は、製品次元の一部にレベル番号を付けたものです。100 はレベル 2、100-10 はレベル 1、100-10-12 および 100-10-16 はレベル 0 です。

図 5　レベル



世代名とレベル名

レポートのメンテナンスを簡便化するため、世代やレベルに名前を割り当て、この名前をその世代またはレベル内のすべてのメンバーの省略名として使用できま

す。アウトラインの変更は自動的にレポートに反映されます。このため、世代名やレベル名の使用時にメンバー名が変更されたり、データベース・アウトラインから削除されたりした場合、レポートに変更を加える必要はありません。

標準次元および属性次元

Essbase には、標準次元と属性次元があります。この章では、主に標準次元について説明します。なぜなら、Essbase では、属性次元のメンバーにはストレージが割り当てられないからです。属性次元の場合、ユーザーが関連データを要求すると、動的にメンバーの計算が行われます。

属性次元とは、標準次元に関連付けられた特殊なタイプの次元です。第 10 章「属性の操作」を参照してください。

疎次元および密次元

多次元データベースのデータ・セットの多くは、次の 2 つの特性を備えています:

- データは、スムーズかつ均等に分散されるわけではありません。
- 大多数のメンバーの組合せには、データは存在しません。たとえば、すべての製品が国内の全地域で販売されているとは限りません。

Essbase では、アプリケーションの標準次元を密次元と疎次元の 2 タイプに分割することで、パフォーマンスが最大化されます。Essbase では、この分割によりマトリックス方式のデータ・アクセスの長所を活かしながら、スムーズに分散されないデータに対応できます。また、Essbase ではメモリーとディスクの要件を最小限に抑えながら、データ取得時間を短縮できます。

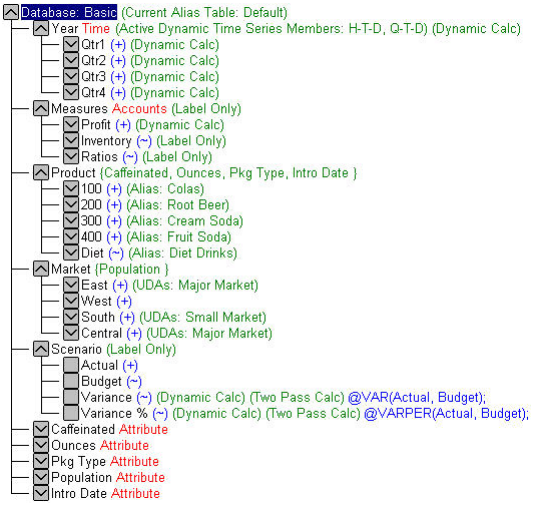
大多数の多次元データベースは本質的に疎であり、大多数のメンバーの組合せにはデータ値がありません。疎次元とは、使用可能なデータ位置にデータが入っているパーセンテージが低い次元です。

たとえば、図 6 内の Sample.Basic データベースのアウトラインには、年、製品、市場、メジャーおよびシナリオ次元が含まれます。製品は製品単位、市場は製品の販売地域、メジャーは勘定科目データを表します。すべての製品がすべての市場で販売されているわけではないので、市場と製品は疎次元になっています。

大多数の多次元データベースには、密次元も含まれます。密次元とは、あらゆる次元の組合せで 1 つ以上のセルに値が入っている可能性が高い次元です。たとえば、Sample.Basic データベースでは、すべての市場のほとんどすべての製品に対して、勘定科目データが存在するので、メジャーは密次元になっています。年とシナリオも密次元です。年は各月、シナリオは勘定科目の値が実績値なのか予算値なのかを表します。

Caffeinated、Intro Date、Ounces、Pkg Type および Population は属性次元です。第 10 章「属性の操作」を参照してください。

図 6　Sample.Basic データベース・アウトライン



密次元と疎次元の選択

ほとんどのデータ・セットでは、既存のデータは予測可能な密と疎のパターンに従います。正確にパターン照合を行うと、非常に疎な多数のデータ・ブロックではなく、ある程度密な適切な数のデータ・ブロックに既存のデータを保管できます。

デフォルトでは、新規次元は疎に設定されています。密次元にするか疎次元にするかを決定するには、Essbase の自動構成機能が役立ちます。

➤ 密次元と疎次元の自動構成の方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元の疎/密の設定」を参照してください。

Essbase では、次の要素に基づいて、疎と密の次元構成の推奨が行われます:

- 次元の時間タグと勘定科目タグ
- データ・ブロックの見積りサイズ
- 指定した次元の特性

推奨構成を適用するか、自動構成機能をオフにして、次元ごとに手動で疎または密のプロパティを設定できます。属性次元は常に疎次元です。属性次元に関連付けられるのは疎な標準次元のみである点に注意してください。

注： 密次元と疎次元の自動構成は推定にすぎません。データベースや複数のユーザーの構成にロードするデータの種類は考慮されません。

Sample.Basic の密/疎の構成

The Beverage Company (TBC)のデータを表す Sample.Basic データベースについて考えます。

TBC はすべての製品をすべての市場で販売しているわけではないので、データ・セットは適度に疎です。製品次元と市場次元のメンバーの組合せの多くには、データ値が存在しません。たとえば、フロリダではカフェイン・フリー・コーラが販

売されていない場合、カフェイン・フリー・コーラ(100-30) ->フロリダ(Florida)の組合せに対するデータ値は存在しません。よって、製品次元と市場次元は疎次元になります。このため、これらの次元内の特定のメンバーの組合せに対してデータ値が存在しない場合、Essbase では、その組合せに対するデータ・ブロックは作成されません。

次に、年次元、メジャー次元、シナリオ次元のメンバーの組合せについて考えます。これらの次元のメンバーの組合せには、ほとんどの場合、データ値が存在します。たとえば、一部の製品は 1 月に販売されるので、Sales->January->Actual のメンバーの組合せにはデータ値が存在します。このため、年次元、メジャー次元、シナリオ次元は密次元になります。

Sample.Basic データベースの標準次元の疎と密の構成は、次のように要約できます:

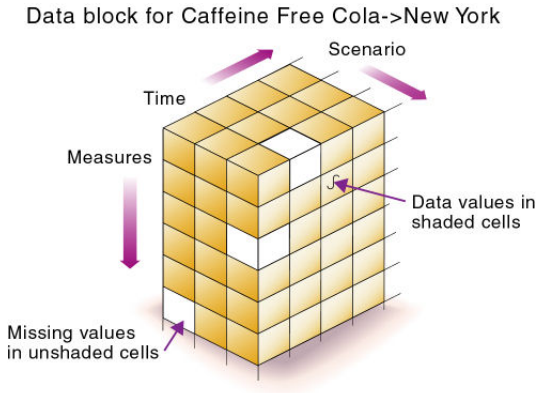
- 疎の標準次元は、製品次元および市場次元です。
- 密の標準次元は、年次元、メジャー次元およびシナリオ次元です。

Essbase では、製品次元と市場次元のメンバーの一意の組合せに対して、データ・ブロックが 1 つずつ作成されます(68 ページの「データ・ストレージ」を参照)。各データ・ブロックは、密次元のデータのみを表します。データ・ブロックには、多くの場合、空のセルがいくつか含まれます。

たとえば、図 7 は、疎メンバーであるカフェイン・フリー・コーラ(100-30)とニューヨーク(New York)の組合せについて説明しています:

- 1 月のこの組合せに対応する勘定科目データ(メジャー次元によって表される)が存在する場合には、2 月の勘定科目データおよび年次元の全メンバーの勘定科目データもおそらく存在するはずです。
- メジャー次元の特定のメンバーに対応するデータ値が存在する場合には、メジャー次元の他のメンバーに対する勘定科目データ値も存在する可能性があります。
- Actual という勘定科目データ値が存在する場合には、Budget という勘定科目データ値も存在する可能性があります。

図 7　Sample.Basic データベースの密なデータ・ブロック



密次元と疎次元の選択のシナリオ

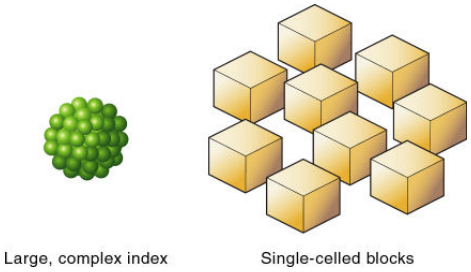
次のシナリオでは、異なる標準次元を選択したときのデータベースに対する影響を示します。これらのシナリオは、7 つ以上の次元と数百個のメンバーを持つ一般的なデータベースを基にしています:

シナリオ 1: 疎の標準次元のみ

すべての次元を疎次元にした場合、Essbase により、データ値を 1 つのみ含むシングル・データ・セルで構成されるデータ・ブロックが作成されます。データ・ブロックごとにインデックス項目が作成されるので、このシナリオの場合は既存の各データ値に対してインデックス・エントリが作成されます。

この構成では、大量のメモリーを必要とするインデックスが生成されます。インデックス・エントリが多いほど、Essbase による特定ブロックの検索の所要時間が長くなります。

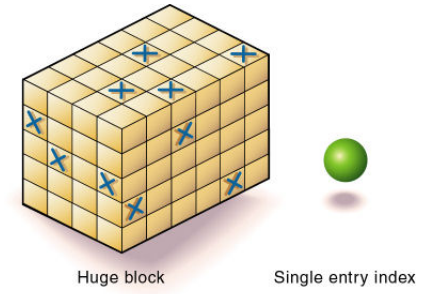
図 8　疎の標準次元のみで構成されたデータベース



シナリオ 2: 密の標準次元のみ

図 9 のようにすべての次元を密次元にした場合、Essbase により、インデックス項目とサイズの大きい疎なブロックが 1 つずつ作成されます。ほとんどのアプリケーションでは、この構成は、他の構成の何千倍ものストレージを必要とします。Essbase では、データ値を検索する際、メモリー全体をロードする必要があるので、膨大な量のメモリーが必要になります。

図 9　密の標準次元のみで構成されたデータベース



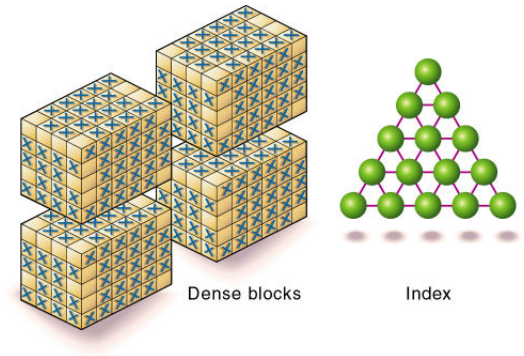
シナリオ 3: 密および疎の標準次元

会社のデータに関する知識に基づいて、疎な標準次元と密な標準次元をすべて識別しました。疎な標準次元と密な標準次元の数がほぼ同じになるのが理想です。

そうでない場合は、おそらく非一般的なデータ・セットを使用することになり、次元を定義する際により細かい調整が必要になります。

図 10 に示すように、Essbase により、メモリーに収容しやすい密なブロックと比較的小さなインデックスが作成されます。データベースは、最小限のリソースを使用して、効果的に実行されます。

図 10　密次元と疎次元の理想的な構成

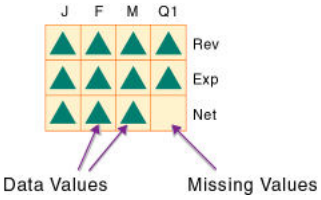


シナリオ 4: 多次元に関する一般的な問題

時間、勘定科目、領域および製品の 4 つの標準次元を持つデータベースがあるとします。次の例では、時間次元と会計次元は密次元、領域次元と製品次元は疎次元です。

図 11 の 2 次元データ・ブロックは、密次元(時間次元と会計次元)のデータ値を表しています。時間次元のメンバーは、J、F、M および Q1 です。勘定科目次元のメンバーは、Rev、Exp および Net です。

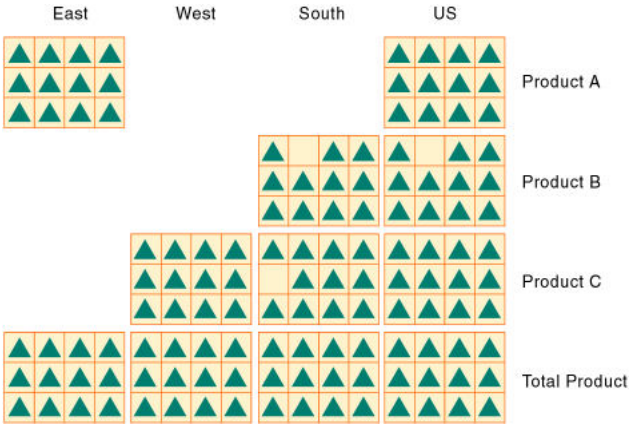
図 11　時間次元と会計次元の 2 次元データ・ブロック



疎な標準次元のメンバーの組合せに対して Essbase でデータ・ブロックが作成されます(メンバーの組合せに対して 1 つ以上のデータ値が存在する場合)。疎次元は領域次元と製品次元です。領域次元のメンバーは、East、West、South および Total US です。製品次元のメンバーは、Product A、Product B、Product C および Total Product です。

図 12 には、11 個のデータ・ブロックが表示されています。West と South の Product A、East と West の Product B、East の Product C にはデータ値がありません。このため、Essbase ではこれらのメンバーの組合せに対してデータ・ブロックが作成されていません。Essbase によって作成されたデータ・ブロックの中には、いくつかの空のセルが含まれます。この例では、すべての疎が効果的にインデックスに集められ、すべてのデータが完全に使用されているブロックに集められています。この構成では、データの格納と取得を効果的に実行できます。

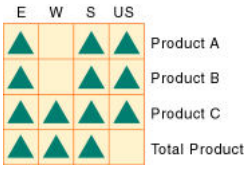
図 12　領域次元と製品次元の疎メンバーに対して作成されたデータ・ブロック



次に、選択した密次元と疎次元の反転について考えます。次の例では、領域次元と製品次元は密次元、時間次元と会計次元は疎次元です。

図 13 では、2 次元のデータ・ブロックは密次元である領域次元と製品次元のデータ値を表しています。West 領域では、Product A および Product B に対するデータは使用できません。US の Total Product に対するデータも使用できません。

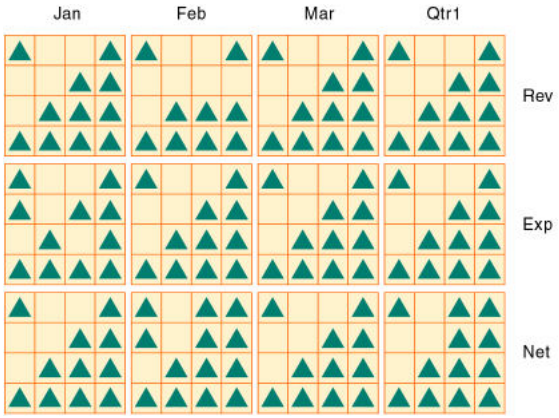
図 13　領域次元と製品次元の 2 次元データ・ブロック



疎な標準次元のメンバーの組合せに対して Essbase でデータ・ブロックが作成されます(メンバーの組合せに対して 1 つ以上のデータ値が存在する場合)。疎次元は時間次元と会計次元です。

図 14 には、12 個のデータ・ブロックが表示されています。時間次元と会計次元のメンバーのすべての組合せに対してデータ値が存在します。このため、Essbase ではすべてのメンバーの組合せに対してデータ・ブロックが作成されます。データ値は、すべての領域のすべての製品に対して存在するわけではないので、データ・ブロックには空のセルも多数含まれます。空のセルが多いデータ・ブロックには、データが効率的に保管されません。

図 14　時間次元と会計次元の疎メンバーに対して作成されたデータ・ブロック



データ・ストレージ

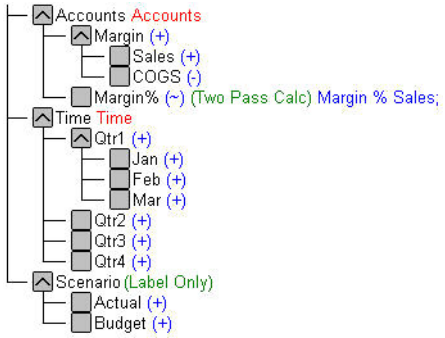
多次元データベース内の各データ値は、単一のセルに保管されます。各標準次元に沿って座標を指定することにより、特定のデータ値を参照できます。

注： Essbase では、属性次元のデータは保管されません。Essbase では、属性次元は、ユーザーがデータを取得する際に動的に計算されます。

図 15 の単純なデータベースについて考えます。このデータベースには、会計、時間およびシナリオの 3 つの次元があります:

- 会計次元には、Sales、COGS、Margin、Margin%の 4 つのメンバーがあります。
- 時間次元には 4 つの四半期メンバーがあり、Qtr1 には 3 つの月メンバーがあります。
- シナリオ次元には予算値を表す Budget と実績値を表す Actual の 2 つの子メンバーがあります。

図 15　多次元データベース・アウトライン



データ値

ある次元の 1 つのメンバーと他の各次元の 1 つのメンバーの交差は、特定のデータ値を表します。図 16 の例には 3 つの次元(勘定科目、時間およびシナリオ)があるため、次元とデータベース内のデータ値をキューブで表すことができます。

図 16　3 次元データベース

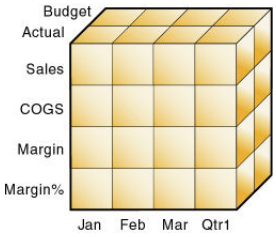
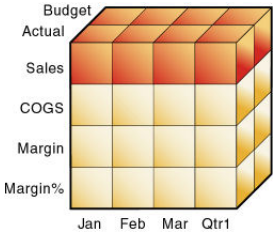


図 17 に示すように、Sales を指定すると、Sales が Actual および Budget と交差する 8 つの Sales 値を含む、データベースのスライスを指定することになります。

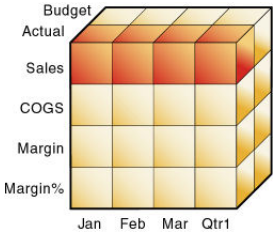
図 17　データベースの Sales スライス



データベースをスライスするということは、1 つ以上の次元をある定数値で固定するということです。その際、固定されていないその他の次元は可変です。

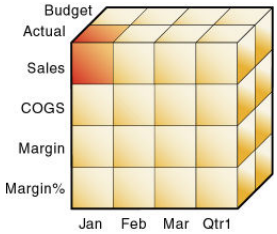
図 18 に示すように、Actual Sales を指定すると、Actual および Sales が交差する 4 つの Sales 値を含む、データベースのスライスを指定することになります。

図 18　データベースの Actual、Sales スライス



データベース内の 1 つのセルにデータ値が 1 つ保管されています。多次元データベース内の特定のデータ値を参照するには、各次元でそのメンバーを指定します。図 19 では、Sales、Jan、Actual のデータ値が含まれるセルが濃い色になっています。このデータ値は、次元間演算子(->)を使用して、Sales -> Actual -> Jan と表現することもできます。

図 19　データベースの Sales -> Jan -> Actual スライス



データ・ブロックおよびインデックス・システム

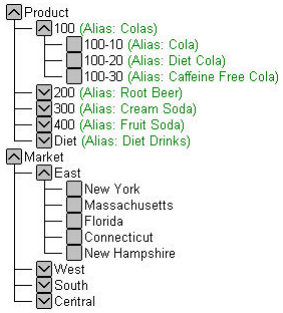
Essbase では、データ・ブロックとインデックス・システムという 2 つのタイプの内部構造を使用して、データの保管およびアクセスを行います。

Essbase では、疎な標準次元メンバーの一意な組合せごとに、データ・ブロックが 1 つ作成されます(その疎次元メンバーの組合せに 1 つ以上のデータ値が存在する場合)。データ・ブロックは、疎次元メンバーの組合せに対するすべての密次元メンバーを表します。

Essbase では、データ・ブロックごとにインデックス項目が 1 つ作成されます。インデックスは、疎な標準次元メンバーの組合せを表します。インデックスには、疎な標準次元のメンバーの一意な組合せ(データ値が 1 つ以上存在)ごとに、対応するエントリが 1 つずつ含まれます。

たとえば、図 20 に示す Sample.Basic データベースのアウトラインでは、「Product」次元と「Market」次元は疎次元です。

図 20　Sample.Basic データベースの「Product」次元と「Market」次元



New York の Caffeine Free Cola のデータが存在する場合、Essbase では、Caffeine Free Cola (100-30) -> New York という疎次元メンバーの組合せに対して、データ・ブロックとインデックス項目が作成されます。Florida で Caffeine Free Cola が販売されていない場合、Caffeine Free Cola (100-30) -> Florida という疎次元メンバーの組合せに対して、データ・ブロックやインデックス・エントリは作成されません。

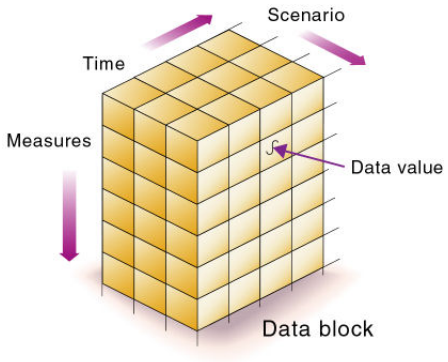
Caffeine Free Cola (100-30) -> New York というデータ・ブロックは、Caffeine Free Cola (100-30) -> New York に対するすべての「Year」次元、「Measures」次元および「Scenario」次元を表します。

一意のデータ値は、特定のデータ・ブロックの特定のセル内に存在すると考えられます。Essbase でデータ値を検索するときは、インデックスによって適切なデータ・ブロックが検索されます。次に、データ・ブロック内で、目的のデータ値を

含むセルが検索されます。インデックス項目は、データ・ブロックへのポインタを提供します。インデックスには既存のデータ・ブロックへのポインタのみが含まれるので、疎なデータを効率的に処理できます。

図 21 は、Sample.Basic データベースのデータ・ブロックの一部を示しています。ブロックの各次元は、Sample.Basic データベース内の密次元(「Time」次元、「Measures」次元および「Scenario」次元)を表します。疎次元である「Product」次元と「Market」次元のメンバーの一意な組合せごとに、データ・ブロックが存在します(この組合せに対してデータ値が 1 つ以上存在する場合)。

図 21　Sample.Basic データベースのデータ・ブロックの一部



各データ・ブロックは多次元配列になっており、この配列には、密次元メンバーの可能な組合せ 1 つ 1 つに対する固定の順序付けられた位置の情報が含まれます。ブロック内のセルにアクセスする際、逐次検索やインデックス検索は行われません。検索はほぼ一瞬で行われ、取得および計算速度が最適化されます。

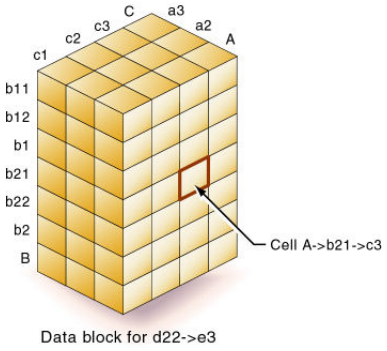
Essbase では、データベース・アウトラインにおける密次元内のメンバーの順序に従って、データ・ブロック中のセルが順序付けされます。

```
   A (Dense)
  a1
  a2
B (Dense)
  b1
    b11
    b12
  b2
    b21
    b22
C (Dense)
  c1
  c2
  c3
D (Sparse)
  d1
  d2
    d21
    d22
E (Sparse)
  e1
  e2
```

e3

図 22 のブロックは、前のデータベース・アウトライン内の疎なメンバー d22 と e3 の組合せに含まれる 3 つの密次元を表します。Essbase では、メンバーの組合せはクロス次元演算子で表されます。次元間演算子の記号は->です。したがって、d22 -> e3 は、d22 と e3 のブロックを指します。A、b21 および c3 の交差は、A -> b21 -> c3 と記述します。

図 22　d22 -> e3 に対する密次元を表すデータ・ブロック



Essbase では、疎次元 D および E のメンバーの一意な組合せごとに、データ・ブロックが作成されます(その組合せに対して 1 つ以上のデータ値が存在する場合のみ)。

図 22 のようなデータ・ブロックには、データ値がないセルが含まれる場合があります。データ・ブロックは、ブロック内に 1 つ以上データ値が存在する場合に作成されます。Essbase では、欠落した値があるデータ・ブロックはディスク上で圧縮され、各データ・ブロックはメモリーに読み込まれた時点で完全に展開されます。データの圧縮はオプションですが、デフォルトでは使用可能になっています。853 ページの「データ圧縮」を参照してください。

密な標準次元と疎な標準次元を慎重に選択することで、データ・ブロックに多数の空のセルが含まれるのを防ぎ、ディスク・ストレージの要件を最小化するとともに、パフォーマンスを改善できます。Essbase では、空のセルは#MISSING データと呼ばれます。

複数のデータ・ビュー

多次元データベースでは、複数のデータ・カテゴリ間の関係を分析する必要があるユーザーのため、データ・セットを複数のビューで表示する機能がサポートされています。データベースを異なる方法でスライスすることで、データの様々なパースペクティブが得られます。たとえば、図 23 の 1 月のスライスでは、年次元が 1 月に固定されたすべてのデータ値を確認できます。

図 23　1 月のデータ

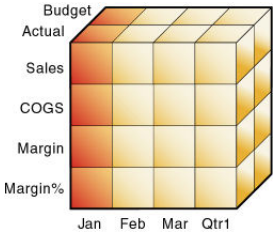


図 24 のスライスでは、2 月のデータを確認できます:

図 24　2 月のデータ

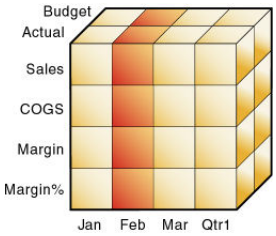
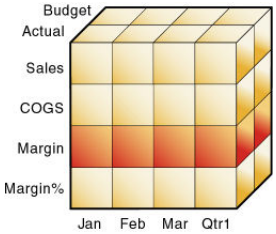


図 25 のスライスでは、収益マージンのデータを確認できます:

図 25　収益マージンのデータ



Essbase ソリューション

最適化された Essbase データベースを作成するには、次の点を確認します:

- 会社でデータを使用する方法
- 次元の構築と順序付けの方法
- 使用するデータ圧縮方式のタイプ
- 計算の作成と順序付けの方法

次のトピックを参照してください:

- 多次元データベースの開発プランニングの詳細は、第 4 章「ケース・スタディ: 単一サーバー、多次元データベースの設計」を参照してください。
- 密次元と疎次元の選択の詳細は、63 ページの「密次元と疎次元の選択」を参照してください。
- データのロードの詳細は、第 17 章「データ・ロードおよび次元構築の理解」を参照してください。

- データの圧縮およびデータベースの最適化の詳細は、853 ページの「データ圧縮」を参照してください。
- データベースの計算の詳細は、第 22 章「Essbase データベースの計算」を参照してください。

4 ケース・スタディ: 単一サーバー、多次元データベースの設計

この章の内容

この章の情報の一部は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

データベース設計のプロセス

多次元データベースを実装するには、Essbase をインストールした後、アプリケーションとデータベースを設計して作成します。データ・ソースを分析し、慎重に要件を定義して、単一サーバーを使用する方法とパーティションに分割する方法のどちらがよりニーズに合っているかを判断します。アプリケーションをパーティションに分割するかどうかを判断する基準として、228 ページの「データベースのパーティション化のガイドライン」を参照してください。

この章では、ケース・スタディを使用して、データベース・プランニング・プロセスの概要を説明します。また、組織向けに単一サーバーによる多次元データベース・ソリューションを設計する際の作業ルールについても説明します。第 6 章「アプリケーションとデータベースの作成」を参照してください。

図 26 は、データベース設計のサイクルのプロセスを示しています。次の基本手順が含まれます:

1. 業務ニーズの分析およびプラン設計。

   ユーザーと組織の情報ニーズを満たすアプリケーションおよびデータベースを作成する必要があります。そのため、ソース・データを特定し、ユーザーの情報アクセスのニーズを定義し、セキュリティの考慮事項を確認して、デー

タベース・モデルを設計します。77 ページの「分析とプランニング」を参照してください。

2. データベース・アウトラインのドラフト作成。

   アウトラインとは、データベースの構造を決めるものです。つまり、保管する情報と、異なる情報同士の相互関連付けを決定します。88 ページの「アウトラインのドラフト作成」を参照してください。

3. システム要件の確認。

   データベースの効率とパフォーマンスは、システム要件を満たす方法およびシステム・パラメータの定義方法によって影響されます。93 ページの「システム要件のチェック」を参照してください。

4. データベースへのテスト・データのロード。

   アウトラインの定義およびセキュリティ・プランの立案が完了したら、データベースにテスト・データをロードし、このプロセス以降の手順を処理できるようにします。93 ページの「テスト・データのロード」を参照してください。

5. 計算の定義。

   アウトラインの集計をテストし、専用の計算のための式および計算スクリプトを記述してテストします。94 ページの「計算の定義」を参照してください。

6. レポートの定義。

   ユーザーは、印刷やオンラインのレポートおよびスプレッドシート、または Web を介してデータにアクセスします。ユーザーに事前定義済のレポートを提供する予定がある場合は、レポート・レイアウトを設計してレポートを実行します。102 ページの「レポートの定義」を参照してください。

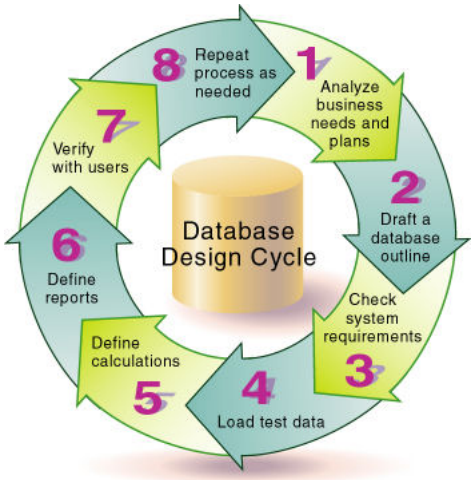
7. ユーザーと合同での確認。

   ユーザーのニーズを満たすデータベースを作成するには、ユーザーに意見を求め、その内容を慎重に検討します。103 ページの「設計の確認」を参照してください。

8. プロセスの繰返し。

   設計を調整するには、手順 1 から 7 を繰り返します。

図 26　データベース設計のサイクル



ケース・スタディ: The Beverage Company

この章では、The Beverage Company(TBC)という名前の架空の会社の要件に基づいてデータベース・プランニング・プロセスを開発し、この TBC を例として、Essbase データベースの構築方法を説明します。TBC は Essbase のインストールに含まれる Sample.Basic アプリケーションのバリエーションです。

TBC は、国内外で清涼飲料水の製造、販売、流通を行っています。TBC のアナリストは、予算予測を策定し、毎月の予算予測と実績を比較します。アナリストが追跡する財務尺度は、損益および在庫です。

TBC では、スプレッドシート・パッケージを使用して予算データを準備し、差異レポートの作成を実行しています。複数の市場で様々な製品のプランニングと追跡を実施しているため、データの取得や分析に時間がかかります。先月、アナリストは勤務時間の大部分をデータの入力とレポートの準備に費やしていました。

TBC は、財務データの中央リポジトリを作成するには、Essbase が最適のツールであると判断しています。データ・リポジトリは、アナリストが組織のどこにいてもアクセスできる 1 台のサーバー上に置かれます。ユーザーはこのサーバーにアクセスして、様々なソースからデータをロードしたり、必要なデータを取得したりできます。なお、TBC には様々なユーザーが存在するため、ユーザーごとにデータ・アクセスのセキュリティ・レベルを変更する必要があります。

分析とプランニング

システムを細かく調整して、ビジネス情報を効果的に分析できるようにするには、Essbase 多次元データベースの設計と操作が鍵となります。多次元データベースのサイズおよびパフォーマンスの変動を考えると、最適化されたデータベースを開発することが重要です。データ・ソース、ユーザーの要件および予想されるデータベース要素をアウトライン化した詳細なプランを立案することで、開発および実装の所要時間を短縮できます。

プランニングと分析のフェーズでは、次のタスクを実行します:

- 78 ページの「ソース・データの分析」

- 79 ページの「ユーザー要件の明確化」
- 79 ページの「マルチ・ユーザー環境におけるセキュリティのプランニング」
- 79 ページの「データベース・モデルの作成」

多次元アプリケーションを設計する場合、次の要因を検討します:

- 社内での情報の流れ: 誰がどのデータを何の目的で使用するか
- 会社が作成するレポートのタイプ: レポートに関するユーザーのニーズを満たすために、アウトラインにどのようなタイプのデータを含める必要があるか

注: アプリケーションごとにデータベースを 1 つのみ定義することで、メモリーの使用率を向上させ、データベース管理を簡易化できます。ただし、オプションの Essbase 通貨換算モジュールを使用するアプリケーションは、この推奨要件に当てはまりません。通貨換算アプリケーションは、通常、メイン・データベースと独立した通貨データベースで構成されます(第 14 章「通貨換算アプリケーションの設計および作成」を参照)。

ソース・データの分析

まずデータベースに含めるソース・データを評価します。データの場所とデータベースのデータを更新する頻度を考慮してください。この事前調査を行うことで、データベースのアウトラインを作成して Essbase データベースにデータをロードするときの時間を節約できます。

データベースのスコープを決定します。膨大な数の製品を含む製品ファミリが多数存在する場合、製品ファミリのデータ値のみを保管できます。各ユーザー部門のメンバーにインタビューを実施して、どのようなデータを処理しているのか、現在はどのような方法でデータを処理しているか、将来どのような方法でデータを処理していきたいかを確認します。

レポートと分析に関するニーズを綿密に定義します。

- ユーザーはどのようなデータの表示方法および分析方法を望んでいますか?
- データベースには、どの程度詳細なデータを保管しますか?
- そのデータは、分析とレポート作成に関する目標を満たしていますか?
- 満たしていない場合、どのようなデータを追加する必要がありますか。また、必要なデータはどこにありますか?

現在のデータの場所を決定します。

- また、各部門では現在どこにデータを保管しているかを確認します。
- Essbase で使用できるフォームのデータですか?
- 各部門では、Windows サーバー上のリレーショナル・データベース、UNIX サーバー上のリレーショナル・データベース、Excel スプレッドシートのうち、どこにデータを保管していますか?
- 誰がどのくらいの頻度でデータベースを更新しますか?

- データを更新する必要があるユーザーは、データにアクセスできますか?

データを Essbase にロードする準備ができていることを確認します。

- データ・ソースは 1 つですか、それとも複数ですか?
- Essbase で使用できるフォーマットのデータですか。Essbase にロード可能な有効なデータ・ソースのリストについては、274 ページの「データ・ソース」を参照してください。
- 使用するすべてのデータが、すぐに使用できる状態になっていますか?

ユーザー要件の明確化

情報の要件についてユーザーと話し合います。ユーザーが使用する情報と、他のユーザーに確認してもらうために作成する必要があるレポートを確認します。次の要件について、判断します:

- ユーザーはどのようなタイプの分析を必要としていますか?
- ユーザーは、どのような要約レベルおよび詳細レベルの情報を必要としていますか?
- アクセス可能な情報を、ユーザーごとに個別に設定する必要がありますか?

マルチ・ユーザー環境におけるセキュリティのプランニング

セキュリティ権限の設定方法をプランするときは、ユーザーの情報の要件を考慮します。分析の最後に、ユーザーと権限のリストを作成します。

次のチェックリストを使用して、セキュリティ・プランを立てます:

- ユーザーは誰ですか? また、ユーザーにどのような権限を与えますか?
- データ・ロード権限を誰に与えますか?
- どのユーザーをグループ化できますか? また、グループ化した場合、そのグループに対して同様の権限を与えることができますか?

第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」を参照してください。

データベース・モデルの作成

次に、データベースのモデルを紙に書き出します。モデルを構築するには、業務にとって重要なパースペクティブとビューを識別します。これらのビューが、データベース・モデルの次元になります。

多くの業務では、次の項目を分析対象としています:

- 期間
- 会計方法

- シナリオ
- 製品
- 流通チャネル
- 地域
- ビジネス・ユニット

次の項目を参照して、情報を収集し判断してください。

分析対象の明確化

業務情報の主要分野を明確にした後、Essbase データベース設計の次の手順は、データベースでどのようにデータを分析するかを決定することです。たとえば、次の内容を検討します:

- 時間で分析する場合、どの期間が必要ですか?分析には現在の年のみを含めますか、複数の年を含めますか?分析には四半期ごとのデータと月ごとのデータを含めますか?季節ごとのデータを含めますか?
- 地理的な地域で分析する場合、地域をどのように定義しますか?地域を販売区域ごとに定義しますか?都道府県や市区町村などの地理的な境界で地域を定義しますか?
- 製品ラインで分析する場合、各製品のデータを確認しますか?データを製品クラスに要約できますか?

業務の分野を問わず、分析に必要なパースペクティブと詳細を決定する必要があります。分析するビジネス領域ごとに、異なるデータ・ビューが提供されます。

次元およびメンバーの決定

各ビジネス・ビューをデータベース内の個々の標準次元として表現できます。分類や属性(製品のサイズや色など)によってビジネス領域を分析する必要がある場合、属性次元を使用して分類ビューを表現できます。

選択した次元によって、データに対して実行できる分析のタイプが決まります。Essbase では、分析のニーズに合わせて必要な数の次元を使用できます。一般的な Essbase データベースの場合、標準次元(非属性次元)が少なくとも 7 つと、さらに多数の属性次元が含まれます。

必要な次元とメンバーをほぼ把握できている場合は、次の項目を検討して、仮のデータベース設計を作成します:

- 81 ページの「次元間の関係」
- 81 ページの「次元とメンバーの構造の例」
- 83 ページの「次元およびメンバーを決定するためのチェックリスト」

データベース・モデルの次元を決定したら、各次元のパースペクティブ内の要素またはアイテムを選択します。これらの要素がそれぞれの次元のメンバーになります。たとえば、時間のパースペクティブには、分析対象の期間(四半期、四半期内の月など)が含まれます。各四半期と各月は、作成した時間を表す次元のメン

バーになります。四半期と月は、メンバーとその子からなる 2 レベルの階層を表します。四半期内の月を集計すると、その四半期の合計が得られます。

次元間の関係

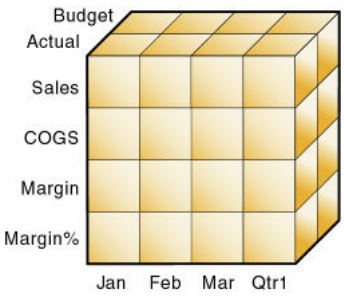
次に、ビジネス領域間の関係について考えます。Essbase データベースは、ユーザーが多くのパースペクティブから情報を簡単に分析できるような構造になっています。たとえば財務アナリストが、次のような疑問を持つとします:

- 特定の月の売上はどのくらいか?この数字は過去 5 年間の同月の売上と比べてどうか?
- 収益マージンの増加率は何%か?
- 実績値と予算値の差異はどのくらいか?

つまり、アナリストは、次の 3 つのパースペクティブから情報を調査する必要があります - 時間、勘定科目およびシナリオ。図 27 のサンプル・データベースでは、これら 3 つのパースペクティブが 3 つの次元として表されています。1 つの次元が 3 つの軸に沿って表現されます:

- 時間次元: Jan、Feb、Mar および Qtr1 の合計で構成され、X 軸で示されます。
- 会計次元: 会計値(Sales、COGS、Margin、Margin%など)で構成され、Y 軸で示されます。
- 別の視点から見るためのもう 1 つの次元(予定値 Budget および実績値 Actual)は、Z 軸で示されます。

図 27　3 つのデータベース次元を表しているキューブ



このキューブ内のセル(メンバーが交差している箇所)には、交差している 3 つのメンバーすべてに対応したデータが含まれています。たとえば、1 月の販売実績などです。

次元とメンバーの構造の例

表 2 は、プランナが次元として定義した TBC のビジネス領域の要約です。次元は、分析対象の主要なビジネス領域を表します。プランナは 3 つの列を作成しています。左端の列に次元、右の 2 列にメンバーが入ります。列 3 のメンバーは列 2 のメンバーのサブカテゴリです。列 3 のメンバーがさらにサブカテゴリ・レベルに分割される場合もあります。たとえばメジャー次元の Margin は、Sales と COGS に分割されます。

表 2　TBC のサンプルの次元

次元	メンバー	子メンバー
年	Qtr1	Jan、Feb、Mar
年	Qtr2	Apr、May、Jun
年	Qtr3	Jul、Aug、Sep
年	Qtr4	Oct、Nov、Dec
メジャー	Profit	Margin: Sales、COGS Total Expenses: Marketing、Payroll、Miscellaneous
メジャー	Inventory	Opening Inventory、Additions、Ending Inventory
メジャー	Ratios	Margin %、Profit %、Profit per Ounce
製品	Colas (100)	Cola (100-10)、Diet Cola (100-20)、Caffeine Free Cola (100-30)
製品	Root Beer (200)	Old Fashioned (200-10)、Diet Root Beer (200-20)、Sarsaparilla (200-30)、Birch Beer (200-40)
製品	Cream Soda (300)	Dark Cream (300-10)、Vanilla Cream (300-20)、Diet Cream Soda (300-30)
製品	Fruit Soda (400)	Grape (400-10)、Orange (400-20)、Strawberry (400-30)
市場	East	Connecticut、Florida、Massachusetts、New Hampshire、New York
市場	West	California、Nevada、Oregon、Utah、Washington
市場	South	Louisiana、New Mexico、Oklahoma、Texas
市場	中部	Colorado、Illinois、Iowa、Missouri、Ohio、Wisconsin
シナリオ	Actual	該当なし
シナリオ	Budget	該当なし
シナリオ	Variance	該当なし
シナリオ	Variance %	該当なし

また、プランナは 2 つの属性次元の追加を行い、サイズ別および容器別に製品を分析できるようにしました:

表 3　TBC サンプルの属性次元

次元	メンバー	子メンバー
オンス	Large Small	64、32、20 16、12
パッケージ・タイプ	Bottle Can	該当なし

次元およびメンバーを決定するためのチェックリスト

モデル・データベースの次元およびメンバーを決定するときには、次のチェックリストを使用します:

- 次元の候補としては何がありますか?
- 他の次元を分類または説明する次元はありますか?そのような次元は、属性次元の候補です。
- ユーザーは次元のビューを限定することを希望していますか?ユーザーが次元を限定するためのカテゴリは、属性次元の候補です。
- メンバーの候補としては何がありますか?
- データにはいくつのレベルが必要ですか?
- データをどのように集計しますか?

データベース設計の分析

初期の次元設計が準備段階の間に、一連のガイドラインに基づいて設計を見直す必要があります。ガイドラインを使用することで、データベースを調整して、多次元技術を活用できるようになります。ガイドラインは、効果的な設計を実現し、集計と計算の目標を達成するのに役立つプロセスまたは課題です。

潜在的データ・ポイントを記述するために必要なメンバーの数によって、次元の数が決まります。次元を削除すべきかどうかわからない場合は、削除を保留にして、その次元を削除すべきか保持すべきかの確信が得られるまで、さらに多くの分析ルールを適用します。

次のトピックの情報を使用して、データベース設計を分析し、改善します。

密次元および疎次元

どの次元が疎で、どの密次元がパフォーマンスに影響するかについて考えます。初めに、62 ページの「疎次元および密次元」を参照してください。91 ページの「パフォーマンスを最適化するためのアウトラインの設計」を参照してください。

標準次元および属性次元

簡単にするため、このトピックの例では、2 つの次元として最初に設計されたものとは別の配置を使用します。すべての次元の組合せに同じ論理を適用できます。

製品を複数の市場の複数の顧客に販売する会社における設計について考えてみます。この場合、市場は各顧客に対して一意です:

```
          Cust A  Cust B  Cust C
New York    100    N/A    N/A
Illinois    N/A    150    N/A
California  N/A    N/A    30
```

Cust A は New York のみ、Cust B は Illinois のみ、Cust C は California のみに存在します。会社は、1 つの標準次元でデータを定義できます:

```
  Market
 New York
  Cust A
 Illinois
  Cust B
 California
  Cust C
```

しかし、データのサンプリング規模を大きくしたところ、多数の顧客が各市場に存在することがわかったとします。たとえば、Cust A と Cust E が New York に、Cust B、Cust M および Cust P が Illinois に、Cust C と Cust F が California に存在するとします。この状況では、一般的に、大きい次元である顧客を標準次元として定義し、小さい次元である市場を属性次元として定義します。市場次元のメンバーを顧客次元のメンバーの属性として関連付けます。市場次元のメンバーは、顧客の場所を表します。

```
  Customer (Standard dimension)
 Cust A (Attribute:New York)
 Cust B (Attribute:Illinois)
 Cust C (Attribute:California)
 Cust E (Attribute:New York)
 Cust F (Attribute:California)
 Cust M (Attribute:Illinois)
 Cust P (Attribute:Illinois)
Market (Attribute dimension)
 New York
 Illinois
 California
```

別の状況について考えます。同様に、製品を複数の市場の複数の顧客に販売する会社ですが、この会社は異なる複数の市場に存在する顧客に製品を発送することがあります:

```
         Cust A  Cust B  Cust C
New York   100    75    N/A
Illinois   N/A    150   N/A
California  150   N/A    30
```

Cust A は New York と California に存在します。Cust B は New York と Illinois に存在します。Cust C は California のみに存在します。この状況で属性次元を使用しても機能しません。顧客のメンバーは、複数の属性メンバーを持つことができません。このため、次のように 2 つの標準次元を使用してデータを設計します:

```
  Customer
 Cust A
 Cust B
 Cust C
Market
 New York
 Illinois
```

```
California
```

次元の組合せ

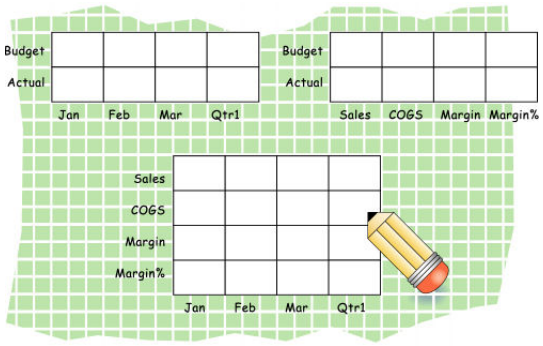
2 つの次元の各組合せを 2 次元マトリックスに配置します。たとえば、TBC の次元の候補(表 2 を参照)には、次の組合せがあります:

- 年とメジャー
- 年と製品
- 年と市場
- 年とシナリオ
- メジャーと製品
- メジャーと市場
- メジャーとシナリオ
- 市場と製品
- 市場とシナリオ
- シナリオと製品
- オンスとパッケージ・タイプ

オンスとパッケージ・タイプは、属性次元として製品次元に関連付けられているので、製品次元と一緒に検討できます。

各次元を視覚的に表すために、マトリックスを作成し、最初の世代のメンバーのいくつかを配置します。図 28 は、3 つの次元の簡単なマトリックスを示しています。

図 28　次元の関係の分析



次元の各組合せに関する 3 つの質問について考えてみます:

- その組合せによって、分析値が追加されますか?
- その組合せによって、レポート作成のユーティリティが追加されますか?
- 使用されない組合せが、過剰に生じないようになっていますか?

組合せごとにこの質問に答えることによって、その組合せがデータベースにとって有効であるかどうかを判断できます。すべての質問の答えが「はい」であるの

が理想です。そうでない場合は、より有効な次元にデータを配置しなおすことを検討します。このプロセスで、必要な情報についてユーザーと話し合います。

アウトラインにおける繰返し

アウトラインにある要素の繰返しは、ほとんどの場合、次元を分割する必要があることを示しています。次の例で、繰返しを避ける方法を説明します。

図 29 では、左側の列「Repetition」では、「Accounts」次元の「Budget」と「Actual」の下に、「Profit」、「Margin」、「Sales」、「COGS」および「Expenses」が繰り返し表示されています。右側の列「No Repetition」では、「Budget」と「Actual」を別の次元(「Scenario」)に分け、「Accounts」次元には「Profit」、「Margin」、「Sales」、「COGS」および「Expenses」メンバーが 1 セットのみ表示されています。この方法によって、アウトラインが単純になり、データベースの他の次元の予算と実績の数値の表示がわかりやすくなります。

図 29　シナリオ次元を作成して繰返しをなくす例

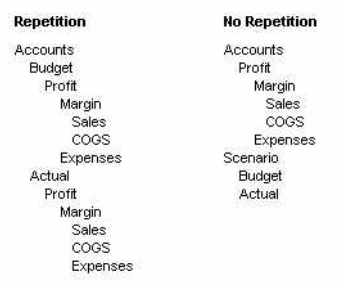


図 30 では、左側の列「Repetition」で、ダイエット飲料を分析するために「Diet」次元の共有メンバーを使用しています。メンバー「100-20」、「200-20」および「300-20」は、「Diet」の下で 1 回、それぞれの親の下で 1 回表示されています。右側の列「No Repetition」では、ブール・タイプ(True または False)の「Diet」属性次元を作成して、アウトラインを単純にしています。すべてのメンバーはそれぞれの親の下に 1 回のみ表示され、該当する属性(「Diet: True」または「Diet: False」)でタグ付けされています。

図 30　属性次元を作成して繰返しをなくす例

属性次元には、その他の分析機能もあります。171 ページの「属性次元の設計」を参照してください。

次元間の無関係性

次元間の無関係性は、ある次元の多数のメンバーが他の次元にとって無関係であるときに起こります。Essbase では、要約(次元)レベルでのみ Essbase によって保管されるデータとして、無関係データを定義します。このような場合、データベースから次元を削除して、そのメンバーを他の次元に追加するか、モデルを個別のデータベースに分けることができます。

たとえば、TBC は、メジャー次元のメンバーとして給与を分析することを考えました。ただし、給与情報は、ほとんどの場合、企業データベースのコンテキストで無関係であることがわかりました。ほとんどの給与は機密情報で、個人に適用されます。個人と給与は一般的に、1 つのセルを表し、他の次元と交差する必要はありません。

TBC は、従業員を個別の次元に分けることを考えました。表 4 は、次元間の無関係性について、TBC が従業員次元の候補を分析した様子を示しています。候補の従業員次元のメンバー(表のヘッダー行)を、メジャー次元のメンバー(左端の列)と比較しています。メジャー次元のメンバー(Revenue など)は All Employees に関係します。Salary メジャーのみが個々の従業員に関係しています。

表 4　次元間の無関係性の例

	Joe Smith	Mary Jones	Mike Garcia	All Employees
Revenue	無関係	無関係	無関係	関係あり
Variable Costs	無関係	無関係	無関係	関係あり
COGS	無関係	無関係	無関係	関係あり
Advertising	無関係	無関係	無関係	関係あり
Salaries	関係あり	関係あり	関係あり	関係あり
Fixed Costs	無関係	無関係	無関係	関係あり
Expenses	無関係	無関係	無関係	関係あり
Profit	無関係	無関係	無関係	関係あり

データベースを分割する理由

個別の従業員情報はデータベースの他の情報とは関係がなく、従業員次元を追加すると必要なデータベース・ストレージが大幅に増えるので、TBC では人事部(HR)データベースを個別に作成しました。新しい HR データベースには、関連する次元のグループが含まれ、給与、福利厚生、保険および 401(k)プランの情報が保管されます。

データベースを分割する理由は様々です。たとえば、ある会社で、タイム・ゾーンの異なる複数の海外の子会社が含まれる組織データベースを運営しているとします。各子会社には、時間依存の財務計算が必要です。財務計算を適切なタイミングで提供できるように、タイム・ゾーンが同じ子会社のグループのデータベースを分割できます。また、パーティション化されたアプリケーションを使用して、子会社ごとに情報を分割できます。

データベース設計分析のチェックリスト

次のチェックリストを使用して、データベース設計を分析します。

- 次元数は最小限に抑えていますか?
- 次元の各組合せに関して、次のことを確認しましたか:
  - その組合せによって、分析値が追加されますか?
  - その組合せによって、レポート作成のユーティリティが追加されますか?
  - 使用されない組合せが、過剰に生じないようになっていますか?
- アウトライン内における繰返しを回避しましたか?
- 次元間の無関係性を回避しましたか?
- 必要に応じてデータベースを分割しましたか?

アウトラインのドラフト作成

Essbase でアプリケーションとデータベースを作成して、アウトラインの最初のドラフトを構築できるようになりました。ドラフトでは、すべての次元、メンバーおよび集計を定義します。アウトラインを使用して、集計の要件を設計し、式と計算スクリプトが必要な場所を特定します。

注： データベースを作成し、そのアウトラインを構築する前に、データベースの追加先の Essbase アプリケーションを作成します。

TBC のプランナは、データベース・アウトラインに関して次のようなドラフトを発行しました。このプランでは、年、メジャー、製品、市場、シナリオ、パッケージ・タイプおよびオンスが次元名です。TBC が集計、計算、式およびレポートの要件をどのように予想したかについて説明します。プランナは、製品を説明するときに製品名ではなく製品コードを使用しています。

- **年。**TBC では、毎月データを収集して、その月次データを四半期または年ごとに集計する必要があります。1 月、2 月、3 月などのメンバーに保管される月次データは、四半期に集計されます。第 1 四半期、第 2 四半期などのメンバーに保管される四半期ごとのデータは、年に集計されます。
- **メジャー。**売上高、売上原価、マーケティング、給与計算、その他、期首在庫、追加および期末在庫が標準のメジャーです。Essbase では、マージン、合計支出、利益、合計在庫、利益%、マージン%および 1 オンス当たりの利益をこのメジャーから計算できます。TBC では、月ごと、四半期ごと、年ごとにメジャーを計算する必要があります。
- **製品。**製品コードは、100-10、100-20、100-30、200-10、200-20、200-30、200-40、300-10、300-20、300-30、400-10、400-20 および 400-30 です。各製品は、それぞれのファミリ(100、200、300 および 400)に集計されます。各製品はオンス属性次元とパッケージ・タイプ属性次元のメンバーに関連付けられているので、TBC では各集計によってサイズ別およびパッケージ別の分析を行えます。

- **市場。**複数の州で 1 つの地域が構成されます。4 つの地域で 1 つの市場が構成されます。州は、コネチカット、フロリダ、マサチューセッツ、ニューハンプシャー、ニューヨーク、カリフォルニア、ネバダ、オレゴン、ユタ、ワシントン、ルイジアナ、ニューメキシコ、オクラホマ、テキサス、コロラド、イリノイ、アイオワ、ミズーリ、オハイオおよびウィスコンシンです。各州は、その地域(東部、西部、南部または中央)に集計されます。各地域は市場に集計されます。
- **シナリオ。**TBC では、予算と実績データを取り出して、追跡します。マネージャは、予算と実績、および予算と実績の差異と差異のパーセンテージを監視して追跡する必要があります。
- **パッケージ・タイプ。**TBC では、製品のパッケージングが売上高と利益に与える影響を確認したいと考えています。パッケージ・タイプ属性次元を作成することで、ユーザーは製品のパッケージ・タイプがビンであるのか缶であるのかに基づいて、製品情報を分析できます。
- **オンス。**TBC では、市場によって異なるサイズ(オンス単位)で製品を販売しています。オンス属性次元を作成することで、ユーザーは各市場の適切な販売サイズを監視できます。

次のトピックでは、次元とメンバー・プロパティの基本を概観し、パフォーマンスに影響を与えるアウトライン設計について説明します。

次元プロパティおよびメンバー・プロパティ

次元とメンバーのプロパティでは、多次元構造の設計での次元とメンバーの役割を定義します。そのプロパティは次のとおりです:

- 次元タイプと属性の関連付け。89 ページの「次元タイプ」を参照してください。
- データ・ストレージ・プロパティ。90 ページの「メンバーのストレージ・プロパティ」を参照してください。
- 集計演算子。94 ページの「次元およびメンバーの集計」を参照してください。
- 式。98 ページの「式および関数」を参照してください。

次元とメンバーのプロパティの詳細なリストは、第 9 章「次元およびメンバーのプロパティの設定」を参照してください。

次元タイプ

次元タイプは、Essbase に用意されているプロパティの 1 つで、次元に特別な機能を追加します。最もよく使用される次元タイプは、時間、勘定科目および属性です。このトピックでは、TBC データベースの次の次元を使用して、次元タイプを説明します。

```
  Database:Design
Year (Type: time)
Measures (Type: accounts)
```

```
Product
Market
Scenario
Pkg Type (Type: attribute)
Ounces (Type: attribute)
```

表 5 で、各 Essbase 次元タイプを定義します。

表 5　次元タイプ

次元タイプ	説明
なし	特定の次元タイプを指定しません。
時間	データをレポートおよび更新する期間を定義します。「時間」というタグを付けることができる次元は 1 つのみです。時間次元によって、期首タイム・バランスや期末タイム・バランスなどの複数の会計次元の関数が使用可能になります。
会計	利益や在庫などの測定するアイテムを含み、Essbase の組込み会計機能を使用可能にします。勘定科目として定義できる次元は 1 つのみです。 2 つの形式の会計次元の計算については、97 ページの「会計次元の計算」を参照してください。
属性	別の関連次元のメンバーを分類する際に使用するメンバーを含みます。 たとえば、Pkg Type 属性次元には、Product 次元のメンバーに適用される容器の各タイプ(ビンや缶など)に対応するメンバーが含まれます。
国	ビジネス活動が行われる場所に関するデータを含みます。国次元では、各メンバーで使用される通貨を指定できます。 たとえば、カナダにはバンクーバー、トロント、モントリオールという 3 つの市場があり、すべての市場で同じ通貨(カナダ・ドル)を使用します。
通貨パーティション	現地通貨メンバーをアプリケーションで定義されている基本通貨から分離します。この次元タイプは、メイン・データベースでのみ使用し、通貨換算アプリケーション専用です。分析用に基本通貨を米国ドルにすることができ、現地通貨メンバーには、各地域の通貨タイプに基づいた値を入れることができます。

メンバーのストレージ・プロパティ

メンバーにデータ・ストレージ・プロパティを指定できます。データ・ストレージ・プロパティでは、集計を格納する場所とタイミングを定義します。たとえば、デフォルトでは、メンバーは、「データの格納」としてタグ付けされます。Essbase では、データの保管メンバーの値を合計し、親レベルでその結果を格納します。

メンバーのデータ・ストレージ・プロパティ・タグを変更することで、各メンバーのデフォルトのロジックを変更できます。たとえば、データの保管メンバーをラベルのみメンバーに変更できます。ラベルのみタグが指定されたメンバーには、データが関連付けられません。

表 6 で、Essbase データ・ストレージ・プロパティがメンバーに与える影響について説明します。

表 6　Essbase データ・ストレージ・プロパティ

データ・ストレージ・プロパティ	メンバーへの影響
データの保管	メンバーはデータを保管します。データの保管はデフォルトのストレージ・プロパティです。
動的計算	メンバーに関連付けられたデータはユーザーによって要求されるまで計算されません。計算されたデータは保管されません。要求の完了後に破棄されます。
動的計算および保管	メンバーに関連付けられたデータはユーザーによって要求されるまで計算されません。計算されたデータは保管されます。
共有メンバー	メンバーに関連付けられたデータは、同じ名前を持つ別のメンバーから取得されます。
共有しない	暗黙的な共有関係が存在する場合に、メンバーに関連付けられたデータは、その親および子に複写されます。
ラベルのみ	ラベルのみメンバーにはデータが関連付けられませんが、値を表示できます。ラベルのみメンバーは、メンバーをグループ化し、ナビゲーションとレポートを容易にします。通常、ラベルのみメンバーは計算されません。 たとえば、メジャー次元のメンバー「比率」に 3 つの子(マージン%、利益%および 1 オンス当たりの利益)が設定されているとします。メンバー「比率」では、メンバーのカテゴリを定義しています。マージン%、利益%および 1 オンス当たりの利益は集計しても比率として意味のある数値にロール・アップされません。このため、比率にラベルのみのタグが付きます。

次元およびメンバー・プロパティのチェックリスト

- 時間次元を明確化できましたか?
- 会計次元を明確化できましたか?
- データに外貨が含まれますか?含まれる場合、通貨パーティション次元を明確化できましたか?
- 別の属性次元として定義するべき次元の特性を明確化できましたか?
- 特殊なデータ・ストレージ・プロパティを必要とするメンバーはどれですか?

パフォーマンスを最適化するためのアウトラインの設計

属性次元をアウトラインの最後に置きます。疎次元の前に密次元を置きます。

アウトライン内の次元の位置および次元のストレージ・プロパティは、パフォーマンスの 2 つの面に影響を及ぼす可能性があります。1 つは計算の実行速度、もう 1 つはユーザーが情報取得に要する時間です。

次のトピックでは、パフォーマンスの最適化の基本について説明します。

クエリーのパフォーマンスの最適化

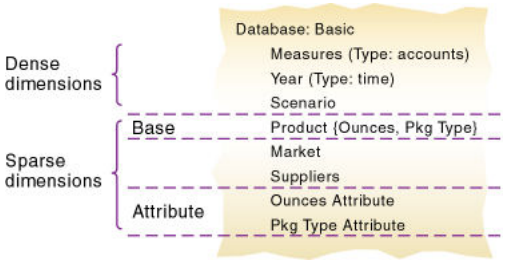
クエリーのパフォーマンスを最適化するには、アウトラインの設計時に、次のガイドラインに従います:

- アウトラインに属性次元がある場合、その属性次元が動的計算疎次元のみであることを確認します。
- アウトラインでは、クエリーの少ない疎次元の前にクエリーの多い疎次元を置きます。

図 31 のアウトラインは、最適なクエリー・パフォーマンスを実現します:

- このアウトラインには属性次元があるので、標準次元およびすべての標準次元メンバーに対するストレージ・プロパティは、データの保管として設定されます。
- 最もクエリーの多い疎次元として、「Product」次元が疎次元の 1 番目に配置されています。「Base」次元は一般的に、他の次元よりも多くクエリーされます。

図 31　クエリー所要時間を最適化するためのアウトラインの設計



計算のパフォーマンスの最適化

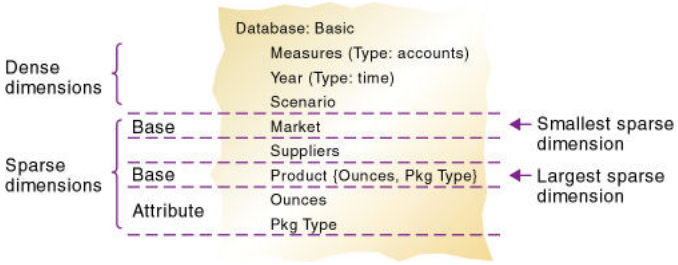
計算のパフォーマンスを最適化するには、アウトライン内の疎次元をメンバー数の順に配列します。その際、最もメンバー数の少ない次元を先頭に配置します。

953 ページの「計算パフォーマンスの設計」を参照してください。

図 32 のアウトラインは、最適な計算のパフォーマンスを実現します:

- 最も小さい疎の標準次元である「Market」は、アウトライン内の疎次元の先頭に配置されています。
- 疎次元の中で最も大きい標準次元である「Product」は、最初の属性次元のすぐ上に配置されています。アウトラインに属性次元がない場合は、「Product」次元がアウトラインの最後になります。

図 32　計算所要時間を最適化するためのアウトラインの設計



計算と取得の両方のニーズへの対応

図 31 と図 32 のアウトライン例には、同じ次元が含まれていますが、この 2 つのアウトラインはそれぞれ異なります。状況に最適なアウトラインの順序を決定するには、データベースで計算を実行するために必要な時間に対するユーザーのデータ取得要件を優先させます。データベースの更新と再計算の頻度はどれくらいですか?ユーザー・クエリーの性質は?ユーザー・クエリーの予想量は?

1 つの回避策として、最初に計算を最適化するために次元を配置します。計算の実行後、取得を最適化するために次元の順序を手動で変更できます。次元の順序を変更した後にアウトラインを保存する場合、データベースをインデックスのみで再構築することを選択します。再び計算を実行する間に、計算を最適化するためにアウトラインの次元の順序を変更します。

システム要件のチェック

ここでは、データベースのシステム要件を確認します。

- 十分なディスク・スペースがあることを確認します。

  1160 ページの「ディスク・スペース要件の決定」を参照してください。

- 十分なメモリーがあることを確認します。

  1175 ページの「メモリー要件の決定」を参照してください。

- キャッシュが正しく設定されていることを確認します。

  第 55 章「Essbase キャッシュの最適化」を参照してください。

テスト・データのロード

計算、集計およびレポートをテストするためには、データベースにデータを用意する必要があります。設計プロセス中に、ダミー・データまたは実際のデータのサブセットをロードすると、柔軟性が生まれ、テストと結果の分析に必要な時間を短縮できます。

データのロードの詳細な指示は、次の章を参照してください:

- 第 17 章「データ・ロードおよび次元構築の理解」
- 第 18 章「ルール・ファイルの使用」
- 第 19 章「ルール・ファイルを使用したレコード、フィールドおよびデータへの操作の実行」

- 第 20 章「データ・ロードまたは次元構築の実行およびデバッグ」
- 第 21 章「高度な次元構築の概念の理解」

予備テストを行い、データベース設計が適切であると判断した場合は、最終的なデータベースに読み込む実際のデータの完全なセットをテスト・ロードします。可能な場合は、テスト・ルール・ファイルを使用します。この最終テストによって、前の設計プロセス・フェーズで予測できなかった、ソース・データの問題が明らかになる可能性があります。

計算の定義

計算は、特定のタイプのデータを導き出すために必要です。計算から導き出されたデータを計算済データと呼び、基本の計算されていないデータを入力データと呼びます。

次のトピックでは、TBC アプリケーションの製品次元とメジャー次元を使用して、多くの Essbase データベースで見られる、一般的な計算のいくつかを紹介します。

ブロック・ストレージ計算の詳細は、次の章を参照してください:

- 第 22 章「Essbase データベースの計算」
- 第 32 章「カスタム定義計算関数の作成」
- 第 26 章「高機能計算についての理解」

次元およびメンバーの集計

標準次元のメンバーを定義すると、Essbase によって自動的に、集計時にメンバーが加算されることを意味する+(加算を表すプラス符号)集計演算子がメンバーにタグ付けされます。メンバー集計プロパティは、必要に応じて、表 7 に示されている演算子のいずれかに変更できます。

集計は、Essbase で最もよく使用される計算です。このトピックでは、製品次元を使用して集計について説明します。

TBC のアプリケーションには、次のようにいくつかの集計パスがあります:

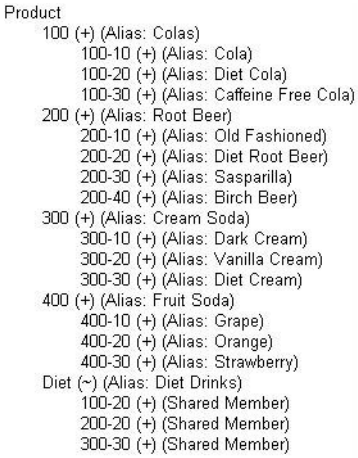
- 各製品をロール・アップしたものが製品ファミリであり、製品ファミリを集計したものが製品です。TBC のアウトラインには複数の集計パスも必要です。一部の製品は複数のカテゴリに集計する必要があります。
- 州をロール・アップしたものが地域であり、地域を集計したものが市場です。
- 月をロール・アップしたものが四半期であり、四半期を集計したものが年です。

次のトピックでは、集計の詳細について説明します:

- 95 ページの「集計における位置および演算子の影響」
- 96 ページの「共有メンバーの集計」
- 96 ページの「集計のチェックリスト」

集計演算子は、Essbase で分岐の各メンバーのデータを親へロール・アップする方法を定義します。たとえば、デフォルトの加算(+)演算子を使用した場合、Essbase では、図 33 に示すように、100-10、100-20 および 100-30 を加算して、その結果を親 100に保管します。

図 33　TBC の製品次元



製品次元の大部分では、加算(+)演算子が使用されています。この演算子は、各メンバー・グループが加算され、親にロール・アップされることを示します。Diet にはチルダ(~)演算子が使用されています。この演算子は、Essbase では親である製品への集計に Diet メンバーが入らないことを示します。Diet メンバーは、共有されるメンバーのみで構成されています。TBC 製品管理グループは、レポートでダイエット飲料を分離することを望んでいるため、TBC は全体の集計に影響を与えない Diet メンバーを個別に作成しました。

集計における位置および演算子の影響

Essbase では、上から下の方向に分岐のデータを計算します。たとえば、加算(+)演算子がタグ付けされた 2 つのメンバーの次に、乗算(*)演算子がタグ付けされた 3 番目のメンバーがあるとします。Essbase では、最初の 2 つのメンバーを加算し、その合計に 3 番目のメンバーを乗算します。

Essbase では、必ず最上位メンバーから集計を始めるので、メンバーの順序とラベルが重要です。145 ページの「異なる演算子を持つ場合のメンバーの計算」を参照してください。

表 7 に、Essbase の集計演算子を定義します。

表 7　集計演算子

集計演算子	説明
プラス記号(+)	デフォルトの演算子です。Essbase によって、分岐のメンバーに対して前に実行された計算の結果にメンバーが加算されます。
ダッシュ(-)	Essbase によって、メンバーに-1 が乗算され、その積が、分岐のメンバーに対して前に実行された計算の結果に加算されます。

集計演算子	説明
アスタリスク(*)	Essbase によって、メンバーに、分岐のメンバーに対して前に実行された計算の結果が乗算されます。
スラッシュ(/)	Essbase によって、分岐のメンバーに対して前に実行された計算の結果が、メンバーで除算されます。
パーセント記号(%)	Essbase によって、分岐のメンバーに対して前に実行された計算の合計が、メンバーで除算されます。その結果に 100 が乗算されます。
チルダ(~)	Essbase では、親への集計にメンバーの値は使用されません。
キャレット(^)	Essbase で、次元の集計にメンバーの値が使用されることはありません。

共有メンバーの集計

共有メンバーも集計パスに影響を与えます。共有メンバーの概念では、2 つのメンバーが同じ名前を使用して同じデータを共有できます。共有メンバーは、他のメンバーに含まれているデータへのポインタを保管するので、Essbase では、データを 1 回のみ保管します。共有メンバーは同じ次元に属する必要があります。データは、複数のメンバーで共有できます。

集計のチェックリスト

次のチェックリストを使用して、集計を定義します:

- アウトラインにおける集計を明確化しましたか?
- 各メンバーに適切な集計演算子のタグを付けましたか?
- 所定のメンバーに対して、共有メンバーのタグを指定しましたか?
- 共有メンバーを(共有ではなく)属性次元内に指定した場合、より効率的になりますか?

メジャー次元の例でのタグおよび演算子

メジャー次元は、時間データと勘定科目データの両方を使用するので、TBC アウトラインで最も複雑な次元です。この次元には、Essbase でのアウトラインの計算に役立つ式と特殊なタグも保管されています。このトピックでは、TBC がメジャー次元(勘定科目タグが付けられた次元)に保管した式とタグについて説明します。

(図 34 にある)TBC によって定義されたメジャー次元のタグをよく見てください。メジャー次元のほとんどのプロパティについては、この章の前のトピックで説明しています(加算(+)、減算(-)および集計なし(~)演算子、勘定科目タグとラベルのみタグ):

- Inventory メンバーおよび Ratios メンバーは、ユーザーによるデータの操作を支援するためのものです。このメンバーにデータは含まれていません。したがって、ラベルのみタグが付いています。

- メジャー次元自体にはラベルのみタグが付いています。メジャーの一部のメンバーには、動的計算タグが付いています。動的計算については、100 ページの「動的計算」を参照してください。
- メジャーの一部のメンバーにはタイム・バランス・タグ(TB First または TB Last)が付いています。タイム・バランス・タグについては、140 ページの「タイム・バランス・プロパティの設定」を参照してください。

図 34　TBC のメジャー次元

```
Measures Accounts (Label Only)
    Profit (+) (Dynamic Calc)
        Margin (+) (Dynamic Calc)
            Sales (+)
            COGS (-) (Expense Reporting)
        Total Expenses (-) (Dynamic Calc) (Expense Reporting)
            Marketing (+) (Expense Reporting)
            Payroll (+) (Expense Reporting)
            Misc (+) (Expense Reporting)
    Inventory (~) (Label Only)
        Opening Inventory (+) (TB First) (Expense Reporting)
        Additions (~) (Expense Reporting)
        Ending Inventory (~) (TB Last) (Expense Reporting)
    Ratios (~) (Label Only)
        Margin % (+) (Dynamic Calc) (Two Pass Calc) Margin % Sales;
        Profit % (~) (Dynamic Calc) (Two Pass Calc) Profit % Sales;
        Profit per Ounce (~) Profit/@ATTRIBUTEVAL(Ounces);
```

会計次元の計算

このトピックでは、勘定科目タグが付けられた次元を計算するための 2 つのフォーム(タイム・バランス・プロパティと差異レポート)について説明します。

タイム・バランス・プロパティ

表 9 のメジャー次元には、TB 期首と TB 期末という 2 つのタグがあります。これらのタグは、タイム・バランス・タグまたはプロパティと呼ばれ、勘定科目タグが付けられた次元でのデータの計算方法に関する指示を Essbase に提供します。タグを使用するには、勘定科目タグが付けられた次元と時間タグが付けられた次元が必要です。期首、期末、平均および支出タグは、会計次元メンバーでのみ使用できます。

TBC メジャー次元では、期首在庫データは、各月の初めに TBC が持っている在庫を表します。期首在庫の四半期ごとの値は、四半期の期首の値に等しいです。期首在庫には、タイム・バランス・タグ「TB 期首」が必要です。

期末在庫データは、各月の終わりに TBC が持っている在庫を表します。期末在庫の四半期ごとの値は、四半期の期末の値に等しいです。期末在庫には、タイム・バランス・タグ「TB 期末」が必要です。表 8 では、会計次元のタイム・バランス・タグを定義しています。

表 8　勘定科目のメンバーのタグ

タグ	説明
タイム・バランス期末	最後の子メンバーの値が親に渡されます。たとえば、3 月が第 1 四半期に渡されます。
タイム・バランス期首	最初の子の値が親に渡されます。たとえば、1 月が第 1 四半期に渡されます。

表 9 で、第 1 四半期(右から 2 番目の列)と年(一番右側の列)は、会計次元のタイム・バランス・プロパティが時間次元の集計に与える影響を示しています。データは第 1四半期分のみが表示されています。

表 9　タイム・バランス・プロパティが TBC の集計に及ぼす影響

次元	1 月	Feb	Mar	第 1 四半期	年
勘定科目メンバー 1	11	12	13	36	Qtr1 + Qtr2 + Qtr3 + Qtr4
勘定科目メンバー 2 (TB 期首)	20	25	21	20	20
勘定科目メンバー 3 (TB 期末)	25	21	30	30	Qtr4 の値

通常、時間次元の親の計算は、その親の子の集計と式に基づきます。ただし、勘定科目の分岐のメンバーが「TB 期首」とマークされている場合、時間次元のすべての親は、「TB 期首」とマークされているメンバーに合わせます。

例については、140 ページの「タイム・バランス・プロパティの設定」を参照してください。

差異レポート

TBC の Essbase 要件の 1 つは、実績データと予算データの差異レポートを実行できることです。差異レポート計算では、会社の支出を表すすべてのアイテムに支出レポート・タグを付ける必要があります。在庫メンバー、合計支出メンバーおよび COGS メンバーにそれぞれ、差異レポート用の支出レポート・タグを付けます。

Essbase には、支出と支出外という 2 つの差異レポート・プロパティが用意されています。デフォルトは支出外です。差異レポート・プロパティでは、メンバーの実績データと予算データの差異を、そのメンバー式の@VAR 関数または@VARPER 関数を使用して Essbase で計算する方法を定義します。

メンバーに「支出」のタグを付けた場合、@VAR 関数では予算から実績を減算します。たとえば、予算額が$100 で、実績の金額が$110 の場合、その差異は-10 です。

支出レポート・タグを付けなかった場合、@VAR 関数では実績から予算を減算します。たとえば、予算額が$100 で、実績の金額が$110 の場合、その差異は 10 です。

式および関数

式は、データベース・アウトラインのメンバー間の関係を計算します。式は、アウトラインのメンバーに適用するか、計算スクリプトに置くことができます。このトピックでは、式を使用して TBC がどのようにパフォーマンスを最適化したかについて説明します。

関数は、専用の計算を実行し、メンバーのセットまたはデータ値のセットを戻す、事前定義ルーチンです。式は、演算子、関数、次元名、メンバー名および数値定数で構成されます。

Essbase では次の演算子をサポートします:

- 算術演算を実行する算術演算子
- 論理条件を計算に組み込む条件演算子
- データベース・メンバーの特定の組合せのデータ値を示す次元間演算子

Essbase の関数には、Essbase の計算機能を拡張するための 100 を超える事前定義ルーチンが組み込まれています。Essbase では次の関数をサポートします:

- ブール関数。この関数は TRUE または FALSE の値を戻すので、条件付テストを実行できます。
- 数学関数。この関数は専用の数学計算を実行します。
- 関係関数。この関数は、現在のメンバーの位置に基づいて、計算時にデータベース内のデータ値を調べます。
- 範囲関数。この関数は、メンバーの範囲を、別の関数またはコマンドに対する引数として宣言します。
- 財務関数。この関数は専用の財務計算を実行します。
- メンバー・セット関数。この関数は、指定されたメンバーに基づいており、メンバーのリストを生成します。
- 割当て関数。この関数は、親レベルで入力された値を子メンバーに割り当てます。
- 予測関数。この関数は、データの平滑化、データの補間または将来の値の計算のために、データを操作します。
- 統計関数。この関数は高度な統計値を計算します。
- 日付関数および時刻関数。この関数は、計算式の中で日付特性および時刻特性を使用します。
- 計算モード関数。この関数は、式を計算するために Essbase で使用する計算モードを指定します。

メジャー次元では、次の式を使用します:

- Margin = Sales - COGS
- Total Expenses = Marketing + Payroll + Miscellaneous
- Profit = Margin - Total Expenses
- Profit % = Profit % Sales
- Margin % = Margin % Sales
- Profit per Ounce = Profit / @ATTRIBUTEVAL(@NAME(Ounces))

Essbase では、集計演算子を使用して、Margin、Total Expenses および Profit メンバーを計算します。Margin%式では、%演算子を使用しています。これは、「Margin を Sales のパーセンテージで表す」を意味します。Profit%式でも同じ%演算子を使用しています。Profit per Ounce 式では、除算演算子(/)と関数(@ATTRIBUTEVAL)を使用して、オンス単位で作られた製品の 1 オンス当たりの収益性を計算します。

注： Profit per Ounce の式では、@ATTRIBUTEVAL 関数の文字列 Ounces を処理するために、@NAME 関数も使用しています。

演算子、関数および構文の詳細なリストは、『Oracle Essbase テクニカル・リファレンス』を参照してください。第 23 章「ブロック・ストレージ・データベース用の式の作成」も参照してください。

動的計算

データベース全体の計算を設計するときに、メンバーを動的計算メンバーとして定義できます。メンバーに動的計算タグを付けると、Essbase では、通常のデータベース計算時にメンバーの組合せを事前に計算するのではなく、データの取得時にメンバーの組合せを計算します。動的計算によって、通常のデータベース計算にかかる時間は短くなりますが、動的に計算されたデータ値を取得する時間が増える可能性があります。

図 35 で、TBC のメジャー次元のメンバー Profit、Margin、Total Expenses、Margin %および Profit %に、動的計算タグが付けられています。

図 35　TBC のメジャー次元における動的計算タグ

```
Measures Accounts (Label Only)
    Profit (+) (Dynamic Calc)
        Margin (+) (Dynamic Calc)
            Sales (+)
            COGS (-) (Expense Reporting)
        Total Expenses (-) (Dynamic Calc) (Expense Reporting)
            Marketing (+) (Expense Reporting)
            Payroll (+) (Expense Reporting)
            Misc (+) (Expense Reporting)
    Inventory (~) (Label Only)
        Opening Inventory (+) (TB First) (Expense Reporting)
        Additions (~) (Expense Reporting)
        Ending Inventory (~) (TB Last) (Expense Reporting)
    Ratios (~) (Label Only)
        Margin % (+) (Dynamic Calc) (Two Pass Calc) Margin % Sales;
        Profit % (~) (Dynamic Calc) (Two Pass Calc) Profit % Sales;
        Profit per Ounce (~) Profit/@ATTRIBUTEVAL(Ounces);
```

データベース全体の計算を実行すると、動的計算メンバーとそれに対応する式は、計算されません。これらのメンバーは、ユーザーが、たとえば Smart View から要求したときに計算されます。Essbase では計算された値は保管されません。値は、次の取得時に再計算されます。ただし、最初の取得後に動的に計算された値を保管することを選択できます。

データ値を動的に計算すべきか否かを決定する際、次の点に関する優先度を検討してください:

- 定期的な計算時間(バッチ計算)の最適化
- ディスク・スペースの使用率の低下
- データベースの再構築時間の短縮
- ユーザーがデータ取得に要する時間の短縮
- バックアップ時間の短縮

第 27 章「データ値の動的計算」を参照してください。

2 パス計算

TBC データベースでは、Margin %と Profit %にラベル 2 パスが付いています。このデフォルトのラベルは、一部のメンバー式を 2 回計算して、目的の値を算出する必要があることを示します。2 パス・プロパティは、勘定科目タグが付けられた次元のメンバー、動的計算タグまたは動的計算および保管タグが付けられたメンバーに対してのみ作用します。

次の例では、(式 Profit % Sales に基づく) Profit %に 2 パス・タグが付いている理由を説明します。表には、5 つの列(列ヘッダーは左から「次元」、「Jan」、「Feb」、「Mar」、「Qtr1」)と 3 つの行(「Profit」、「Sales」、「Profit %」)があります。Jan、Feb、Mar および Qtr1 は、年次元のメンバーです。Profit、Sales および Profit %は、メジャー(勘定科目)次元のメンバーです。

表 10 では、Essbase にロードする初期データを定義しています。Profit -> Jan、Profit -> Feb および Profit -> Mar のデータ値は 100 です。Sales -> Jan、Sales -> Feb および Sales -> Mar のデータ値は 1000 です。

表 10　Essbase にロードされたデータ

次元	Jan	Feb	Mar	Qtr1
Profit	100	100	100	該当なし
Sales	1000	1000	1000	該当なし
Profit %	該当なし	該当なし	該当なし	該当なし

まず、Essbase では、メジャー次元を計算します。表 11 の Profit % -> Jan、Profit % -> Feb および Profit % -> Mar のデータ値は 10%です。

表 11　Essbase でメジャー次元を計算後のデータ

次元	Jan	Feb	Mar	Qtr1
Profit	100	100	100	
Sales	1000	1000	1000	
Profit %	10%	10%	10%	該当なし

次に、Essbase では、年次元を計算します。データは、次元全体でロール・アップされます。表 12 の Profit -> Qtr1 (300)および Sales -> Qtr1 (3000)のデータ値は正しいです。Profit %に 2 パス計算のタグが付けられているので、Profit % -> Qtr1 (30%)のデータ値は正しくありません。

表 12　Essbase で年次元を計算後のデータ

次元	Jan	Feb	Mar	Qtr1
Profit	100	100	100	300
Sales	1000	1000	1000	3000
Profit %	10%	10%	10%	30%

Essbase では、メンバー Profit %の出現ごとに利益パーセンテージを再計算します。表 13 の Profit % -> Qtr1 (10%)は、2 回目のパスが実行された後で正しくなります。

表 13　Essbase で利益パーセンテージを再計算後のデータ

次元	Jan	Feb	Mar	Qtr1
Profit	100	100	100	300
Sales	1000	1000	1000	3000
Profit %	10%	10%	10%	10%

計算のチェックリスト

次のチェックリストを使用して、計算を定義します:

- デフォルトの計算ロジックで正確な結果が出ますか?
- 式が必要なメンバーはどれですか?
- 「タイム・バランス」タグが必要なメンバーはどれですか?
- 差異レポートが必要なメンバーはどれですか?
- 2 パス計算が必要なメンバーはどれですか?
- 「動的計算」タグを付けられるメンバーはどれですか?

注： Essbase のトリガー機能によって、データベース内のデータ変更の効率的な監視が可能になります。117 ページの「トリガー定義の理解」を参照してください。

レポートの定義

設計が確実にユーザー情報の要件を満たすようにするには、データをユーザーに表示されるとおりに表示する必要があります。ユーザーは通常、スプレッドシート、印刷されたレポートまたは Web 上で公開されたレポートを通してデータを表示します。オラクル社およびそのパートナーは、ユーザーが使用するレポート作成システムを生成するための多くのツールを提供しています。

いくつかのツールは、データの表示やフォーマットをすばやく行ったり、データベース設計がユーザーのニーズを満たしているかどうかをテストしたりするのに役立ちます。管理サービス・コンソールにあるレポート・スクリプト・エディタを使用すると、レポート・スクリプトをすばやく記述できます。スプレッドシートに精通しているユーザーは、Smart View を使用できます(この場合は Provider Services が必要です)。

設計フェーズでは、次の項目を確認してください:

- データのグループ化と順序付け。設計で使用可能になった交差によって、ユーザーが必要なデータが提供されますか?
- 合計のレベル。たとえば、アウトライン設計の階層をドリル・ダウンおよびドリル・アップする Smart View のユーザーに必要な集計レベルは何ですか?

- 属性のレポート。データベース設計によって、特定の次元またはメンバーの特性または属性に基づいた分析が容易になりますか?たとえば、16 オンスのビン詰めコーラの売上高と 32 オンスのビン詰めコーラの売上高の比較など、サイズやパッケージングの特定の組合せで売上高を比較する必要がありますか?

事前に指定された使用レポートを作成したり、テスト・データに対してテストしたりする場合は、必ず適切なツールを使用するようにしてください。設計したレポートによって、元の目的を満たす情報が提供される必要があります。レポートを使いやすくして、データの適切な組合せと、適切なデータ量が提供されるようにしてください。多数の列や行が含まれているレポートは使用しにくいため、すべてを包含した 1 つのレポートではなく、複数のレポートを作成することが必要になる場合があります。

設計の確認

データを分析して予備の設計を作成した後、ユーザーとともに設計のすべての側面をチェックします。データベースがユーザーの分析やレポート作成のニーズを満たしていることはすでに確認しているはずです。データベースが、ユーザーのすべての目標を満たしていることを確認します。

計算は、ユーザーが必要としている情報を提供していますか?ユーザーはレポートをすばやく生成できますか?ユーザーは集計回数に満足していますか?つまり、データベースがユーザーのために機能しているかどうかをユーザーに尋ねます。

設計サイクルの終了近くになったら、実際のデータを使用してテストします。アウトラインは正常に構築されますか?すべてのデータがロードされますか?データベースがいずれかの領域で失敗した場合は、設計サイクルの手順を繰り返して問題の原因を識別します。

Essbase によって、問題の分離に役立ついくつかの情報源が提供されます。これらの情報源には、アプリケーションや Essbase サーバーのログ、例外ログ、Administration Services からアクセス可能なデータベース情報などが含まれます。問題に関連するドキュメントのトピック、たとえば、セキュリティ、計算、レポートまたは一般的なエラー・メッセージに関するトピックを参照してください。このガイドの索引を使用すると、問題の解決に役立ちます。トラブルシューティング、ログ、最適化、パフォーマンス、リカバリ、リソース、エラー、警告などの用語を探してください。

理想的なデータベース・ソリューションに到達するには、設計プロセスの手順を何度か繰り返す必要もあると思われます。

5 Administration Servicesについて

この章の内容

はじめに

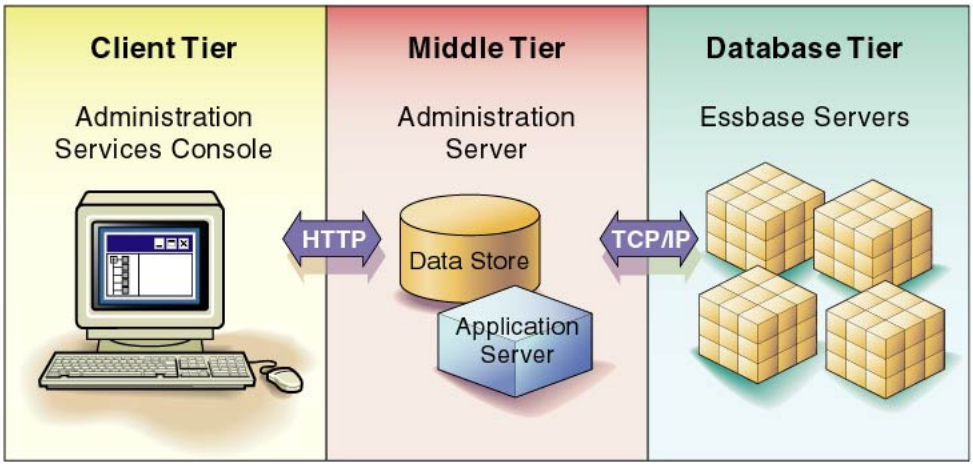
Administration Services は、Essbase のためのクロスプラットフォームの管理ツールです。Administration Services は、Essbase 管理サーバーと呼ばれる Java 中間層サーバーと、管理サービス・コンソールと呼ばれる Java クライアント・コンソールで構成されています。

管理サービス・コンソールは、管理者が Essbase 環境を、エンタープライズ・ビューと呼ばれる 1 つのナビゲーション・ツリーから管理できるようにするグラフィカル・ユーザー・インタフェース(GUI)です。このコンソールでは、ウィザードやエディタのほか、管理者が Essbase サーバーの固有のセットを表示、管理および維持するのに役立つその他のツールが提供されます。このコンソールには、コンソールから別のプログラムに切り替えなくてもデータをプレビューできるデータ・プレビュー・グリッドが含まれています。

Administration Services のアーキテクチャ

Administration Services は、クライアントのユーザー・インタフェース(UI)、中間層サーバー、1 つ以上の Essbase サーバーから構成される 3 層システムで Essbase サーバーと連携して動作します。中間層は、UI と Essbase サーバーの間の対話およびリソースを調整します。これらの各層(同じコンピュータまたはプラットフォーム上に存在する場合と存在しない場合があります)は、下の図に示す次のコンポーネントを含んでいます:

図 36　Administration Services のアーキテクチャ



- クライアント層(管理サービス・コンソール): Java ベースのクライアント・コンソールが、Essbase 環境を管理するための UI を提供します。
- 中間層(Essbase 管理サーバー): Java ベースのサーバーが、Essbase サーバーに接続するための通信、セッションおよびセキュリティ情報を保持しています。
- データベース層(Essbase サーバー): 1 つ以上の Essbase サーバーが、多次元データベース情報を保管および処理します。Essbase サーバーは、Administration Services とは別にインストールされます。

Administration Services の配置

Administration Services は、様々なシナリオで配置できます。たとえば、UNIX を実行しているコンピュータに Essbase サーバーをインストールし、Windows を実行しているコンピュータに Essbase 管理サーバーと管理サービス・コンソールをインストールしたり、Essbase 管理サーバーと管理サービス・コンソールを別のコンピュータやプラットフォームにインストールしたりできます。また、中間層の Essbase 管理サーバーは、既存のフレームワーク内での特定のサードパーティ製品(たとえば、アプリケーション・サーバーやリレーショナル・データベース)の置き換えもサポートしています。

Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

Administration Services の開始

Administration Services を開始するには、Essbase 管理サーバーを開始してから、管理サービス・コンソールを開始します。Oracle Essbase Administration Services Online Help の「Administration Services の開始」を参照してください。

Administration Services ユーザーについて

既存の Essbase ユーザーは、Essbase 管理サーバー上のユーザーとしても作成されるまで、Administration Services を使用できません。ユーザー設定ウィザードを使用すると、Essbase 管理サーバー・ユーザーを手順に従って作成し、それらのユーザーを適切な Essbase サーバーに関連付けられます。Essbase 管理サーバー上の Excel スプレッドシート・ユーザーを作成する必要はありません。

➤ Administration Services ユーザーを作成するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「ユーザー設定ウィザード」を参照してください。

Administration Services への接続

Administration Services では、個々の Essbase サーバーへの接続は中間層の Essbase 管理サーバーによって処理されます。Administration Services を開始したときに Essbase サーバーが開始された場合は、エンタープライズ・ビューに追加した各 Essbase サーバーに自動的に接続されます。『Oracle Essbase Administration Services オンライン・ヘルプ』の「Oracle Essbase Administration Services Online Help の接続およびポートについて」を参照してください。

管理サービス・コンソールから、Essbase サーバーの異なるインスタンスに同時に接続できます。

Administration Services のユーザー名とパスワードが、Essbase サーバーのユーザー名とパスワードとは異なる可能性があります。Administration Services のユーザー名とパスワードがわからない場合は、Administration Services 管理者に確認してください。

Administration Services に最初に接続した後、ユーザー設定ウィザードを使用して Administration Services ユーザーを作成し、各ユーザーのエンタープライズ・ビューに Essbase サーバーを追加できます。『Oracle Essbase Administration Services オンライン・ヘルプ』の「Oracle Essbase Administration Services Online Help への接続」を参照してください。

エンタープライズ・ビューへの Essbase 管理サーバーの追加

Administration Services に接続するたびに、選択した Essbase 管理サーバーが、左のナビゲーション・ペイン内のナビゲーション・ツリーであるエンタープライズ・ビューに表示されます。各ユーザーは、エンタープライズ・ビューに Essbase 管理サーバーの固有のセットを追加できます。

➤ エンタープライズ・ビューに Essbase 管理サーバーを追加するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「エンタープライズ・ビューへの Essbase 管理サーバーの追加」を参照してください。

エンタープライズ・ビューへの Essbase サーバーの追加

Administration Services に接続するたびに、選択した Essbase サーバーが、左のナビゲーション・ペイン内のナビゲーション・ツリーであるエンタープライズ・ビューに表示されます。各ユーザーは、エンタープライズ・ビューに Essbase サーバーの固有のセットを追加できます。次の方法を使用して、エンタープライズ・ビューに Essbase サーバーを追加できます:

- エンタープライズ・ビューの「Essbase サーバー」ノードを右クリックし、「Essbase サーバーの追加」を選択します。
- 「ファイル」>「ウィザード」>「ユーザーの設定」を選択します。ウィザードの指示に従います。
- 既存のサーバーの場合は、サーバーのプロパティを編集します。

また、ナビゲーション・ペイン内の別のタブに、エンタープライズ・ビュー・ツリーのカスタム・ビューも作成できます。Oracle Essbase Administration Services Online Help の「カスタム・ビューについて」を参照してください。

➤ エンタープライズ・ビューに Essbase サーバーを追加するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「エンタープライズ・ビューへの Essbase サーバーの追加」を参照してください。

Essbase サーバーの接続およびポートについて

Essbase サーバーで使用可能なポートの数は、ライセンスされている同時接続の数を表します。Essbase には、システム管理者のために予約ポートが 1 つ用意されています。システム管理者は、他のポートがすべて使用されているとき、この予約ポートを使用して 1 人以上のユーザーをログアウトします。第 46 章「Essbase サーバー、アプリケーションおよびデータベースの実行」を参照してください。

Administration Services では、ポートは Essbase サーバーの接続が確立されている場合にのみ使用されます。『Oracle Essbase Administration Services オンライン・ヘルプ』の「Oracle Essbase Administration Services Online Help の接続およびポートについて」を参照してください。

Essbase 管理サーバーについて

中間層の Essbase 管理サーバーは、サーバー間にわたる操作、ユーザー・プリファレンスの持続性および Essbase サーバーへのアクセスをサポートするためのビジネス・ロジックを提供します。システム管理者が Essbase 管理サーバー上のユーザーを作成した後、Essbase 管理サーバーがそれらのユーザーの Essbase への接続を管理します。

エンタープライズ・ビューに表示される Essbase 管理サーバーのノード名は、サーバー・コンピュータの名前と同じです。

Essbase 管理サーバーには、Essbase サーバー・ポートとは別の構成可能な通信ポートがいくつかあります。デフォルトの通信ポートのいずれかが別のアプリケーションによって使用されている場合は、Essbase 管理サーバーを実行するために別のポート値を指定する必要があります。

注： Essbase 管理サーバーのポートの値を変更した場合は、管理サービス・コンソールにログオンするときに、この新しいポート値を指定する必要があります。

➤ デフォルトのポート値を変更するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の Administration Server の通信ポートの指定に関する項を参照してください。

第 I I 部

アプリケーションとデータベースの設計および作成

アプリケーションとデータベースの設計および作成の内容：

6 アプリケーションとデータベースの作成

この章の内容

この章の情報の一部は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 第 61 章「集約ストレージ・アプリケーション、データベースおよびアウトライン」

アプリケーションとデータベースの作成プロセス

➤ アプリケーションおよびデータベースを作成するには:

1 **アプリケーションを設計します。**

51 ページの「Essbase の実装の開始」を参照してください。

2 **アプリケーションを作成します。**

117 ページの「アプリケーションの作成」を参照してください。

3 **データベースを作成します。**

118 ページの「データベースの作成」を参照してください。

4 **必要に応じて、Essbase サーバー、アプリケーションまたはデータベース・レベルで代替変数を設定します。**

118 ページの「代替変数の使用」を参照してください。

5 **必要に応じて、データベースのロケーション別名を設定します。**

122 ページの「ロケーション別名の使用」を参照してください。

6 アウトラインを作成します。

第 7 章「データベース・アウトラインの作成および変更」を参照してください。

アプリケーションおよびデータベースの理解

Essbase アプリケーションは、1 つ以上の Essbase データベースおよび関連するファイルを含む管理構造です。Essbase アプリケーションおよびデータベースは、Essbase サーバー上に存在します。サーバー・コンピュータは、複数のアプリケーションを保管できます。

Essbase データベースは、多次元データ・ストレージ・アレイを含むデータ・リポジトリです。多次元データベースは、ユーザーがデータを分析して、意味のあるビジネス上の意思決定を行えるように、データの複数のビューをサポートします。57 ページの「多次元データベースの理解」および847 ページの「ストレージの割当て」を参照してください。

データベース・アーティファクトの理解

データベースに関連したファイルは、アーティファクト(またはオブジェクト)と呼ばれます。データベース・アーティファクトは、計算の定義やデータに対するレポート作成などの、1 つ以上の Essbase データベースに対するアクションを実行します。デフォルトでは、アーティファクトは Essbase サーバー上の関連付けられたデータベース・フォルダに保管されており、クライアント・コンピュータまたはその他の使用可能なネットワーク・ディレクトリにも保存できます。ただし、クライアント・コンピュータ上でデータをロードしたり、データを計算したりはできません。

Essbase には、次の一般的なタイプが用意されています:

- データベース・アウトライン(ストレージ構造の定義)
- データ・ソース
- データのロードや動的な次元構築に関する規則(ルール・ファイル)
- データの計算方法を定義するスクリプト(計算スクリプト)
- データに関するレポートを生成するスクリプト(レポート・スクリプト)
- セキュリティ定義
- リンク・レポート・オブジェクト(LRO)
- パーティション定義

これらのアーティファクトの一部(計算スクリプトや LRO など)はオプションです。778 ページの「アプリケーションおよびデータベースのファイル・タイプ」を参照してください。

管理サービス・コンソールでは、データベース・アーティファクトは、エンタープライズ・ビュー・ツリーの関連するアプリケーションまたはデータベースの下に表示されます。

データベース・アウトラインの理解

データベース・アウトラインは、すべての次元、メンバー、別名、プロパティ、タイプ、連結および数学的な関係を含む、多次元データベースの構造を定義します。アウトラインで定義されているこの構造によって、データベースでのデータの保管方法が決定されます。

データベースが作成されるとき、そのデータベースのアウトラインが Essbase によって自動的に作成されます。このアウトラインは、データベースと同じ名前(dbname.otl)を持っています。たとえば、サンプル・アプリケーション内で Basic データベースが作成されるときは、次のディレクトリにアウトラインが作成されます:

```
ARBORPATH
/app/sample/basic/basic.otl
```

118 ページの「データベースの作成」および第 7 章「データベース・アウトラインの作成および変更」を参照してください。

データ・ソースの理解

データ・ソースとは、Essbase データベースにロードされる外部データのことです。データ・ソースの一般的なタイプには、次のものがあります:

- テキスト・ファイル
- スプレッドシート・ファイル
- スプレッドシートの監査ログ・ファイル
- SQL データベースのような外部データベース

274 ページの「サポートされるデータ・ソース」を参照してください。

データ・ロードと次元構築のルール・ファイルの理解

Essbase データベースが作成されたとき、このデータベースにデータは含まれていません。データ・ロードのルール・ファイルは、Essbase データベースに外部データ・ソース・ファイルのデータがロードまたはコピーされたときに、そのデータに対して Essbase で実行される一連の操作です。次元構築のルール・ファイルは、外部データ・ソース内のデータに基づいて、アウトライン内の次元およびメンバーを動的に作成または変更します。ルール・ファイルは通常、特定のデータベースに関連付けられていますが、複数のデータベースで使用するためのルールを定義できます。1 つのルール・ファイルを、データ・ロードと次元構築の両方に使用できます。ルール・ファイルの拡張子は.rul です。

280 ページの「ルール・ファイル」および第 18 章「ルール・ファイルの使用」を参照してください。

計算スクリプトの理解

計算スクリプトは、Essbase にデータベース内のデータの計算方法を指示する一連の手順を含むテキスト・ファイルです。計算スクリプトは、データベース・アウトラインで定義されている連結や数学的な操作とは別の計算を実行します。計算スクリプトは、メンバーに対する特定の数学的な操作を実行するため、通常は特定のデータベースに関連付けられています。ただし、複数のデータベースで使用するための計算スクリプトを定義できます。計算スクリプト・ファイルの拡張子は.csc です。

第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。

レポート・スクリプトの理解

レポート・スクリプトは、データベースからレポートを作成するためのデータ取得、フォーマットおよび出力手順を含むテキスト・ファイルです。レポート・スクリプトは通常、特定のデータベースに関連付けられていますが、複数のデータベースで使用するためのレポート・スクリプトを定義できます。レポート・スクリプトの拡張子は.rep です。

第 34 章「レポート・スクリプトの作成」を参照してください。

セキュリティ定義の理解

Essbase は、アプリケーション、データベース、その他のアーティファクトへのアクセスを管理するための包括的なシステムを提供します。各アプリケーションおよびデータベースには、ユーザー・アクセスを制限するための独自のセキュリティ定義が含まれています。

第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」を参照してください。

LRO の理解

LRO とは、Essbase データベース内の特定のデータ・セルに関連付けられたアーティファクトのことです。LRO は、セル上の追加の情報を提供することによってデータ分析機能を拡張できます。

LRO は、次のいずれかです:

- 説明テキスト(セル・ノート)
- テキスト、音声、動画または静止画像を含む個別のファイル
- Web サイトの URL
- 別の Essbase データベース内のデータへのリンク

第 11 章「Essbase データへのオブジェクトのリンク」を参照してください。

メンバー選択の定義の理解

Smart View 内で、ユーザーはメンバー選択機能を使用して、メンバーの取得を定義したり保存したりできます。メンバー指定ファイルの拡張子は.sel です。

Oracle Hyperion Smart View for Office User's Guide を参照してください。

トリガー定義の理解

トリガー機能を使用すると、データベース内のデータ変更を効率的に監視できます。データがトリガーで指定されているルールに違反した場合、Essbase は関連情報をファイルに記録するか、またはトリガーによっては電子メール・アラートを送信します。たとえば、西部地域で、ある月の販売が前の年の同じ月の販売を下回った場合は販売マネージャに通知します。

795 ページの「データ変更の監視(トリガーを使用)」を参照してください。

アプリケーションとデータベースの作成

アプリケーションを作成した後、そのアプリケーションのデータベースを作成します。データベースに注釈を付けられます。

アプリケーションの作成

Essbase サーバー上でアプリケーションを作成すると、Essbase によって、そのアプリケーションのサブディレクトリがその Essbase サーバーの ARBORPATH/app ディレクトリに作成されます。新しいサブディレクトリには、そのアプリケーションと同じ名前が付けられます。たとえば、ARBORPATH/app/ p1 となります。管理サービス・コンソールでは、アプリケーションおよびデータベースがエンタープライズ・ビュー内のツリー構造に表示されます。

アプリケーションに名前を付ける前に、1201 ページの「アプリケーションとデータベースの命名規則」を参照してください。

また、既存のアプリケーションのコピーであるアプリケーションも作成できます。784 ページの「アプリケーションのコピーまたは移行」を参照してください。

➤ アプリケーションを作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アプリケーションの作成	Oracle Essbase Administration Services Online Help
MaxL	**create application**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	CREATEAPP	『Oracle Essbase テクニカル・リファレンス』

データベースの作成

データベースを作成すると、Essbase によって、そのデータベースのサブディレクトリがアプリケーション・ディレクトリ内に作成されます。新しいサブディレクトリには、そのデータベースと同じ名前が付けられます。たとえば、ARBORPATH/app/app1/dbname1 となります。管理サービス・コンソールでは、アプリケーションおよびデータベースがエンタープライズ・ビュー内のツリー構造に表示されます。

通常のデータベースまたは通貨データベースを作成できます。第 14 章「通貨換算アプリケーションの設計および作成」を参照してください。

データベースに名前を付ける前に、1201 ページの「アプリケーションとデータベースの命名規則」を参照してください。

➤ データベースを作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データベースの作成	Oracle Essbase Administration Services Online Help
MaxL	**create database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	CREATEDB	『Oracle Essbase テクニカル・リファレンス』

「通貨換算」オプションの使用が必要なデータベースを除き、アプリケーション当たり 1 つのデータベースを作成することをお薦めします。

データベースへのコメント

データベースのステータスや、更新の最終期限などについてユーザーにメッセージをブロードキャストする必要がある場合、データベース・ノートは有効な情報を提供できます。ユーザーは、Smart View でデータベース・ノートを表示できます。

➤ データベースにコメントを付けるには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「データベースへのコメント」を参照してください。

代替変数の使用

代替変数は、定期的に情報を変更するためのグローバル・プレースホルダです。変数値を変更すると、その変数が使用されている他の場所にも反映されるため、手動による変更が削減されます。

たとえば、多くのレポートがレポート期間に依存しています。現在の月に基づいてレポートを生成した場合は、毎月レポート・スクリプトを手動で更新する必要があります。サーバー上で設定される CurMnth などの代替変数を使用すると、割り当てられた値を毎月該当する時間間隔に変更できます。 レポート・スクリプトで変数名を使用した場合は、最終レポートを実行すると、情報が動的に更新されます。

次の領域では、(記載のない場合)集約ストレージ・アプリケーションとブロック・ストレージ・アプリケーションの両方で代替変数を使用できます:

- 集約ストレージ・アウトライン式

  1028 ページの「MDX 式の使用」を参照してください。

- ブロック・ストレージ・アウトライン式

  398 ページの「式での代替変数の使用」を参照してください。

- 計算スクリプト(ブロック・ストレージ・データベースのみ)

  計算スクリプトでは、代替変数およびランタイム代替変数がサポートされています。第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。

- レポート・スクリプト

  第 34 章「レポート・スクリプトの作成」を参照してください。

- データ・ロードのルール・ファイルのヘッダー定義とフィールド定義。次元名およびメンバー名に対して変数名を入力できます。

  Oracle Essbase Administration Services Online Help のルール・ファイルのヘッダーの設定およびフィールド名のマッピングのトピックを参照してください。

- パーティション定義

  225 ページの「パーティション定義での代替変数」を参照してください。

- SQL データ・ソースのルール・ファイルにおけるデータ・ソース名(DSN)の指定

  『Oracle Essbase SQL インタフェース・ガイド』を参照してください。

- SQL データ・ソースのルール・ファイルにおける SELECT、FROM または WHERE 句

  『Oracle Essbase SQL インタフェース・ガイド』を参照してください。

- セキュリティ・フィルタ

  703 ページの「フィルタでの代替変数の使用」を参照してください。

- MDX ステートメント

  617 ページの「MDX クエリーでの代替変数の使用」を参照してください。

- Smart View。

  Oracle Hyperion Smart View for Office User's Guide を参照してください。

Administration Services、MaxL または ESSCMD を使用して、Essbase サーバー上で代替変数を設定できます。次のいずれかのレベルで変数を設定します:

- Essbase サーバー- Essbase サーバー上のすべてのアプリケーションおよびデータベースから変数にアクセスできます
- アプリケーション- アプリケーション内のすべてのデータベースから変数にアクセスできます
- データベース- 指定したデータベース内の変数にアクセスできます

代替変数の名前と値の設定ルール

代替変数の名前と値には、次のルールが適用されます:

- 代替変数名は英数字またはアンダースコア(_)で構成する必要があり、付録 A「制限」で指定された制限を超えることはできません。
- 代替変数名には、ハイフン(-)、アスタリスク(*)、スラッシュ(/)などの、英数字以外の文字を含めることはできません。MDX で使用される代替変数名には、スペース、句読点または大カッコ([])を使用しないでください。
- サーバー、アプリケーションおよびデータベース・レベルに同じ名前を持つ代替変数が存在する場合、それらの変数の優先度は、データベースレベルの代替変数、アプリケーションレベルの変数、サーバーレベルの変数の順になります。
- 代替変数値には、先頭のアンパサンド(&)を除く任意の文字を含めることができます。代替変数値が、付録 A「制限」で指定された制限を超えることはできません。
- 重複するメンバー名に代替変数値を設定するには、二重引用符で囲まれた修飾メンバー名を使用します。たとえば、&Period の値を"[2006].[Qtr1]"にできます。
- 代替変数の使用を指定する場合は、修飾名の一部として代替変数を挿入しないでください。たとえば、"[2004].[&CurrentQtr]"を指定することは無効です。
- 代替変数値が、数字で始まるか、またはスペースや1206 ページの「計算スクリプト、レポート・スクリプト、式、フィルタ、代替変数値および環境変数値での命名規則」に示されている任意の特殊文字を含むメンバー名である場合は、変数の入力方法に別のルールが適用されます:
  - MDX ステートメントで使用する場合は、メンバー名の値を大カッコ([ ])で囲みます。
  - MDX ステートメントで使用しない場合は、メンバー名の値を引用符(" ")で囲みます。
- 代替変数値が数値である場合は、変数の入力方法に別のルールが適用されます:
  - 代替変数値を MDX ステートメントで使用しない場合は、値を引用符で囲みます。たとえば、変数名が Month であり、それに対応する値が 01 (1 月に対応)の場合は、01 の前後に引用符を置きます("01")。代替変数は通常、ブロック・ストレージ・データベースで使用され、MDX ステートメントでは使用されません。
  - 集約ストレージ・アウトライン内の式のように、代替変数値を MDX ステートメントでのみ使用し、かつその値が数値またはメンバー名である場合は、値を引用符で囲まないでください。

  注： 代替変数値が数値であるか、数値で始まるメンバー名や前述の特殊文字を含むメンバー名が MDX と非 MDX の両方で使用される場合は、値を引用符で囲んでいない代替変数と値を引用符で囲んだ代替変数の 2 つを作成してください。

代替変数の設定

Essbase サーバー上で、サーバー、アプリケーションまたはデータベース・レベルで代替変数を設定できます。代替変数を設定する前に、120 ページの「代替変数の名前と値の設定ルール」を参照してください。

➤ 代替変数を設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	代替変数の管理	Oracle Essbase Administration Services Online Help
MaxL	**alter system** **alter application** **alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	CREATEVARIABLE	『Oracle Essbase テクニカル・リファレンス』

新しい代替変数値が式、パーティション定義およびセキュリティ・フィルタで確実に使用可能になるように、アプリケーションを停止して再起動してください。代替変数の他のすべての使用は、使用時に動的に解決されます。

代替変数の削除

不要になった代替変数を削除することが必要な場合もあります。

➤ 代替変数を削除するには、次のツールを使用します:

ツール	トピック	参照
Administration Services	代替変数の管理	Oracle Essbase Administration Services Online Help
MaxL	**alter system** **alter application** **alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	DELETEVARIABLE	『Oracle Essbase テクニカル・リファレンス』

代替変数の更新

既存の代替変数を変更または更新できます。代替変数を更新する前に、120 ページの「代替変数の名前と値の設定ルール」を参照してください。

➤ 代替変数を更新するには、次のツールを使用します:

ツール	トピック	参照
Administration Services	代替変数の管理	Oracle Essbase Administration Services Online Help

ツール	トピック	参照
MaxL	**alter system** **alter application** **alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	UPDATEVARIABLE	『Oracle Essbase テクニカル・リファレンス』

代替変数のコピー

代替変数は、適切なアクセス権がある任意の Essbase サーバー、アプリケーションまたはデータベースにコピーできます。

➤ 代替変数をコピーするには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「代替変数のコピー」を参照してください。

ロケーション別名の使用

ロケーション別名は、データ・ソースの記述子です。ロケーション別名は、データ・ソースの別名の名前を、そのデータベースの場所にマッピングします。ロケーション別名はデータベース・レベルで設定され、別名、サーバー、アプリケーション、データベース、ユーザー名およびパスワードを指定します。ロケーション別名は、計算スクリプトが実行されているデータベースで設定します。

ロケーション別名を作成したら、別名を使用してそのデータベースを参照できます。データベースの場所が変更された場合は、それに応じて場所の定義を編集します。

ロケーション別名は@XREF および@XWRITE 関数での使用できます。@XREF の場合、別のデータベースからデータ値を取得して現在のデータベースでの計算に組み込むことができます。この場合、ロケーション別名は、値の取得元のデータベースを指し示します。@XWRITE の場合、値を別の Essbase データベースまたは同じデータベースに書き込むことができます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

保護モード(SSL)でロケーション別名を使用する場合は、次の考慮事項が適用されます:

- Essbase が SSL 接続を使用できるようにするには、ENABLESECUREMODE を TRUE に設定する必要があります。
- 保護ポートにロケーション別名が設定されている場合は、サーバー名の指定に:secure を追加する必要があります。たとえば、MaxL を使用した場合、

```
create location alias EasternDB from Sample.Basic to East.Sales at Easthost:
6423:secure as User1 identified by password1;
```

ロケーション別名の作成

個々のデータベースに対してロケーション別名を作成できます。

➤ ロケーション別名を作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ロケーション別名の作成	Oracle Essbase Administration Services Online Help
MaxL	**create location alias**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	CREATELOCATION	『Oracle Essbase テクニカル・リファレンス』

ロケーション別名の編集または削除

作成済のロケーション別名を編集または削除できます。

➤ ロケーション別名を編集または削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ロケーション別名の編集または削除	Oracle Essbase Administration Services Online Help
MaxL	**display location alias** **drop location alias**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	LISTLOCATIONS DELETELOCATION	『Oracle Essbase テクニカル・リファレンス』

7 データベース・アウトラインの作成および変更

この章の内容

この章の情報の一部は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 第 61 章「集約ストレージ・アプリケーション、データベースおよびアウトライン」

この章のすべての例は、Sample.Basic データベースに基づいています。

アウトラインの作成プロセス

この項では、アウトライン・エディタを使用したアウトライン作成の概要について説明します。アウトライン・エディタについては、Oracle Essbase Administration Services Online Help を参照してください。

アウトラインに関する基本情報については、第 3 章「多次元データベースの理解」を参照してください。

➤ アウトラインを作成する手順は次のとおりです:

1 **データベースを作成します。新しいデータベースには、空白のアウトラインが自動的に含まれます。**

117 ページの「アプリケーションとデータベースの作成」を参照してください。

2 **アウトラインを開きます。**

126 ページの「アウトラインの作成および編集」を参照してください。

3 アウトラインに次元およびメンバーを追加します。

128 ページの「アウトラインへの次元およびメンバーの追加」を参照してください。

4 各次元を密または疎として設定します。

128 ページの「データ・ストレージ・プロパティの設定」を参照してください。

5 次元およびメンバーをアウトラインに位置付けます。

128 ページの「次元およびメンバーの位置付け」を参照してください。

6 次元とメンバーのプロパティを設定します。

第 9 章「次元およびメンバーのプロパティの設定」を参照してください。

7 必要に応じて、属性次元を作成して、適切な基本次元に関連付けます。

第 10 章「属性の操作」を参照してください。

8 アウトラインを確認してから保存します。

図 37 および131 ページの「アウトラインの保存」を参照してください。

アウトラインの作成および編集

データベース・アウトラインによって、データベースの構造が定義されます。アウトライン・エディタには、アウトラインの次元階層が視覚的に表示されます。

データベースが作成されるとき、そのデータベースのアウトラインが Essbase によって自動的に作成されます。このアウトラインは、データベースと同じ名前(`dbname.otl`)を持っており、Essbase サーバー上のデータベース・ディレクトリに保管されます。新しいアウトライン内には、次の方法でコンテンツを作成できます:

- アウトライン・エディタで、データベースの作成時にデフォルトで作成される空のアウトラインを開き、手動でコンテンツを追加します。

  Oracle Essbase Administration Services Online Help の「アウトラインを開いて編集する」を参照してください。
- 既存のアウトラインを現在のデータベースにコピーし、既存のアウトラインを変更します。
- データ・ソースとルール・ファイルを使用してコンテンツを作成します。

  第 17 章「データ・ロードおよび次元構築の理解」を参照してください。

注意 同じログイン ID を使用して、管理サービス・コンソールの 2 つのインスタンスで同じアウトラインを開いた場合は、保存するたびに、もう一方のインスタンスの変更が上書きされます。保存されたり、上書きされたりした変更を追跡することは困難な場合があるため、この方法はお薦めできません。

➤ アウトラインを作成するか、既存のアウトラインを開くには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アウトラインを開いて編集する	Oracle Essbase Administration Services Online Help
MaxL	**create database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	CREATEDB	『Oracle Essbase テクニカル・リファレンス』

➤ 既存のアウトラインをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アウトラインのコピー	Oracle Essbase Administration Services Online Help
MaxL	**create database as**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYDB	『Oracle Essbase テクニカル・リファレンス』

アウトラインのロックおよびロック解除

アウトライン・エディタでは、アウトラインが編集モードで開かれたとき、そのアウトラインは常にロックされます。アウトラインが閉じられたとき、Essbase によってそのアウトラインのロックが解除されます。アウトラインがロックされていると、他のユーザーによるそのアウトラインの上書き保存、名前変更、削除または編集は Essbase によって許可されません。ロックされたアウトラインを編集しようとすると、アウトライン・ビューアでアウトラインを表示するかどうかを選択するよう求められます。

管理者権限を持っている場合は、ロックされたアウトラインのロックを解除できます。ロックされたアウトラインのロックを強制的に解除する前に、そのアウトラインで作業しているユーザーがいないことを確認してください。

注： Essbase では、アウトラインのロックおよびロック解除に、他のデータベース・アーティファクトとは異なるプロセスを使用しています。788 ページの「アーティファクトのロックおよびロック解除」を参照してください。

➤ アウトラインのロックを解除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アウトラインのロックおよびロック解除	Oracle Essbase Administration Services Online Help
MaxL	**create database as**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	UNLOCKOBJECT	『Oracle Essbase テクニカル・リファレンス』

アウトラインへの次元およびメンバーの追加

アウトラインに次元およびメンバー階層を追加するには、次のいくつかの方法があります:

- アウトライン・エディタを使用して手動で
- データ準備エディタを使用して、データ・ソースとルール・ファイルによって

次元とメンバーに名前を付ける前に、1201 ページの「アプリケーションとデータベースの命名規則」を参照してください。

➤ アウトライン・エディタを使用してアウトラインに次元とメンバーを追加するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アウトラインへの次元の追加」および「次元へのメンバーの追加」を参照してください。

➤ データ準備エディタを使用してアウトラインに次元とメンバーを追加するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元構築ルール・ファイルの作成」を参照してください。

➤ アウトライン・エディタから(ルール・ファイルを使用して)次元とメンバーを動的に追加するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「ルール・ファイルを使用したアウトラインの動的な更新」を参照してください。

データ・ストレージ・プロパティの設定

次元を作成してアウトラインを保存すると、Essbase によって、そのアウトライン内の新しい次元が自動的に疎として設定されます。データベースの最適な構成に応じて、次元ストレージ・タイプを変更できます。

属性次元を関連付けることを予定している標準次元はすべて疎として設定します。63 ページの「密次元と疎次元の選択」を参照してください。

➤ アウトライン・エディタを使用してデータ・ストレージ・プロパティを設定するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元の疎/密の設定」を参照してください。

次元およびメンバーの位置付け

次元は、アウトライン内の最も高いレベルの組織です。次元にはメンバーが含まれており、このメンバーは階層内の他のメンバーの内部にネストできます。58 ページの「次元とメンバー」を参照してください。

以降の項では、次元とメンバーをアウトライン内に位置付ける方法について説明します。

注： アウトライン内の次元の相対的な場所が、計算や取得のパフォーマンス時間に影響を与える場合があります。91 ページの「パフォーマンスを最適化するためのアウトラインの設計」を参照してください。

次元およびメンバーの移動

次元およびメンバーを作成した後、それらをアウトライン内で再調整できます。アウトライン内のメンバーと次元を移動する前に、次の点を考慮してください:

- アウトラインの次元とメンバーの位置によって、パフォーマンスは影響を受けます。

  172 ページの「アウトラインのパフォーマンスの最適化」を参照してください。

- 次元とメンバーを移動すると、計算と取得のパフォーマンスに影響が出る場合があります。

  91 ページの「パフォーマンスを最適化するためのアウトラインの設計」を参照してください。

- メンバーを移動すると、共有メンバーがアウトラインの実際のメンバーより前に移動することがあります(これはお薦めしません)。
- 属性以外の次元またはメンバーを追加、削除または移動すると、Essbase でデータベースの再構築が行われるため、データを再計算する必要があります。
- 属性次元をアウトラインの最後に置きます。そうしないと、アウトラインの確認中に、属性次元をそこに移動するよう求めるプロンプトが Essbase に表示されます。

➤ アウトライン・エディタを使用して次元およびメンバーを位置付けるには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アウトライン内の次元およびメンバーの操作」を参照してください。

次元およびメンバーのソート

Essbase では、アウトライン内の次元または次元内のメンバーを、アルファベット順の昇順(A から Z)またはアルファベットの逆順(Z から A)に整列できます。次元およびメンバーのソート結果のリストについては、129 ページの「次元およびメンバーの移動」を参照してください。

アウトライン内の数値属性次元のレベル 0 のメンバーをソートすると、メンバーはその値によってソートされます。例として、図 37 に、メンバーを昇順にソートした後のサイズ属性次元のテキストと数値のバージョンを示しています。数値属性次元(右側)のメンバーは、メンバーの数値によって順序付けられるため、メンバー 8 は他のメンバーの前になります。テキスト属性次元(左側)では、文字が左から右にソートされるため、メンバー 8 は他のメンバー 24 の後になります。

図 37　数値とテキストの属性次元の昇順によるソートの比較

Sizes Attribute (Type: Text)	Sizes Attribute (Type: Numeric)
Ounces	Ounces
12	8
16	12
24	16
8	24

ブール属性次元はソートできません。166 ページの「属性タイプの理解」を参照してください。

➤ アウトライン・エディタを使用してメンバーをソートする方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「メンバーのソート」を参照してください。

アウトラインの確認

アウトラインの確認は、アウトラインを保存するときに自動的に行うか、任意に手動で行うことができます。アウトラインを確認すると、Essbase で次の点が確認されます:

- すべてのメンバーと別名の名前が有効であること。メンバーと別名は、他のメンバー、別名、世代またはレベルと同じ名前を持つことはできません。

  1201 ページの「アプリケーションとデータベースの命名規則」を参照してください。
- 1 つの次元のみに、会計、時間、通貨タイプまたは国タグが付けられていること。
- 共有メンバーが有効であること。

  148 ページの「共有メンバーの規則の理解」を参照してください。
- レベル 0 のメンバーに、「ラベルのみ」タグが付けられていないこと。
- ラベルのみのメンバーに、式が割当てられていないこと。
- ラベルのみメンバーの子孫に動的計算タグが付いていないこと。

  147 ページの「ラベルのみメンバーの理解」を参照してください。
- 通貨カテゴリと通貨名が、通貨アウトラインに対して有効であること。
- 疎次元の動的計算メンバーの子が 100 以下であること。
- 親メンバーに子が 1 つあり、その子が動的計算メンバーである場合、親メンバーも動的計算であること。
- 親メンバーに子が 1 つあり、その子が動的計算の 2 パス・メンバーである場合、親メンバーも動的計算の 2 パスであること。
- ブール属性次元の 2 つの名前が、アウトラインに対して定義されている 2 つのブール属性次元メンバー名と同じであること。
- 日付属性次元のレベル 0 のメンバーの名前が、日付フォーマット名の設定(mm-dd-yyyy または dd-mm-yyyy)に一致すること。この次元にメンバーがない場合、次元名はレベル 0 のメンバーであるため、次元名はこの設定に一致する必要があります。

- 数値属性次元のレベル 0 のメンバーの名前が数値であること。この次元にメンバーがない場合、次元名はレベル 0 のメンバーであるため、次元名は数値である必要があります。
- 属性次元がアウトラインの終端に位置付けられ、すべての標準次元に従っていること。
- 標準次元のレベル 0 の動的計算メンバーが式を持つこと。
- メンバーの式が有効であること。
- ハイブリッド分析アウトラインで、ハイブリッド分析に対応できるのが次元のレベル 0 のメンバーのみであること。

アウトラインの確認中は、Essbase で次のような変換も同時に行われ、適切な数値属性次元のメンバー名がアウトラインに表示されます:

- メンバー名の先頭にあるマイナス符号を、末尾に移動します(-1 が 1-になるなど)。
- メンバー名内の先行する値 0 または末尾に続く値 0 が、取り除かれます(1.0 が 1 に、00.1 が 0.1 になるなど)。

166 ページの「属性タイプの理解」を参照してください。

➤ アウトラインを確認するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アウトラインの確認」を参照してください。

アウトラインの保存

アウトラインは、Essbase サーバーまたはクライアント、あるいはネットワークに保管できます。デフォルトでは、アウトラインは Essbase によって Essbase サーバー上のデータベース・ディレクトリに保存されます。アウトラインに対する変更を保存する場合は、Essbase でアウトラインが再構築されることがあります。たとえば、メンバー名を市場から地域に変更すると、Essbase によって市場に関連して保管されているデータが地域に移動されます。アウトラインを保存するたびに、それが正しいことを保証するために Essbase によってアウトラインが確認されます。

➤ アウトラインを保存する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アウトラインの保管」を参照してください。

また、次の項も参照してください。

標準次元を追加したアウトラインの保存

1 つ以上の新しい標準(属性以外)次元を追加する場合は、アウトラインを保存する前に、データベース内の既存のデータすべてを新しい各次元のメンバーにマッピングする必要があります。たとえば、Sample.Basic アウトラインにチャネルという次元を追加することは、Sample.Basic 内の以前のすべてのデータを特定のチャネルまたはすべてのチャネルの合計に関連付けることを意味します。

1 つ以上の標準次元を削除したアウトラインの保存

1 つ以上の標準(属性以外)次元を削除する場合は、削除する各次元の 1 つのメンバーのみに関連付けられているデータを保持し、他の次元の 1 つのメンバーに関連付ける必要があります。たとえば、アウトラインから市場という次元を削除することは、再構築操作の後データベース内に残るすべてのデータが、市場次元の単一の指定されたメンバーに関連付けられることを意味します。

属性次元を削除すると、Essbase によってその基本次元に対する関連付けが削除されます。第 10 章「属性の操作」を参照してください。

削除メンバーを使用したサブ・データベースの作成

➤ サブ・データベースを作成するには:

1 **既存のアウトラインから次元を削除します。**

2 **別名を使用してデータベースを保存し、保持するメンバーを指定します。**

次元を削除したときに保持できるのは、1 つのメンバーのみです。132 ページの「1 つ以上の標準次元を削除したアウトラインの保存」を参照してください。

8 重複メンバーのアウトラインの作成および使用

この章の内容

この章の情報は、ブロック・ストレージと集約ストレージ・データベースに適用されます。

関連項目:

- 第 6 章「アプリケーションとデータベースの作成」
- 第 7 章「データベース・アウトラインの作成および変更」

アウトラインでの重複するメンバー名の作成

アウトライン内で重複する名前を使用できるのは、アウトラインでメンバーの重複を許可するオプションが有効になっている場合のみです。

アウトラインでメンバー名の重複が使用可能になっている場合、Essbase ではアウトラインに同じ名前を使用する複数のメンバーが表示されます。名前は通常の方法で作成します。1202 ページの「次元、メンバーおよび別名の命名規則」を参照してください。

アウトラインを保管すると、Essbase によって重複するメンバー名を持つアウトラインが検証され、保存されます。重複するメンバー名は、修飾名のフォーマットによって区別されます。

図 38 に、重複するメンバー名のあるアウトラインの例を示します。State メンバー New York と City メンバー New York は、アウトラインには New York と表示されます。

図 38　重複するメンバー名「New York」

図 38 の例の修飾メンバー名は、[State].[New York]および[City].[New York]です。135 ページの「重複するメンバー名および別名を指定する構文」を参照してください。

➤ 重複するメンバー名が使用可能なアウトラインを作成したり、一意のメンバー名のアウトラインを重複するメンバー名のアウトラインに変換するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	重複メンバー名のアウトラインの作成	Oracle Essbase Administration Services Online Help
MaxL	create database	『Oracle Essbase テクニカル・リファレンス』

注： アウトラインの変更を重複メンバー名のアウトラインに変換する前に、その変更を保存します。未保存の変更が含まれているアウトラインは変換できません。アウトラインを重複メンバー・アウトラインに変換した後、その他のアウトラインの変更を続ける前にそのアウトラインを保存します。重複メンバー・アウトラインを変換して一意のメンバー・アウトラインに戻すことはできません。

重複メンバー・アウトライン内では、特定の次元、世代およびレベルを一意または重複としてタグ付けして、データベース内での重複メンバー名の使用を制限できます。これにより、重複メンバー・アウトライン内のメンバー名の一意性をより細かいレベルで指定できます。

重複するメンバーのアウトラインを作成すると、デフォルトではアウトラインのすべての次元に重複のタグが付けられます。

次元内で重複メンバーを使用可能な場合は、次元内の特定の世代またはレベルを一意としてタグ付けできます。メンバーが重複するプロパティに割当てられている場合は、一意のプロパティが優先します。

➤ 次元内の重複メンバー名を使用可能または使用不可にするには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元に対する一意のタグ付け」を参照してください。

➤ 次元の特定の世代またはレベルでの重複メンバー名を使用不可にするには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「世代およびレベルの名前付け」を参照してください。

注： 重複メンバー・アウトラインの属性次元には、接頭辞または接尾辞は付けられません。これらは一意のアウトライン内の属性次元に適用されます。たとえば、重複メンバーのブール属性次元では、メンバーには、TRUE および FALSE メンバーに付けられた次元、親、親の親、または祖先は組み込まれません。173 ページの「属性次元のメンバー名における、接頭辞および接尾辞のフォーマットの設定」を参照してください。

アウトラインでの重複するメンバー名および別名の制限

データベース・アウトライン内に重複するメンバー名と別名を作成するとき、次のものを必ず一意にする必要があります:

- 次元名
- 世代名/レベル名
- 親メンバーの下の兄弟

別名を使用する場合は、重複する別名を含む別名テーブルは、重複メンバー・アウトラインでのみ有効であるという、追加の制限が適用されます。

注： メンバー、次元または別名には、引用符(" ")、大カッコ([ ])またはタブを使用しないでください。たとえば、[New York].[Area 1]というメンバー名は作成できません。アウトラインの確認では、一連の無効な文字を含むメンバー名に対するエラーは表示されず、アウトラインを保存できます。ただし、Essbase でデータを正確にクエリーできなくなります。

重複するメンバー名および別名を指定する構文

アウトライン内に重複するメンバー名が存在している場合でも、データベース内のメンバーは、それぞれの非共有のメンバー名によって一意に識別されます。重複するメンバー名は、修飾名のフォーマットによって区別されます。Administration Services エディタを使用している場合は、アウトライン・ツリーの修飾メンバー名を選択できます。重複するメンバーの修飾メンバー名は、Administration Services のアウトライン・ビューアの「メンバーのプロパティ」ダイアログ・ボックスで確認できます。重複するメンバー名を指定するには、修飾名を使用する必要があります。

修飾メンバーや別名は、次のいずれかのフォーマットで指定できます:

- 完全修飾メンバー名
- 祖先を区別することによって修飾されるメンバー名
- ショートカットの修飾メンバー名

注： 修飾名は、すべて別名で構成するか、すべてメンバー名で構成する必要があります。修飾名でメンバー名と別名を混在できません。

完全修飾メンバー名の使用

完全修飾メンバー名は、重複するメンバー名または別名と、次元名までを含むすべての祖先で構成されます。それぞれの名前は、大カッコ([ ])で囲み、ピリオド(.)で区切る必要があります。構文は次のとおりです:

```
[
DimensionMember
].[
Ancestors...
].[
DuplicateMember
]
```

例:

```
[Market].[East].[State].[New York]
[Market].[East].[City].[New York]
```

祖先の区別によるメンバーの修飾

祖先を区別することによって修飾されるメンバー名では、メンバー名または別名と、重複するメンバーまたは別名を一意に識別する祖先までを含むすべての祖先が使用されます。パス内の最初の祖先は、常に一意のメンバー名になります。それぞれの名前は、大カッコ([ ])で囲み、ピリオド(.)で区切る必要があります。構文は次のとおりです:

```
[
DifferentiatingAncestor
].[
Ancestors...
].[
DuplicateMember
]
```

例:

```
[State].[New York]
[City].[New York]
```

ショートカットの修飾メンバー名の使用

Essbase では、重複メンバー・アウトライン内のメンバーのショートカットの修飾名が、内部で構築されます。これらは、Administration Services を使用して、メンバー名を右クリックし、「メンバー名を挿入」をクリックすることで、スクリプト内に挿入できます。スクリプト、スプレッドシートまたは MDX クエリーに、ショートカット修飾名を手動で挿入することもできます。

Essbase では、表 14 に示されている構文を使用して、ショートカットの修飾名を構築します。スクリプト、スプレッドシートおよび MDX クエリーでメンバーを参照するときに Essbase で使用される構文と同じ構文を使用するのが最適ですが、必須ではありません。

表 14　ショートカットの修飾名の構文

シナリオ	修飾名構文	例
重複するメンバー名は世代 2 に存在する	[DimensionMember].[DuplicateMember]	[Year].[Jan] [Product].[Jan]
重複するメンバー名はアウトライン内に存在するが、次元内で一意である	[DimensionMember]@[DuplicateMember]	[Year]@[Jan]
重複するメンバー名に一意の親がある	[ParentMember].[DuplicateMember]	[East].[New York]
重複するメンバー名は世代 3 に存在する	[DimensionMember].[ParentMember].[DuplicateMember]	[Products].[Personal Electronics].[Televisions]
重複するメンバー名は、指定された世代またはレベルに存在し、そのメンバーは世代またはレベルで一意の存在です	[DimensionMember]@[GenLevelName]|[DuplicateMember]	[2006]@[Gen1]|[Jan]
シナリオによっては、祖先を区別する方法がショートカットとして使用されます。	[DifferentiatingAncestor].[Ancestors...].[DuplicateMember]	[2006].[Qtr1].[Jan]

一意のメンバー名アウトラインでの修飾メンバー名の使用

修飾メンバー名は、一意のメンバー名アウトライン(重複するメンバー名が使用可能になっていないアウトライン)でのメンバー名の参照にも適用できます。修飾メンバー名を使用すると、共有メンバーと元のメンバーを区別できます。

たとえば、Sample Basic データベースにおいて、メンバー[100-20]は親[100]の下の元のメンバーで、親[Diet]の下に共有メンバーが関連付けられています。次のクエリーで示すように、共有メンバー[100-20]は、一意の名前[Diet].[100-20]を使用して明示的に参照できます:

```
   SELECT
 {Sales}
ON COLUMNS,
 {[[Diet]].[100-20]]} PROPERTIES MEMBER_UNIQUE_NAME
ON ROWS
FROM Sample.Basic;
```

重複するメンバー名の使用

このトピックでは、Administration Services で重複するメンバー名を定義する構文について説明します。

重複するメンバー名を指定するには、次のものを使用します:

- Smart View (Oracle Hyperion Smart View for Office User's Guide を参照)
- API (『Oracle Essbase API リファレンス』を参照)

MaxL および MDX で重複するメンバー名を使用するには、『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： 別名とメンバー名が同一であるのに同じメンバーが示されない場合は、別名による検索がクライアント(Administration Services や API など)でサポートされていません。

Administration Services では、メンバー選択ツールを使用してアウトライン・ツリーから重複するメンバー名を挿入すると、修飾メンバー名が自動的に挿入されます。

重複するメンバー名をエディタに直接入力する場合は、修飾メンバー名を入力し、その修飾名を二重引用符("")で囲みます。たとえば、

```
"[State].[New York]"
```

MDX および MaxL では、修飾メンバー名は引用符で囲みません。Oracle Essbase Administration Services Online Help の「MDX スクリプトでの次元名およびメンバー名の挿入」を参照してください。Administration Services のデータ・プレビュー機能では、メンバー・アウトラインの重複はサポートされていません。

9 次元およびメンバーのプロパティの設定

この章の内容

この章の情報の一部は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 第 61 章「集約ストレージ・アプリケーション、データベースおよびアウトライン」

次元タイプの設定

次元を特定のタイプとしてタグ付けすると、その次元はそのタイプ用に設計された組込み機能にアクセスできます。たとえば、次元を勘定科目として定義した場合は、その次元のメンバーの勘定科目メジャーを指定できます。Essbase では、データベース内の他の次元の前に、2 つのプライマリ次元タイプである時間と勘定科目が計算されます。デフォルトでは、すべての次元は「なし」とタグ付けされます。

次の項では、次元タイプについて説明します。

➤ 次元タイプを設定する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元タイプの設定」を参照してください。

時間次元の作成

データを収集および更新する回数を示すメンバーが含まれている場合は、次元を「時間」としてタグ付けします。たとえば、Sample.Basic データベースでは、年次元は「時間」としてタグ付けされ、その子孫(すべての四半期メンバーと 1 月などの月)も同様にタグ付けされます。時間次元により、期首と期末のタイム・バランスなど、いくつかの会計次元関数も使用可能になります。

次元に「時間」タグを付ける場合のルール:

- アウトラインでは 1 つの次元のみに、「時間」タグを付けることができます。
- 時間次元のすべてのメンバーは、時間プロパティを継承します。
- 「時間」タグが付けられれていない次元に、時間メンバーを追加できます。
- 時間次元を持たないアウトラインを作成できます。

➤ 次元に「時間」タグを付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「時間次元のタグ付け」を参照してください。

会計次元の作成

利益や在庫などの測定するアイテムが含まれている次元には、勘定科目のタグを付けます。

次元に勘定科目タグを付ける場合のルール:

- アウトラインでは 1 つの次元のみに、勘定科目タグを付けることができます。
- 会計次元のすべてのメンバーは、勘定科目プロパティを継承します。
- 会計次元のメンバーが、アウトラインの 2 番目のパスで計算されるように指定できます。157 ページの「2 パス計算の設定」を参照してください。
- 会計次元を持たないアウトラインを作成できます。

➤ 次元に勘定科目のタグを付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「会計次元のタグ付け」を参照してください。

次の各項では、会計次元の組込み関数を説明します。

タイム・バランス・プロパティの設定

会計次元メンバーでタイム・バランス・プロパティを使用すると、Essbase が時間次元内のそのメンバーの親をどのように計算するかに影響します。デフォルトでは、時間次元内の親は、その子の集計と式に基づいて計算されます。たとえば、Sample.Basic データベースでは、第 1 四半期メンバーはその子(1 月、2 月および 3 月)の合計です。ただし、タイム・バランス・プロパティを設定することで、親(第 1 四半期など)が別の形でロール・アップされます。

➤ タイム・バランス・プロパティを設定する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「タイム・バランス・プロパティの設定」を参照してください。

タイム・バランスを「なし」に設定する例

「なし」はデフォルト値です。タイム・バランス・プロパティを「なし」に設定すると、Essbase では時間次元内の親は通常の方法でロール・アップされます。親の値は、その子の式と集計のプロパティが基になります。

タイム・バランスを「期首」に設定する例

親の値に分岐の最初のメンバーの値を表現させる場合、タイム・バランスに「期首」を設定します(通常は期間の先頭)。

たとえば、OpeningInventory というメンバーが、その期間の初めの在庫を表していると仮定します。期間が「Qtr1」の場合、OpeningInventory は「Jan」の初めの在庫を表します。つまり、「Qtr1」の OpeningInventory は「Jan」の OpeningInventory と同じです。たとえば、「Jan」の初めに 50 ケースの「Cola」があった場合、「Qtr1」の初めにも 50 ケースの「Cola」があったことになります。

次の集計例に示すように、OpeningInventory に期首とタグ付けします:

```
   OpeningInventory (TB First), Cola, East, Actual,
   Jan(+), 50

OpeningInventory (TB First), Cola, East, Actual, Feb(+), 60
OpeningInventory (TB First), Cola, East, Actual, Mar(+), 70
OpeningInventory (TB First), Cola, East, Actual,
   Qtr1(+), 50
```

タイム・バランスを「期末」に設定する例

親の値を分岐内の最後のメンバー(期間の最後が多い)にする場合は、タイム・バランスを「期末」と設定します。

たとえば、EndingInventory というメンバーが、その期間の最後の在庫を表していると仮定します。期間が「Qtr1」の場合、EndingInventory は「Mar」の最後の在庫を表します。つまり、「Qtr1」の EndingInventory は「Mar」の EndingInventory と同じです。たとえば、「Mar」の最後に 70 ケースの「Cola」があった場合は、「Qtr1」の最後にも 70 ケースの「Cola」があったことになります。

次の集計例に示すように、EndingInventory に期末とタグ付けします:

```
   EndingInventory (TB Last), Cola, East, Actual, Jan(+), 50
EndingInventory (TB Last), Cola, East, Actual, Feb(+), 60
EndingInventory (TB Last), Cola, East, Actual,
   Mar(+), 70

EndingInventory (TB Last), Cola, East, Actual,
   Qtr1(+), 70
```

タイム・バランスを「平均」に設定する例

親の値で子の平均値を表現する場合、タイム・バランスに「平均」を設定します。

たとえば、AverageInventory というメンバーがその期間の在庫の平均を表していると仮定します。期間が「Qtr1」の場合、AverageInventory は「Jan」、「Feb」および「Mar」の間の在庫の平均を表します。

次の集計例に示すように、AverageInventory に平均とタグ付けします:

```
   AverageInventory (TB Average), Cola, East, Actual, Jan(+),
   60

AverageInventory (TB Average), Cola, East, Actual, Feb(+),
   62

AverageInventory (TB Average), Cola, East, Actual, Mar(+),
   67

AverageInventory (TB Average), Cola, East, Actual,
   Qtr1(+), 63
```

スキップ・プロパティの設定

タイム・バランスに「期首」、「期末」または「平均」を設定した場合には、Essbase で、欠落した値や値が 0 のときの処置を指定するために、スキップ・プロパティを設定します。

表 15 で、それぞれの設定によって、欠落した値やゼロ値のときの Essbase での処置がどのように決定されるかを説明します。

表 15　スキップ・プロパティ

設定	Essbase のアクション
なし	親の値の計算時にデータはスキップしません。
欠落	親の値の計算時に#MISSING データをスキップします。
ゼロ	親の値の計算時にゼロと等しいデータはスキップします。
欠落およびゼロ	親の値の計算時に、#MISSING データおよびゼロと等しいデータをスキップします。

メンバーに期末とマークし、スキップ・プロパティを欠落、または欠落およびゼロに設定すると、その期間の親は欠落でない最後の子と一致します。次の例では、「Mar」が値を持たないため、EndingInventory は「Feb」の値を基にしています。

```
   Cola, East, Actual, Jan, EndingInventory (Last),  60
Cola, East, Actual,
   Feb
   , EndingInventory (Last),
   70

Cola, East, Actual, Mar, EndingInventory (Last),  #MI
Cola, East, Actual,
```

```
Qtr1
, EndingInventory (Last),
70
```

差異レポート・プロパティの設定

差異レポート・プロパティでは、メンバー式の中に@VAR 関数または@VARPER 関数があるメンバーでの、Essbase による実績データと予算データの差異の計算方法を決定します。企業の支出を表すすべてのメンバーに、支出プロパティが必要です。

ある期間の支出の予算を設定する場合、実際の支出は予算より少なくなる必要があります。実際の支出が予算を上回る場合、差異は負数になります。@VAR 関数では、「予算 - 実績」が計算されます。たとえば、支出予算が$100 で、実際には$110 出費した場合、差異は-10 になります。

売上などの支出外アイテムの予算を設定する場合、売上実績は予算より多くなる必要があります。売上実績が予算より少ない場合、差異は負数になります。@VAR 関数では、「実績-予算」が計算されます。たとえば、売上予算が$100 で、実際の売上が$110 の場合、差異は 10 になります。

デフォルトでは、メンバーは支出外です。

➤ 差異レポート・プロパティを設定する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「差異レポート・プロパティの設定」を参照してください。

Essbase 通貨換算プロパティの設定

通貨換算プロパティは、為替レートのカテゴリを定義します。これらのプロパティは、会計次元のメンバーに対して、通貨データベースでのみ使用されます。第 14 章「通貨換算アプリケーションの設計および作成」を参照してください。

➤ 通貨換算プロパティを設定する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「勘定科目メンバーへの通貨カテゴリの割当て」を参照してください。

国次元の作成

複数の国でのビジネス・アクティビティを追跡するには、国次元を使用します。たとえば、米国とカナダでのビジネス・アクティビティを追跡する場合、国次元には州、地域および国が含まれます。次元に国のタグが付けられている場合は、通貨名プロパティを設定できます。通貨名プロパティでは、この市場地域で使用される通貨のタイプが定義されます。

国次元では、各メンバーで使用される通貨を指定できます。たとえば、Essbase に同梱されている国際的なアプリケーションとデータベースでは、カナダには 3 つの市場(バンクーバー、トロントおよびモントリオール)があり、そこではカナダ・ドルが使用されています。

この次元タイプは、通貨換算アプリケーションで使用できます。第 14 章「通貨換算アプリケーションの設計および作成」を参照してください。

➤ 次元に国のタグを付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「国次元のタグ付け」を参照してください。

通貨パーティションの作成

アプリケーションで定義されている基本通貨から現地通貨メンバーを分離するには、通貨パーティション・メンバーを使用します。分析用の基本通貨が米ドルの場合、現地通貨メンバーには、その地域の通貨タイプ(カナダ・ドルなど)に基づく値が含まれることがあります。

この次元タイプは、通貨換算アプリケーションで使用できます。第 14 章「通貨換算アプリケーションの設計および作成」を参照してください。

➤ 次元に通貨パーティション・タグを付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「通貨パーティションの作成」を参照してください。

属性次元の作成

データを標準次元の特徴に基づいてレポートおよび集約するには、属性次元を使用します。たとえば、Sample.Basic データベースでは、Product 次元は Ounces 属性次元に関連付けられています。Ounces 属性次元のメンバーは、オンス単位でのサイズに基づいて製品を分類します。

属性次元の使用ルールは、第 10 章「属性の操作」を参照してください。

➤ 次元に属性のタグを付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「属性次元のタグ付け」を参照してください。

メンバー集計の設定

表 16 にリストされているメンバー集計プロパティにより、子がその親にロール・アップされる方法が決まります。デフォルトでは、新しいメンバーには加算(+)演算子が指定されます。つまり、メンバーが追加されることを意味します。たとえば、1 月、2 月および 3 月の数字が追加され、その結果がそれらの親である第 1 四半期に保管されます。

注： Essbase では、属性次元のメンバーを持つ集計プロパティは使用されません。176 ページの「属性データの計算」を参照してください。

表 16　集計演算子

演算子	説明
+	前に実行された他のメンバーの計算結果に、メンバーを加算します。+はデフォルトの演算子です。
-	メンバーに-1 を乗じて、その結果を、前に実行された他のメンバーの合計に加算します。
*	前に実行された他のメンバーの計算結果を、メンバーに乗算します。
/	前に実行された他のメンバーの計算結果を、メンバーで除算します。
%	前に実行された他のメンバーの計算の合計でメンバーを除算します。その結果に 100 を乗算して百分率にします。
~	メンバーは、その親の集計には使用されません。
^	メンバーは、どの次元の集計にも使用されません。

➤ メンバー集計プロパティを設定する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「メンバー集計プロパティの設定」を参照してください。

異なる演算子を持つ場合のメンバーの計算

兄弟が異なる演算子を持つ場合、Essbase ではデータがトップダウンの順序で計算されます。次のメンバーについて考えてみます:

```
  Parent1
Member1 (+)  10
Member2 (+)  20
Member3 (-)  25
Member4 (*)  40
Member5 (%)  50
Member6 (/)  60
Member7 (~)  70
```

Essbase で次のように、「Member1」から「Member4」が計算されます:

```
  (((Member1 + Member2) + (-1)Member3) * Member4) = X
(((10 + 20) + (-25)) * 40) = 200
```

この計算の結果が X の場合、「Member5」は次のように集計されます:

```
  (X/Member5) * 100 = Y
(200/50) * 100 = 400
```

「Member1」から「Member4」の計算の結果が Y の場合、「Member6」は次のように集計されます:

```
   Y/Member6 = Z
400/60 = 66.67
```

「Member7」は非集計(~)に設定されているため、Essbase では集計時に「Member7」が無視されます。

メンバーによるデータ値の保管方法の決定

Essbase でメンバーのデータ値が格納される方法とタイミングを決定できます。たとえば、Essbase にユーザーから要求された場合のみメンバーの値を計算し、その後でデータ値を破棄するように指定できます。表 17 で、各ストレージ・プロパティについて説明します。

表 17　ストレージ・プロパティの選択

ストレージ・プロパティ	動作	参照
保管	メンバーのデータ値を保管します。	146 ページの「保管済メンバーの理解」
動的計算および保管	ユーザーが要求するまでデータ値を計算せず、計算後にデータ値を保管します。	147 ページの「動的計算メンバーの理解」
動的計算	ユーザーが要求するまでデータ値を計算せず、計算後にデータ値を破棄します。	147 ページの「動的計算メンバーの理解」
共有しない	メンバーを暗黙的に共有できないようにします。 共有しないとタグ付けされているメンバーは、明示的に共有することしかできません。メンバーを明示的に共有するには、同じ名前を持つ共有メンバーを作成し、そのメンバーに共有とタグ付けします。	151 ページの「暗黙的な共有の理解」
ラベルのみ	参照用のメンバー(すなわち、データ値を含まないメンバー)を作成します。	147 ページの「ラベルのみメンバーの理解」
共有メンバー	値をメンバー間で共有します。たとえば、Sample.Basic データベースでは、「100-20」のメンバーが「100」の親の下に保管され、「Diet」の親の下で共有されます。	148 ページの「共有メンバーの理解」

➤ メンバー・ストレージ・プロパティを設定する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「メンバー・ストレージ・プロパティの設定」を参照してください。

保管済メンバーの理解

保管済メンバーには、計算後にデータベース内のメンバーとともに保管される計算済の値が含まれます。デフォルトでは、メンバーは保管済メンバーとして設定されます。

➤ メンバーを保管済として定義する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「メンバー・ストレージ・プロパティの設定」を参照してください。

動的計算メンバーの理解

メンバーが動的計算の場合は、Essbase ではユーザーが要求するまでそのメンバーの値は計算されません。ユーザーが表示した後は、そのメンバーの値は Essbase に保管されません。メンバーに動的計算および保管のタグが付けられている場合は、Essbase で動的計算メンバーの場合と同じ操作が実行されますが、データ値は保管されます。第 27 章「データ値の動的計算」を参照してください。

Essbase は属性次元のメンバーに自動的に動的計算をタグ付けします。この設定は変更できません。

➤ メンバーに動的計算タグを付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「メンバー・ストレージ・プロパティの設定」を参照してください。

ラベルのみメンバーの理解

ラベルのみメンバーには、関連付けられた値はありません。これらのメンバーは、メンバーのグループ化や、Smart View からの容易なナビゲーションとレポートのために使用されます。通常、ラベルのみのメンバーには、非集計プロパティを指定する必要があります。144 ページの「メンバー集計の設定」を参照してください。

属性をラベルのみメンバーと関連付けることはできません。属性の関連付けを持つ基本次元メンバーをラベルのみとしてタグ付けすると、Essbase により属性の関連付けが削除され、警告メッセージが表示されます。

ラベルのみメンバーの子孫には動的計算タグが付けられません。次の例では、アウトラインの確認中に、Essbase により、ChildB をラベルのみとしてタグ付けできないことを示すエラー・メッセージが発行されます:

```
ParentA = Label Only
 ChildB = Label Only
   DescendantC = Dynamic Calc
```

この問題を解決するには、DescendantC にデータの保管タグか、動的計算および保管タグを付けます。

➤ メンバーにラベルのみタグを付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』のメンバー・ストレージ・プロパティの設定に関する項を参照してください。

共有メンバーの理解

共有メンバーに関連付けられたデータ値は、同一名を持つ他のメンバーから取得します。共有メンバーは、他のメンバーに含まれているデータへのポインタを保管し、そのデータは一度のみ保管されます。メンバーを共有として定義するには、同一名の非共有メンバーが実際に存在する必要があります。たとえば、Sample.Basic データベースでは、「100」の下の「100-20」メンバーはそのメンバーのデータを保管します。「Diet」の下の「100-20」メンバーはその値をポイントしています。

共有メンバーは、通常、複数の親にまたがる同じメンバーを計算するために使用されます。たとえば、「100」と「Diet」の両方の親にある「Diet Cola」メンバーを計算する場合です。

共有メンバーを使用することで、次元全体でメンバーを繰り返し使用できます。Essbase でデータ値が保管されるのは一度のみですが、複数の場所に表示されます。データ値を一度のみ保管することで、スペースが節約され、処理効率が向上します。

➤ メンバーに共有タグを付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「メンバー・ストレージ・プロパティの設定」を参照してください。

次の項では、共有メンバーの詳細について説明します。

注： 同じ名前を持つメンバーは、共有メンバーではなく重複メンバーであることがあります。第 8 章「重複メンバーのアウトラインの作成および使用」を参照してください。

共有メンバーの規則の理解

共有メンバーを作成するときの規則は次のとおりです:

- 共有メンバーは同じ次元に属する必要があります。たとえば、Sample.Basic データベース内の 100-20 メンバーは、両方とも製品次元内に存在します。
- 共有メンバーは子を持つことができません。
- 同じ名前を持つことができる共有メンバーの数は無制限です。
- 共有メンバーに UDA または式を割り当てることはできません。
- 重複メンバー名を持つメンバーに対し共有メンバーを作成できます。共有メンバーを作成する対象の重複メンバー名を指定します。共有メンバーの基となる、重複メンバーの修飾名は、「メンバー・プロパティ」ダイアログ・ボックスに表示されます。Oracle Essbase Administration Services Online Help の「共有メンバーの定義」を参照してください。
- 属性を共有メンバーに関連付けることはできません。

- 勘定科目プロパティを共有メンバーに割り当てる場合は、共有メンバー上で勘定科目プロパティを変更しても、それらの勘定科目プロパティの値は基本メンバーから取得されます。
- 別名を共有メンバーに割り当てることができます。
- 実際のメンバーは、次元内でその共有メンバーの前に配置する必要があります。
- 属性計算の一部となる実際のメンバーと共有メンバーの関係が複雑にならないようにしてください。そうしないと、計算で予期しない結果が戻されることがあります。181 ページの「属性計算と共有メンバーの理解」を参照してください。

注： 共有メンバーとその基になるメンバーは、同じ親の下には作成できません。重複メンバー・アウトラインでは、兄弟は一意である必要があります。

注： グリッド・クライアント(Smart View など)では、共有メンバーに修飾名([Parent].[Child]など)を付けて表示するように指定できるため、共有メンバーと基本メンバーを簡単に区別できます。重複するメンバー名を使用できるようにアウトラインを設定していない場合でも、共有メンバーを修飾名とともに表示できます。また、修飾メンバー名を使用して、グリッド内またはメンバー選択を使用して共有メンバーを検索することもできます。

ドリルダウン時の共有メンバーの取得の理解

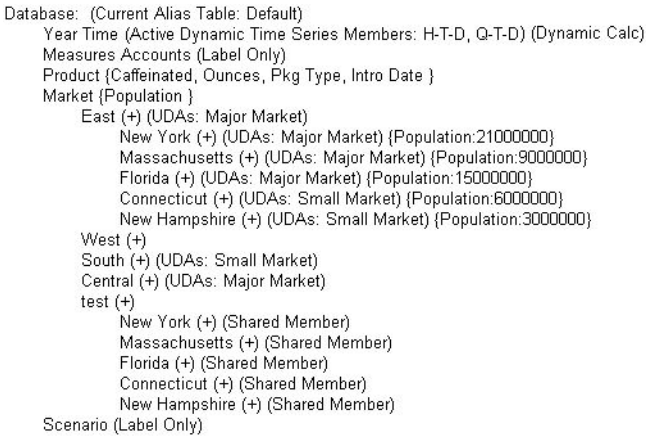
Essbase では、スプレッドシート内の共有メンバーの場所に応じて、ドリルダウン中に共有メンバーが取得されます。Essbase では、このタイプの取得の際は次の 3 つのルールに従います:

- デフォルトでは、Essbase によって保管済メンバー(対応する共有メンバーではなく)が取得されます。
- Essbase では、スプレッドシートの一番下から取得が開始されます。
- 共有メンバーの親が、共有メンバーのいずれかに対応する保管済メンバーの兄弟である場合、Essbase ではその保管済メンバーが取得されます。

単一次元の共有メンバーの例

Sample.Basic アウトラインの East 次元のメンバーをすべて共有メンバーとして持つ test 次元を作成した場合、アウトラインは、図 39 に示すような内容になります:

図 39　単一次元の共有メンバー



「East」の子のみを取得した場合、Essbase ではデフォルトで保管済メンバーが取得されるため、結果はすべて保管済メンバーからのものになります。

ただし、スプレッドシート内でその上の「test」の子を持つデータを取得すると、Essbase では共有メンバーが取得されます:

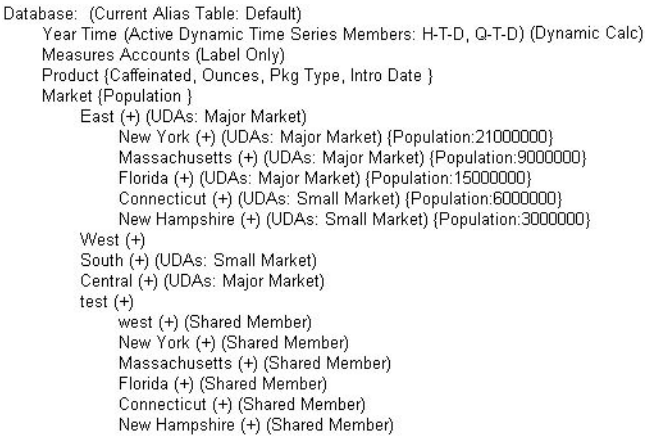
```
   New York
Massachusetts
Florida
Connecticut
New Hampshire
test
```

「test」をその最後の 2 つの子の上に移動すると、Essbase では最初の 3 つの子が共有メンバーとして取得されますが、最後の 2 つの子は保管済メンバーとして取得されます。同様に、リストの途中に共有メンバーの兄弟でないメンバーを挿入すると(たとえば、「Florida」と「Connecticut」の間に「California」を挿入)、Essbase では兄弟でないメンバーと親の間でのみ共有メンバーが取得されます(この場合は、「California」と「test」の間)。

世代間共有メンバーの取得の例

図 40 に示すように、Sample.Basic アウトラインを変更して、対応する保管済メンバーが、その親の兄弟であるような共有メンバーを作成できます:

図 40　世代間共有メンバーの取得



次の順序で共有メンバーを含むスプレッドシートを作成すると、Essbase ではすべての共有メンバーが取得されます。ただし例外として、共有メンバー「west」ではなく、保管済メンバー「West」が取得されます:

```
   West
New York
Massachusetts
Connecticut
New Hampshire
test
```

Essbase によって、この順番でメンバーが取得されるのは、「test」が「west」の親であり、「west」に対応する保管済メンバー「West」の兄弟であるためです。

暗黙的な共有の理解

共有メンバー・プロパティでは、共有データ関係を明示的に定義します。一部のメンバーは、明示的に共有として設定しなくても共有されます。これらのメンバーが暗黙的な共有メンバーです。

Essbase では、次のような場合に共有メンバー関係とみなされます(暗黙的):

- **親が子を 1 つしか持たない場合。**この場合、親と子は同じデータを持ちます。Essbase では、子の集計プロパティは無視され、データは一度しか保管されません。つまり、親は子と暗黙的な共有関係を持ちます。次の例では、親 500 には子 500-10 しかないため、親はその子の値を共有します:

```
     500 (+)
500-10 (+)
```

- **親に集計される子を親が 1 つしか持たない場合。**親が 4 つの子を持っていても、そのうち 3 つは集計なしとマークされている場合、親と集計される子は同じデータを持ちます。Essbase では、子の集計プロパティは無視され、データは一度しか保管されません。つまり、親は子と暗黙的な共有関係を持ちます。次の例では、親 500 にロール・アップされる子は 500-10 のみです。他の

子は集計なし(~)とマークされているため、親は暗黙的に 500-10 の値を共有します。

```
     500 (+)
500-10 (+)
500-20 (~)
500-30 (~)
```

メンバーを暗黙的に共有しないようにする場合は、親に「非共有」とマーク付けして、かわりにデータが複製されるようにします。共有メンバーの動作の説明は、148 ページの「共有メンバーの理解」を参照してください。

別名の設定

別名および別名テーブルに関する情報は、ブロック・ストレージ・データベースおよび集約ストレージ・データベースに適用されます。

別名は、メンバーまたは共有メンバーの代替名です。たとえば、Sample.Basic データベースの製品次元のメンバーは、100 などの製品コードと「Cola」などのよりわかりやすい別名の両方によって識別されます。別名は、別名テーブルに保管されます。別名によってアウトラインやレポートを読みやすくすることができます。

別名テーブルを使用して、メンバーに複数の別名を設定できます。たとえば、様々な種類のレポートに様々な別名を使用できます。例として、100-10 はユーザーにとっては「Cola」の方がわかりやすく、企業幹部にとっては「The Best Cola」の方がわかりやすいことがあります。次のリストは、Sample.Basic データベース内の、2 つの記述的な別名を持つ製品を示しています:

```
   Product  Default       Long Names

100-10  Cola         The Best Cola
100-20  Diet Cola       Diet Cola with Honey
100-30  Caffeine Free Cola  All the Cola, none of the Caffeine
```

Essbase では、ハイブリッド分析対応メンバーの別名はサポートされていません。

別名テーブル

別名は、データベース・アウトラインの一部として、1 つ以上のテーブルに保管されます。別名テーブルでは、名前が付けられた特定の別名セットをメンバー名にマップします。データベース・アウトラインを作成すると、Essbase によって Default という空の別名テーブルが作成されます。他の別名テーブルを作成しない場合、作成する別名は Default 別名テーブルに保管されます。

アウトライン・メンバーのセットごとに別名テーブルを作成できます。アウトラインを表示したりデータを取得したりするときは、別名テーブル名を使用して、表示する別名のセットを指定できます。アウトラインを表示しているときに表示する名前が含まれる別名テーブルを識別することを、別名テーブルをアクティブ

な別名テーブルにすると言います。154 ページの「アクティブな別名テーブルの設定」を参照してください。

Unicode モードのアプリケーションでは、ユーザーの言語ごとに別個の別名テーブルを設定することで、ユーザーが自分の言語でメンバー名を表示できます。第 43 章「Essbase の Unicode 実装の理解」を参照してください。

別名の作成

メンバーの別名を指定できます。別名は、メンバー名と同じルールに従う必要があります。1202 ページの「次元、メンバーおよび別名の命名規則」を参照してください。

既存の別名テーブルに別名を作成するには、次のいずれかの方法を使用します:

- アウトラインの編集中に、別名をメンバーに手動で割り当てる方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元およびメンバーの別名の作成」を参照してください。
- 次元構築およびデータ・ソースを使用して別名を別名テーブルに追加する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「別名を追加するためのルール・ファイルの定義」を参照してください。
- 事前定義のフォーマットで作成された別名テーブルのソース・ファイルから別名の値をインポートする方法は、156 ページの「別名テーブルのインポートおよびエクスポート」を参照してください。

別名テーブルの作成と管理

名前付きの別名テーブルを使用すれば、様々な状況で様々な別名を表示できます。参照:

別名テーブルの作成

別名テーブルには、アウトライン・メンバーに使用する別名のリストが含まれます。次の要件が適用されます:

- 1 つのブロック・ストレージまたは集約ストレージ・アウトラインに最大 32 個の別名テーブルを作成できます。

- 別名テーブル名の命名ルールは、次元の命名ルールと同じです。

  1202 ページの「次元、メンバーおよび別名の命名規則」を参照してください。
- 名前の長さの制限は、アプリケーションの Unicode 関連のモードによって異なります。

  付録 A「制限」を参照してください。
- システム生成のデフォルトの別名テーブルの名前(Default)は変更できませんが、自分で作成する別名テーブルの名前は変更できます。
- **オプション。**別名テーブルには複数の言語コードを指定できます。別名テーブルの作成時には、言語コードは指定されません。154 ページの「別名テーブルの言語コードの操作」を参照してください。

➤ 別名テーブルを作成する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「別名テーブルの作成」を参照してください。

新しい別名テーブルは空です。別名を別名テーブルに追加し、それらの別名をメンバーに割り当てる方法は、153 ページの「別名の作成」を参照してください。

別名テーブルに言語コードを割り当てる方法は、154 ページの「別名テーブルの言語コードの操作」を参照してください。

別名テーブルの言語コードの操作

別名テーブルには複数の言語コードを指定できます。別名テーブルの作成時には、言語コードは指定されません。別名テーブルに言語コードを割り当てるには、バージョン C-API または VB-API の SetAliasTableLanguage API を使用します。(デフォルトの別名テーブルに言語コードを設定することはできません。)

別名テーブルに関連付けられた一連の言語コードを取得するには、バージョン C-API または VB-API の GetAliasTableLanguages API を使用します。

別名テーブルに関連付けられた一連の言語コードを消去するには、バージョン C-API または VB-API の ClearAliasTableLanguages API を使用します。

別名テーブルから別名を消去すると、別名テーブルから言語コードが削除されます。

別名テーブルからコピーすると、コピーした別名テーブルから言語コードが削除されます。

別名テーブルの名前を変更した場合、言語コードは名前が変更された別名テーブルに保持されます。

『Oracle Essbase API リファレンス』を参照してください。

アクティブな別名テーブルの設定

アクティブな別名テーブルには、Essbase でアウトライン内に現在表示されている別名が含まれます。

➤ アウトラインの別名テーブルのリストを表示し、現在の別名テーブルを設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アウトライン・エディタのアクティブな別名テーブルの設定	Oracle Essbase Administration Services Online Help
MaxL	**query database** **alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	LISTALIASES SETALIAS	『Oracle Essbase テクニカル・リファレンス』

別名テーブルのコピー

別名テーブルをコピーするには、そのテーブルがデータベース・ディレクトリ内で存続している必要があります。データベース・ディレクトリ内に存続していないアーティファクトをコピーするには、EXPORT ESSCMD コマンドを使用します。

別名テーブルをコピーすると、その別名テーブルに関連付けられている言語コードは、コピー先の別名テーブルから削除されます。154 ページの「別名テーブルの言語コードの操作」を参照してください。

➤ 別名テーブルをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	別名テーブルのコピー	Oracle Essbase Administration Services Online Help
MaxL	**alter object**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYOBJECT	『Oracle Essbase テクニカル・リファレンス』

別名テーブルの名前変更

別名テーブルの名前を変更すると、そのテーブルに関連付けられている言語コードは、新しい名前の別名テーブルでも保持されます。154 ページの「別名テーブルの言語コードの操作」を参照してください。

➤ 別名テーブルの名前を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	別名テーブルの名前変更	Oracle Essbase Administration Services Online Help
MaxL	**alter object**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	RENAMEOBJECT	『Oracle Essbase テクニカル・リファレンス』

別名テーブルの消去および削除

アウトラインから別名テーブルを削除できます。また、別名テーブル自体は削除しないで、別名テーブルから別名をすべて消去できます。別名テーブルを消去または削除する方法は、Oracle Essbase Administration Services Online Help の「別名テーブルの削除および消去」を参照してください。

別名テーブルから別名を消去すると、その別名テーブルに関連付けられている言語コードが削除されます。154 ページの「別名テーブルの言語コードの操作」を参照してください。

別名テーブルのインポートおよびエクスポート

正しくフォーマットされたテキスト・ファイルは、Essbase に別名テーブルとしてインポートできます。別名テーブルのインポート・ファイルの拡張子は.alt です。別名テーブルのインポート・ファイルは、次の構造にする必要があります:

- ファイルの最初の行は$ALT_NAME で始まります。1 つまたは 2 つのスペースを追加し、その後に別名テーブルの名前を続けます。別名テーブル名に空白文字が含まれている場合は、名前を一重引用符で囲みます。
- ファイルの最後の行は、$END にする必要があります。
- 最初の行と最後の行の間の各行には、1 つ以上のスペースまたはタブで区切られた 2 つの値が含まれます。最初の値は既存のアウトライン・メンバーの名前にする必要があり、2 番目の値はそのアウトライン・メンバーの別名です。
- 空白またはアンダースコアが含まれるメンバー名や別名は、二重引用符で囲む必要があります。

次に別名テーブルのインポート・ファイルの例を示します:

```
   $ALT_NAME  'Quarters'
Qtr1  Quarter1
Jan  January
Feb  February
Mar  March
$END
```

別名テーブルを、Essbase アウトラインからテキスト・ファイルにエクスポートできます。エクスポート・ファイルには、別名とそれに対応するメンバー名(重複メンバーの修飾メンバー名)が含まれます。

➤ 別名テーブルをインポートまたはエクスポートするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	別名テーブルのインポート 別名テーブルのエクスポート	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』

ツール	トピック	場所
ESSCMD	LOADALIAS UNLOADALIAS	『Oracle Essbase テクニカル・リファレンス』

2 パス計算の設定

Essbase では、デフォルトでアウトラインがボトム・アップで計算されます。つまり、最初に子の値が計算され、次に親の値が計算されます。ただし、場合によっては、子の値が親の値やアウトライン内の他のメンバーの値に基づいていることがあります。これらのメンバーの正しい値を取得するため、Essbase では、最初にアウトラインを計算してから、他のメンバーの計算済の値に依存するメンバーを再計算する必要があります。アウトライン内の 2 番目のパスで計算されるメンバーは、2 パス計算と呼ばれます。

976 ページの「ボトムアップ計算の使用」を参照してください。

たとえば、売上高と利益率の比率を計算するには、Essbase で、まず利益率(売上高などの子に基づく親メンバー)を計算する必要があります。新しく計算された利益率の数値に基づいて比率を計算するには、利益率 %の比率メンバーを 2 パス計算としてタグ付けします。Essbase によって、データベースが一度計算され、その後比率メンバーが再度計算されます。この計算によって正しい結果が導き出されます。

2 パス計算は、属性以外のメンバーに与えることができるプロパティですが、実際には次のメンバーに対してのみ作用します:

- 会計次元メンバー
- 動的計算メンバー
- 動的計算および保管メンバー。

2 パス計算が他のメンバーに割り当てられている場合、Essbase によって、その 2 パス計算は無視されます。

➤ メンバーに 2 パスのタグを付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「2 パス計算プロパティの設定」を参照してください。

式の作成

式は、標準次元とメンバーに適用できます。属性次元とそのメンバーには式は設定できません。式により、Essbase でのアウトライン・データの計算方法が決まります。第 23 章「ブロック・ストレージ・データベース用の式の作成」を参照してください。

➤ 次元またはメンバーに式を追加する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アウトライン内の式の作成および編集」を参照してください。

世代およびレベルの命名

アウトライン内の世代とレベルに対して、世代またはレベルを説明する単語や句などを使用して名前を作成できます。たとえば、アウトライン内のすべての市に「Cities」という世代名を作成できます。59 ページの「次元とメンバーの関係」を参照してください。

計算スクリプトまたはレポート・スクリプトで、メンバー名のリストと世代またはレベル番号のリストのいずれかを指定する必要がある場合は、世代およびレベル名を使用します。たとえば、計算スクリプトの計算を特定の世代のすべてのメンバーに限定できます。第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。

それぞれの世代またはレベルには、1 つの名前のみを定義できます。世代とレベルに名前を付ける場合は、メンバーと同じ命名ルールに従います。1202 ページの「次元、メンバーおよび別名の命名規則」を参照してください。

➤ アウトライン・エディタを使用して世代およびレベルに名前を付ける方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「世代およびレベルの名前付け」を参照してください。

UDA の作成

メンバーにユーザー定義の属性(UDA)を作成できます。たとえば、「借方」という UDA を作成できます。UDA は次の場合に使用します:

- 計算スクリプト。UDA を定義した後、計算スクリプト内でその UDA についてメンバーをクエリーできます。たとえば、「借方」という UDA を持つすべてのメンバーに-1 を掛けて、それらのメンバーが正と負のいずれか(データが現在どのように格納されているかによって異なる)として表示されるようにできます。第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。
- データのロード。データをデータベースにロードするとき、その UDA に基づいてデータの符号を変更できます。314 ページの「フィールド符号の反転」を参照してください。

計算を実行したり、属性値に基づいてデータを選択的に取得したり、スプレッドシートでの完全なクロス集計、ピボットおよびドリルダウンを提供したりする場合は、UDA のかわりに属性次元を作成します。169 ページの「属性と UDA の比較」を参照してください。

注： 集約ストレージ・データベースにおいて、UDA を使用してメンバーのグループを定義すると Essbase 関数の実行速度が大幅に低下します。このパフォーマンス低下を回避するには、属性次元を使用してメンバーのグループを定義します。

UDA を作成するときの規則は次のとおりです:

- メンバーごとに複数の UDA を定義できます。
- 単一のメンバーに対して、同じ UDA を 2 回設定することはできません。
- 異なるメンバーに、同じ UDA を設定できます。
- UDA 名は、メンバー名、別名、レベル名または世代名と同じ名前にできます。メンバーと同じ命名ルールに従います。1202 ページの「次元、メンバーおよび別名の命名規則」を参照してください。
- 共有メンバーには UDA を作成できません。
- 属性次元のメンバーには UDA は作成できません。
- UDA は指定したメンバーのみに適用されます。メンバーの子孫と祖先が同じ UDA を自動的に受け取ることはありません。

➤ UDA をメンバーに追加する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「UDA の操作」を参照してください。

コメントの追加

次元とメンバーにはコメントを追加できます。コメントには最大 255 文字を含めることができます。アウトライン・エディタでは、次元またはメンバーの右側に、次のフォーマットでコメントが表示されます:

```
/* comment */
```

➤ 次元またはメンバーにコメントを追加する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元およびメンバーのコメントの設定」を参照してください。

10 属性の操作

この章の内容

この章の情報の一部は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 第 61 章「集約ストレージ・アプリケーション、データベースおよびアウトライン」

属性の使用についてのその他情報については、次を参照してください:

- 342 ページの「属性次元の構築と属性の関連付け」
- 第 15 章「パーティション・アプリケーションの設計」
- 第 16 章「パーティションの作成および管理」
- 第 34 章「レポート・スクリプトの作成」

属性の作成のプロセス

属性は、製品のサイズや色などのデータ特性を記述します。属性を使用して、次元のメンバーをその特性に基づいてグループ化したり分析したりできます。この章では、Essbase サーバーのアウトラインで属性を作成および管理する方法について説明します。

アウトライン・エディタで属性を操作する場合は、次のタスクを実行します:

1. 次元を作成します。

   128 ページの「アウトラインへの次元およびメンバーの追加」を参照してください。アウトライン内で、属性次元をすべての標準次元の後に配置します。

2. 次元に、属性次元タグを付け、テキスト、数値、ブールまたは日付の属性次元タイプを設定します。

   144 ページの「属性次元の作成」を参照してください。

3. メンバーを属性次元に追加します。

   128 ページの「アウトラインへの次元およびメンバーの追加」を参照してください。

4. 基本次元を属性次元に関連付けます。

   165 ページの「属性次元の関連付けの規則の理解」を参照してください。

5. 基本次元のメンバーを属性次元のメンバーに関連付けます。

   165 ページの「属性メンバーの関連付けの規則の理解」を参照してください。

6. 必要な場合は、属性計算を設定します。

   176 ページの「属性データの計算」を参照してください。

属性の理解

Essbase の属性機能を使用して、次元の観点からのみでなく、それらの次元の特性、つまり属性の観点からもデータを取得して分析できます。たとえば、製品の収益性をサイズやパッケージに基づいて分析でき、さらに各市場地域の人口などの市場属性を分析に組み込むことで、より効果的な結論を下すことができます。

このような分析により、小規模市場(人口 6,000,000 未満)では缶で販売されるカフェイン抜き飲料の売上が予想よりも低いことがわかる場合があります。さらに、同様の市場区分での様々な製品の売上と収益の最小値や最大値など、特定の属性条件によって分析を絞り込むことができます。

次に示すような属性による分析によって、深さと奥行きのある見方ができるので、より適切な情報に基づいた決定を下すことができます:

- 共通の特徴(属性)に基づくデータの選択、集約およびレポートを行うことができます。
- 属性をテキスト型、数値型、ブール型または日付型と定義することによって、AND、OR および NOT 演算子や、<、>および=比較演算子など、型に関連した関数を使用したデータのフィルタ(選択)が可能になります。
- 数値属性タイプを使用すれば、属性範囲別に統計値をグループ化できます。たとえば<500,000、500,000-1,000,000、>1,000,000 などの人口グループにグループ化できます。
- Essbase によって自動的に作成される属性計算次元によって、属性データの合計、カウント、最小値または最大値、および平均値を表示できます。たとえば、スプレッドシートに「Avg」と「Bottle」を入力すると、Essbase によってシート上の列と行のすべての交差の平均売上の計算値が取得されます。
- 計算スクリプトやメンバー式で数値属性値を使用して計算を実行できます。たとえば、サイズがオンスで示された製品についてオンス当たりの収益性を判断できます。
- 同一次元の属性データのクロス集計を作成したり、スプレッドシートで詳細データのピボットやドリル・ダウンを行ったりすることができます。

属性のクロス集計は、同じ次元の属性全体のデータ集計を表示するレポートまたはスプレッドシートです。下の例のクロス集計では、製品のパッケージを列で、製品のサイズ(オンス)を行で表示しています。それらの交差には、パッケージ・タイプとサイズの各組合せに対する収益が示されています。

この情報から、「Florida」市場では、どのサイズとパッケージの組合せが最も収益が高かったかがわかります。

```
Product Year Florida Profit Actual
    Bottle   Can    Pkg Type
    ======   =====   ========
 32    946     N/A     946
 20    791     N/A     791
 16    714     N/A     714
 12    241    2,383    2,624
Ounces  2,692    2,383    5,075
```

属性次元の理解

製品は、製品のサイズやパッケージなど、その製品の特性を示す属性を持ちます。属性メンバーは属性次元に存在します。アウトライン内の属性次元には、その名前の横に「属性」という語が付いています。

図 41 では、Product 次元と 3 つの属性次元(Caffeinated、Ounces および Pkg Type)を使用する Sample.Basic アウトラインの一部を示しています。Product 次元の右側の Caffeinated、Ounces および Pkg Type で、これらの属性次元が Product 次元に関連付けられていることを示しています。

図 41　基本次元と属性次元を示すアウトライン

```
Product {Caffeinated, Ounces, Pkg Type }
     100 (+)
          100-10 (+) {Caffeinated:True, Ounces:12, Pkg Type:Can }
          100-20 (+) {Caffeinated:True, Ounces:12, Pkg Type:Can }
          100-30 (+) {Caffeinated:False, Ounces:16, Pkg Type:Bottle }
     200 (+)
     300 (+)
     400 (+)
     Diet (~)
Caffeinated Attribute
     True
     False
Ounces Attribute
     32
     20
     16
     12
Pkg Type Attribute
     Bottle
     Can
```

標準次元とは、属性次元でない次元のことです。属性次元が標準次元に関連付けられている場合、その標準次元はその属性次元の基本次元になります。図 41 では、Product 次元は Caffeinated、Ounces および Pkg Type 属性次元の基本次元です。

注： 属性次元とメンバーは動的計算です。そのため、Essbase では属性情報は取得時に計算されます。属性データはデータベースには保管されません。

属性次元のメンバーの理解

属性次元のメンバーは、関連する基本次元のメンバーの潜在的な属性です。基本次元を属性次元に関連付けた後に、基本次元のメンバーを関連する属性次元のメンバーに関連付けます。市場次元メンバーの「Connecticut」は、「Population」属性次元の「6000000」メンバーに関連付けられます。これによって、「6000000」が「Connecticut」の属性になります。

アウトラインでは、基本次元メンバーの横にある情報がそのメンバーの属性を示しています。たとえば、図 41 では、製品「100-10」は、カフェイン入り、12 オンスの容器で販売、缶入りで販売という 3 つの属性を持つことを示しています。

基本次元および属性次元とメンバーの規則の理解

属性次元とその基本次元のメンバーに関する規則は次のとおりです。

- 疎次元にのみ属性次元タグを付けることができます。
- それぞれの属性次元を基本次元として標準の疎次元に関連付けると、サーバーにアウトラインを保存できます。
- 属性次元はアウトラインの最後の次元にする必要があります。
- 属性次元には、テキスト、数値、ブールまたは日付という型の設定があります。デフォルトの設定はテキストです。型は次元レベルで割り当てられますが、次元のレベル 0 のメンバーのみに適用されます。166 ページの「属性タイプの理解」を参照してください。
- 次元から属性タグを削除すると、Essbase によって、そのメンバー名から接頭辞または接尾辞が削除されます。接頭辞と接尾辞はアウトラインには表示されません。173 ページの「属性次元のメンバー名における、接頭辞および接尾辞のフォーマットの設定」を参照してください。
- 基本次元メンバーは多くの属性を持つことができますが、各属性次元の属性を 1 つしか持つことができません。

  たとえば、製品 100-10 では、サイズとパッケージの属性を持つことができますが、1 つのサイズと 1 つのパッケージ・タイプしか持つことができません。
- 属性は、暗黙的共有メンバー(その子が共有としてタグ付けされている)には関連付けられません。
- Essbase では、ハイブリッド分析対応のメンバーの属性はサポートされていません。

属性値は次のような比較の計算で使用できます:

- > (より大)
- >= (より大か等しい)
- < (より小)

- <= (より小か等しい)
- = = (等しい)
- <>または!= (等しくない)
- IN

属性次元の関連付けの規則の理解

属性次元を標準次元に関連付けると、その標準次元はその属性次元の基本次元になります。

- 属性次元は、疎の標準次元に関連付ける必要があります。
- 標準次元は、複数の属性次元の基本次元になることができます。
- 属性次元は、1 つの基本次元にのみ関連付けることができます。

  たとえば、S、M および L というメンバーを持つサイズ属性次元があるとします。サイズ属性次元を製品次元に関連付けた場合は、サイズ属性次元を市場次元に関連付けることはできません。市場次元のサイズ関連の情報を追跡する場合は、異なる名前を持つ別の属性次元(市場サイズなど)が必要になり、その市場サイズ属性次元を市場次元に関連付ける必要があります。

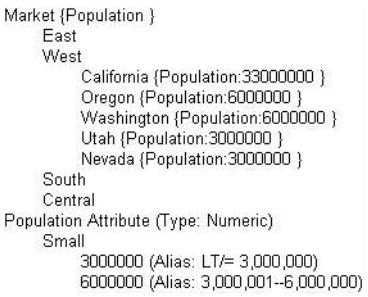
属性メンバーの関連付けの規則の理解

属性次元のメンバーを基本次元のメンバーに関連付ける場合は、次の規則に従います:

- 同じ属性次元の複数のメンバーを、同じ基本次元のメンバーに関連付けることはできません。たとえば、ボトルと缶のパッケージ・タイプを両方とも製品 100-30 に関連付けることはできません。
- 異なる属性次元のメンバーを基本次元の同じメンバーに関連付けることができます。たとえば、16 オンスのボトルで販売されるカフェイン抜きコーラの製品(100-30)は、「Caffeinated:False」、「Ounces:16」および「Pkg Type:Bottle」という 3 つの属性を持つことができます。
- 属性を基本次元のメンバーに関連付けた後に、基本次元メンバーを切り取るかコピーしてアウトライン内の別の場所に貼り付けると、属性の関連付けは失われます。
- Essbase では、それぞれの基本次元のメンバーを、属性次元のメンバーに関連付ける必要はありません。
- 特定の属性次元のメンバーに関連付けられたすべての基本次元メンバーは、同じレベルであることが必要です。

  たとえば、図 42 では、「Population」属性を持つすべての市場次元メンバーはレベル 0 にあります。「Population」属性を持つ市場次元の他のメンバーはレベル 0 のメンバーであるため、レベル 1 のメンバーである「East」を「Population」属性に関連付けることはできません。

図 42　市場次元の同一レベル・メンバーへの属性の関連付け



- 基本次元のメンバーに関連付けることができるのは、属性次元のレベル 0 にあるメンバーのみです。

  たとえば、「Population」属性次元では、「3000000」、「6000000」、「9000000」などのレベル 0 のメンバーのみを市場次元に関連付けることができます。「Small」などのレベル 1 のメンバーを関連付けることはできません。

  属性次元のレベル 0 のメンバー名が属性値になります。属性次元のメンバーのうち、属性値を持つメンバーのみがレベル 0 のメンバーになります。

  属性次元のより上位レベルのメンバーを使用して、データを選択したりグループ化したりできます。たとえば、「Population」属性次元のレベル 1 のメンバーである「Small」を使用して、「3000000」と「6000000」の両方の人口カテゴリの売上を取得できます。

属性タイプの理解

属性次元にはテキスト型、数値型、ブール型または日付型があり、この型によりデータのグループ化、選択または計算のための様々な関数を使用できます。属性タイプは次元レベルで割り当てられますが、属性次元のレベル 0 のメンバーのみに適用されます。

- デフォルトの属性タイプはテキストです。テキスト属性により、計算での基本属性メンバーの選択と属性の比較が可能になります。このような比較を実行すると、Essbase で文字が比較されます。たとえば、アルファベット順では B が C の前にくるため、パッケージ・タイプ「Bottle」はパッケージ・タイプ「Can」より小さくなります。Sample.Basic では、「Pkg Type」がテキスト属性次元の一例です。
- 数値属性次元のレベル 0 のメンバーの名前は、数値です。数値属性次元メンバーの名前(値)は計算に組み込むことができます。たとえば、「Ounces」属性で指定したオンス数を使用して、各製品のオンス当たりの収益を計算できます。

  数値属性は、基本次元の一定範囲の値に関連付けることができます。たとえば、市場の人口グループ別に製品の売上を分析するには、あるグループ内の人口 3,000,000 未満の州、別のグループ内の人口 3,000,001 から 6,000,000 までの州というように続けます。175 ページの「一定範囲の値を表すメンバー名の設定」を参照してください。
- データベース内のすべてのブール属性次元に含まれているメンバーは 2 つのみです。メンバー名はデータベースの設定(たとえば、True と False)と一致する必要があります。複数のブール属性次元が存在する場合は、接頭辞または

接尾辞のメンバー名フォーマットを指定して、メンバー名を確実に一意にします。たとえば、Caffeinated_True や Caffeinated_False です。174 ページの「ブール属性のメンバー名の設定」を参照してください。

- 日付属性を使用すると、日付フォーマット(月-日-年または日-月-年)を指定したり、それに応じて情報を順序付けたりできます。174 ページの「日付属性次元のメンバー名の変更」を参照してください。日付属性は計算で使用できます。たとえば、10-12-1999 に開設された市場から製品の売上を選択する計算で日付を比較できます。

  Essbase では、1970 年 1 月 1 日から 2038 年 1 月 1 日までの日付属性がサポートされています。

属性次元と標準次元の比較

一般に、属性次元とそのメンバーは、標準次元とそのメンバーに似ています。属性には別名やメンバーのコメントを付けることができます。属性次元には階層を含めることができ、世代とレベルに名前を付けることができます。属性次元とメンバーでは、標準次元とメンバーの場合と同じスプレッドシートの操作を実行できます。たとえば、様々な観点からデータを分析するために、スプレッドシート内で取得、ピボットおよびドリル・ダウンを実行できます。

表 18 で、属性次元と標準次元、およびそれらのメンバー間の主な相違点を説明します。

表 18　属性次元と標準次元との相違点

機能	属性次元	標準次元
ストレージ	疎。属性次元の基本次元も疎であることが必要です	密または疎が可能
ストレージ・プロパティ	動的計算のみが可能。したがって、データベース内には保管されません。アウトラインにはこのプロパティは表示されません。	データの保管、動的計算および保管、動的計算、非共有またはラベルのみが可能
アウトライン内の位置	アウトラインの最後の次元であることが必要	アウトライン内のすべての属性次元より上に位置することが必要
パーティション	属性次元に沿って定義することは不可。ただし、属性を使用して基本次元上にパーティションを定義することはできます	標準次元に沿って定義可能
式(メンバーに対する)	関連付け不可	関連付け可能
共有メンバー	不可	可能
2 パス計算メンバー・プロパティ	使用不可	使用可

機能	属性次元	標準次元
実行時の式での 2 パス計算	メンバー式に、属性メンバー名に関連付けられている実行時依存の関数が含まれていて、その式を持つメンバーに 2 パスとタグ付けされている場合、計算ではそのメンバーがスキップされ、警告メッセージが表示されます。実行時依存の関数には、@CURRMBR、@PARENT、@PARENTVAL、@SPARENTVAL、@MDPARENTVAL、@ANCEST、@ANCESTVAL、@SANCESTVAL および @MDANCESTVAL があります。 169 ページの「属性次元での 2 パス計算の理解」を参照してください。	計算は、実行時の式を持ち、2 パスのタグが付けられている標準メンバーに対して実行されます。
2 パス、複数次元: 計算順序	2 パスとタグ付けされたメンバーの計算の順序は、アウトライン内での順序に従います。最後の次元は最後に計算されます。	複数の次元で 2 パスのタグが付けられているメンバーの計算結果は、アウトライン順序には従いません。
メンバー式を含まない 2 パス計算	計算はスキップされ、警告メッセージが表示されます。したがって、2 パスとタグ付けされたメンバーと上位レベルのメンバーのメンバー交差では、標準次元での計算とは異なる結果が戻されることがあります。 169 ページの「属性次元での 2 パス計算の理解」を参照してください。	使用可
存在しない保管ブロックにおける、密の動的計算メンバー	存在しない保管ブロック上の密次元は、計算ではスキップされます。 密メンバーで機能する属性に対して、密メンバーのデータ・ブロックが存在する必要があります。動的計算式があり属性のない密メンバーでデータを取得する際、Essbase は動的にデータ・ブロックを作成して値を戻します。ただし、動的計算密メンバーに属性がある場合、属性メンバーで取得を実行すると #MISSING という結果が生じます。これは、Essbase が密メンバーの動的計算をスキップし、データ・ブロックが作成されないためです。 存在しない保管ブロックを特定するには、データベースをエクスポートするかクエリーを実行して、ブロックにデータが含まれるかどうかを調べます。	使用可
メンバーに対する UDA	不可	可能
集計	すべてのメンバーが、属性計算次元メンバー(Sum、Count、Min、Max および Avg)で計算されます。属性計算中はアウトライン内の集計演算子は無視されます。	希望する集計記号をそれぞれのメンバーに割り当てることによって、集計操作が指示されます
レベル 0 メンバーの型によるメンバー選択	使用可能な型はテキスト、数値、ブールおよび日付です。	すべてのメンバーはテキストとして処理されます
関連付け	基本次元に関連付けられることが必要	該当なし
スプレッドシートのドリルダウン	選択した属性に関連付けられた基本次元データが表示されます。たとえば、属性「Glass」をドリル・ダウンすると、パッケージがガラス瓶の各製品の売上が表示されます。その場合、「Product」は「Pkg Type」属性次元の基本次元です。	標準次元内の下位または兄弟のレベルの詳細が表示されます。たとえば、「QTR1」をドリル・ダウンすると、その四半期の製品とそれらの売上のリストが表示されます。

属性次元での 2 パス計算の理解

次の例は、Sample.Basic データベースの Product 次元に基づき、2 パス計算が属性次元でどのように機能するかを示しています。メンバー"400 - 30"が 2 パスとしてタグ付けされていると仮定します。

メンバー"400 - 30"に次のメンバー式がある場合:

```
=”400-10”;
```

Essbase は"400 - 30"における取得の実行時に、式を実行します。

"400 - 30"に次のメンバー式がある場合:

```
=@CURRMBR("Market");
```

Essbase は、許可されていない@CURRMBR ランタイム関数が式に含まれるため、計算をスキップして次のエラー・メッセージを発行します:

```
Two-pass calc skipped on member [400-30] in attribute calc”
```

"400 - 30"にメンバー式がない場合、2 パスとしてタグ付けされているメンバーには式が必要なため、同じエラー・メッセージが生成されます。

属性と UDA の比較

属性と UDA により、データの特性に基づく分析が可能になります。属性には UDA よりも多くの機能があります。このトピックの表で、次の機能の領域における属性と UDA の相違点を示します:

- データ・ストレージ(表 19)
- データの取得(表 20)
- データ変換(表 21)
- 計算スクリプト(表 22)

表 19　データ・ストレージ- 属性と UDA の比較

データ・ストレージ	属性	UDA
疎次元に関連付けることができます。	サポートされています	サポートされています
密次元に関連付けることができます。	サポートされていません	サポートされています

表 20 データの取得- 属性と UDA の比較

データの取得	属性	UDA
属性値または UDA 値によって集計された合計をグループ化および取得できます。たとえば、値「High Focus Item」を製品次元の様々なメンバーに関連付け、この値を使用してこれらのメンバーのみの合計や詳細を取得します。	サポートされています 簡単	サポートされています 導入が困難。計算スクリプトやコマンドを追加することが必要
属性を階層で分類して、階層の上位レベルにおける集計結果を取得できます。たとえば個々の製品に 8、12、16 または 32 のようなサイズ属性があり、これらのサイズが S、M および L に分類されている場合などです。S の製品の売上合計を表示できます。	サポートされています	サポートされています 導入が困難
同じ基本次元に関連付けられた属性の集約合計を表示する、クロス集約表示を作成できます。	サポートされています 各属性次元のすべての値に対するクロス集計を表示できます。	サポートされていません 特定の UDA 値に基づく合計のみを取得できます。
ブール演算子 AND、OR および NOT を属性値と UDA 値とともに使用して、クエリーの精度を上げることができます。たとえば、製品グループ 100 からカフェイン抜き飲料を選択できます。	サポートされています	サポートされています
属性はテキスト、ブール、日付または数値の型を持つため、適切な演算子や関数を使用して属性データを操作したり表示したりできます。たとえば、特定の日付以降に発売されたすべての製品の売上合計を表示できます。	サポートされています	サポートされていません
数値属性を値の範囲でグループ化し、次元構築プロセスによって基本メンバーを適切な範囲に自動的に関連付けることができます。たとえば、売上を人口の範囲(300 万未満、300 万以上 600 万未満など)に基づいた様々な地域にグループ化できます。	サポートされています	サポートされていません
属性計算次元を使用して、合計、カウント、最小、最大、平均などの属性値の集約を表示できます。	サポートされています	サポートされていません
メンバーを定義する計算で属性を使用できます。たとえば、オンス単位の製品の重さを使用して、メジャー次元のオンス・メンバー当たりの収益を定義できます。	サポートされています	サポートされていません
属性関連情報を使用して特定の基本メンバーを取得できます。	サポートされています 強力な条件選択と値ベースの選択	サポートされています テキスト文字列の一致のみに制限

表 21 データ変換- 属性と UDA の比較

データ変換	属性	UDA
データがデータベースにロードされるときに、UDA の値に基づいてデータの符号を変更できます。たとえば、「借方」という UDA を持つすべてのメンバーの符号を反転させることができます。	サポートされていません	サポートされています

表 22　計算スクリプト- 属性と UDA の比較

計算スクリプト	属性	UDA
メンバーの属性値または UDA 値が特定の値に一致する場合に、そのメンバーに対して計算を実行できます。たとえば、「Bottle」の属性または UDA で、すべての製品の価格を 10%上げることができます。	サポートされています	サポートされています
指定した条件を満たす属性値を持つ基本メンバーに対して、計算を実行できます。たとえば、各基本メンバーのオンス当たりの収益を計算できます。	サポートされています	サポートされていません

属性次元の設計

Essbase には、データベースで属性情報を設計する方法が複数あります。ほとんどの場合、属性次元とそのメンバーを使用してデータの特性を定義するのが最良の方法です。以降のセクションで、属性次元を使用するタイミング、他の機能を使用するタイミング、および属性を使用するときにパフォーマンスを最適化する方法について説明します。

属性次元の使用

最高の柔軟性と機能性を得るために、属性次元を使用して属性データを定義します。属性次元を使用すると、次の機能が得られます:

- 上質で柔軟なデータ取得

  いつでも必要なときに属性データを表示したり、クロス集計を使用して有用な要約を作成したり、タイプに基づく比較を使用して参照するデータのみを選択して表示したりすることなどができます。

- 計算機能の追加

  属性次元のメンバー名に計算を実行して、標準次元のメンバーを定義できるのみでなく、5 つのタイプの属性データの連結(合計、カウント、平均、最小および最大)にアクセスできます。

- 経済性と単純性

  属性次元は疎の動的計算であるため、データとしては保管されません。共有メンバーの使用と比較して、属性次元を使用するアウトラインに含まれるメンバーは少ないため、読取りが簡単です。

162 ページの「属性の理解」を参照してください。

設計の代替アプローチの使用

状況に応じて、次のアプローチのいずれかを検討します:

- UDA。UDA は属性に比べ柔軟性に欠けますが、データの特性に基づいたデータのグループ化や取得ができます。169 ページの「属性と UDA の比較」を参照してください。
- 共有メンバー。たとえば、季節分析を年次元に含めるには、適切な季節の下で共有メンバーとして月を繰り返し指定します。Winter: Jan (共有メンバー)、

Feb (共有メンバー)のように続きます。共有メンバーを使用する場合の主な短所は、カテゴリで多数のメンバーを繰り返し指定すると、アウトラインが大きくなることです。

- 標準の次元とメンバー。標準次元の追加によって柔軟性が得られますが、データベースにストレージの要件と複雑さが加わります。次元の追加による影響を評価する際のガイドラインは、77 ページの「分析とプランニング」を参照してください。

表 23 で、データベース内の属性データの管理に対する代替アプローチを検討する状況について説明します。

表 23　属性次元への代替アプローチの検討

状況	代替アプローチ
密次元の属性を分析する	UDA または共有メンバー
データのバッチ計算を実行する	共有メンバーまたは別個の標準次元のメンバー
属性次元のメンバーの名前を、式の結果である値として定義する	共有メンバーまたは別個の標準次元のメンバー
時間によって変化する属性を定義する	別個の標準次元のメンバー。たとえば、製品のメンテナンス・コストを一定期間追跡するには、メンテナンス時点の製品の寿命が重要です。ただし、属性機能を使用すると、製品に 1 つの寿命を関連付けるだけで済みます。追跡する各期間の個々の次元内に複数のメンバーが必要です。
多数の基本次元メンバーの取得にかかる時間を短縮する	共有メンバーまたは別個の標準次元のメンバーを使用したバッチ計算。

アウトラインのパフォーマンスの最適化

アウトラインのレイアウトとコンテンツは、属性計算とクエリーのパフォーマンスに影響する可能性があります。一般的なアウトラインの設計に関するガイドラインは、91 ページの「パフォーマンスを最適化するためのアウトラインの設計」を参照してください。

属性のクエリーのパフォーマンスを最適化するには、次の設計ヒントを検討してください:

- 属性次元がアウトライン内で唯一の疎の動的計算次元であることを確認します。
- アウトライン内で密次元の後に疎次元を配置します。最も頻繁にクエリーの対象になる次元を疎次元の最初に置いて、属性次元をアウトラインの最後に置きます。ほとんどの場合、基本次元が最も頻繁にクエリーの対象になる次元です。

179 ページの「計算と取得のパフォーマンスの最適化」を参照してください。

属性次元の構築

属性次元を構築するには、次元に属性としてタグ付けし、その次元にタイプを割り当てます。次に、属性次元を基本次元に関連付けます。最後に、属性次元のレベル 0 の各メンバーを、関連する基本次元のメンバーに関連付けます。

➤ 属性次元を構築する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「属性の定義」を参照してください。

➤ 特定の属性メンバーの次元、属性値および属性タイプを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アウトラインでの属性情報の表示	Oracle Essbase Administration Services Online Help
MaxL	**query database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	GETATTRINFO	『Oracle Essbase テクニカル・リファレンス』

属性次元におけるメンバー名の設定

属性機能を使用する場合、Essbase ではデフォルトのメンバー名が設定されます。たとえば、システム定義の True と False により、True と False の他のメンバー名は排除されます。データベースのこれらのシステム定義の名前は変更可能です。日付属性と数値属性も重複可能です。重複名が混同されないようにするため、属性次元内のメンバー名を修飾するための設定を行うことができます。完全修飾属性名はアウトラインには表示されませんが、パーティションを定義する場合や、取得する情報を選択する場合など、メンバーを選択する場所ではどこでも完全な属性名を表示できます。

属性次元を定義または構築する前に、メンバー名の設定を定義します。属性次元とメンバーを定義した後でこの設定を変更すると、メンバー名が無効になることがあります。

次の項では、属性次元のメンバー名の操作方法について説明します:

注： 属性次元を含むアウトライン上でパーティション化を行う場合、この項で説明するメンバーの名前フォーマット設定は、ソースとターゲットのアウトラインで等しくなることが必要です。

属性次元のメンバー名における、接頭辞および接尾辞のフォーマットの設定

この項で説明する情報は、重複メンバー属性次元には適用されません。

ブール、日付および数値属性次元のメンバーの名前は、値です。異なる属性次元内に重複する属性値を存在させることができます。

- ブールの例

アウトライン内に複数のブール属性次元がある場合、これらの各次元の 2 つのメンバーは同じ名前(デフォルトで True および False)を持っています。

- 日付の例

  複数の日付属性次元がある場合、両方の次元内のいくつかのメンバー名は同じになることがあります。たとえば、ある市場で店舗が開店する日付が、製品を発売した日付と同じになることがあります。

- 数値の例

  製品のサイズの属性値が 12 であり、製品のパッケージ単位数の値も 12 であることもあります。この例では、12 という名前を持つメンバーが 2 つになります。

アウトライン内のブール、日付および数値属性次元のメンバー名に接頭辞または接尾辞を付けることで、一意の名前を定義できます。属性名に次元、親、親の親またはすべての祖先を添付できます。たとえば、属性次元のメンバー名を設定する際、接尾辞として次元名を含め、アンダースコアを付けることによって、オンス属性次元のメンバー値 12 は一意の完全な属性メンバー名「12_Ounces」とみなされます。

デフォルトでは、Essbase では、属性次元のメンバー名に接頭辞や接尾辞を付けないことが前提になります。

選択するルールは、アウトライン内のすべての数値、ブールおよび日付属性次元のレベル 0 のメンバー名に適用されます。取得の際に短い名前を表示する場合は、これらの名前の別名を定義できます。

➤ 接頭辞および接尾辞のフォーマットを定義する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「属性次元のメンバーの接頭辞または接尾辞のフォーマットの定義」を参照してください。

ブール属性のメンバー名の設定

属性次元の次元タイプをブールとして設定すると、Essbase ではブール属性設定に指定された名前を持つレベル 0 のメンバーが 2 つ自動的に作成されます。データベース内の最初のブール・メンバー名は、True および False として設定されます。これらのデフォルト名を、たとえば Yes と No に変更するには、データベース内にブール属性次元を作成する前に、ブール属性次元のメンバー名を定義します。

属性次元タイプをブールに設定する場合は、次元内の既存のすべてのメンバーを削除する必要があります。

➤ ブール属性次元のメンバーの名前のデータベース設定を定義する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「ブール属性次元のメンバー名の設定」を参照してください。

日付属性次元のメンバー名の変更

日付属性次元のメンバーのフォーマットは、変更可能です。たとえば、次のような日付フォーマットを使用できます:

- mm-dd-yyyy: 2007 年 10 月 18 日は 10-18-2007 と表示されます。
- dd-mm-yyyy: 2007 年 10 月 18 日は 18-10-2007 と表示されます。

日付メンバー名フォーマットを変更すると、日付属性次元の既存のメンバーの名前が無効になることがあります。たとえば、10-18-2007 メンバーが存在しているときに、フォーマットを dd-mm-2007 に変更すると、アウトライン確認ではこのメンバーは無効とみなされます。日付フォーマットを変更する場合は、日付属性次元を再構築する必要があります。

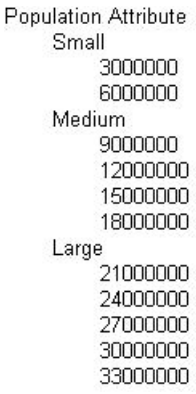
➤ 日付属性次元のメンバー名を変更する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「日付属性次元のメンバー名のフォーマットの設定」を参照してください。

一定範囲の値を表すメンバー名の設定

数値属性次元のメンバーは、単一の数値または一定の範囲内の値を表せます:

- 単一の値の例: オンス属性次元のメンバー 12。これは単一の数値 12 を表します。この属性をすべての 12 オンスの製品に関連付けます。アウトラインには、サイズごとに別のメンバー(たとえば、16、20 および 32)が含まれます。
- 一定の範囲内の値の例: 図 43 に示すような「Population」属性次元:

図 43 「Population」属性次元とメンバー



このアウトラインでは、「Population」属性次元のメンバーは、関連する市場次元内の人口値の範囲を表しています。3000000 メンバーはゼロから 3,000,000 まで、6000000 メンバーは 3,000,001 から 6,000,000 までのように表しています。アウトラインの設定では、各数値メンバーがその範囲の上限を表すように設定されます。

このアウトラインの設定は、数値属性次元のメンバーが、それらが表現する範囲の下限になるように定義できます。たとえば、範囲の下限を定義するように数値メンバーを設定する場合、3000000 メンバーは 3,000,000 から 5,999,999 までの人口を表し、6000000 メンバーは 6,000,000 から 8,999,999 までの人口を表します。

基本次元を構築すると、Essbase によって、基本次元のメンバーが該当する属性範囲に自動的に関連付けられます。たとえば、数値メンバーが範囲の上限を表して

いる場合、Essbase により、人口が 3,269,858 の「Connecticut」市場は、「Population」属性次元の 6000000 メンバーに関連付けられます。

次元構築ルール・ファイルでは、数値属性次元の各メンバーに範囲のサイズを指定します。上の例では、各属性は 3,000,000 の範囲を表しています。

➤ 数値属性次元で範囲を定義する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「値の範囲へのメンバー名の割当て」を参照してください。

注： 数値属性次元のメンバー名の小数点以下の桁数は、6 桁を超えないようにすることをお薦めします。6 桁を超えると、精度調整が原因でアウトラインの確認に成功しないことがあります。

属性計算次元のメンバー名の変更

アウトラインにおける名前の重複を避けるために、属性計算次元またはそのメンバーの名前の変更が必要になる場合があります。

Sum、Count、Min、Max および Avg の、Essbase が属性計算次元で作成するメンバーの名前は、予約語と想定されません。属性計算次元これらの名前を変更してから、属性次元または標準次元のデフォルト名を使用できるためです。次のガイドラインに従います:

- アウトラインが重複メンバー・アウトラインとしてタグ付けされている場合、デフォルト名を使用して他の基本メンバーまたは属性メンバーに名前を付けます。
- アウトラインが一意メンバー・アウトラインとしてタグ付けされている場合、Sum、Count、Min、Max および Avg をメンバー名として使用することは避けます。たとえば、標準次元で Max を使用してから属性次元を作成する場合、Essbase が属性計算次元で Max メンバーを作成すると、Essbase は重複した名前を検知して、名前がすでに使用中であることを示すエラー・メッセージを戻します。

属性計算次元メンバーに使用する名前とは無関係に、その機能はそのまま持続します。たとえば、2 番目の Count メンバーは常に数を数えます。

➤ 属性計算次元のメンバー名を変更する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「属性計算次元のメンバー名の変更」を参照してください。

属性データの計算

Essbase では、Essbase によって作成されたシステム定義次元のメンバーを使用して、取得時に属性データが動的に計算されます。この次元を使用して、合計や平均などの様々な計算関数を同じ属性に適用できます。属性次元のメンバーに対して特定の計算を実行することもできます。たとえば、サイズがオンスで示された製品についてオンス当たりの収益性を判断できます。

以降の説明では、属性次元と Essbase の計算(動的計算など)の概念を理解していることが前提になります。次の項を参照してください。

属性計算次元の理解

アウトライン内に最初の属性次元を作成すると、Essbase によって、デフォルト名 Sum、Count、Min (最小)、Max (最大)および Avg (平均)を持つ 5 つのメンバーで構成される属性計算次元も作成されます。これらのメンバーをスプレッドシートやレポートで使用して、西部地域の 12 オンス・ボトル入りコーラの 1 年間の平均売上などの属性データを動的に計算したり、レポートを作成したりできます。

属性計算次元はアウトラインには表示されません。属性計算次元は、Smart View など、次元メンバーを選択した任意の場所で表示できます。

属性計算次元には次のプロパティがあります:

- システム定義

  アプリケーションで最初の属性次元を作成すると、Essbase によって、属性計算次元とそのメンバー(Sum、Count、Min、Max および Avg)が作成されます。各メンバーは、属性で実行される計算のタイプを表します。

  178 ページの「デフォルトの属性計算メンバーの理解」を参照してください。

- ラベルのみ

  すべてのラベルのみ次元と同様、属性計算次元はその最初の子 Sum の値を共有します。

  90 ページの「メンバーのストレージ・プロパティ」を参照してください。

- 動的計算

  属性計算次元内のデータは、ユーザーが要求したときに計算され、その後は破棄されます。計算された属性データは、データベースには保管できません。

  第 27 章「データ値の動的計算」を参照してください。

- アウトライン・エディタでは非表示

  属性計算次元はアウトライン・エディタでは表示されません。この次元のメンバーは、スプレッドシートとレポートで表示できます。

属性次元に沿った集計はありません。属性データを計算するために、属性次元のメンバーに集計記号(+や-)またはメンバー式をタグ付けすることはできません。属性計算は、時間や計算順序について、動的計算メンバーとしてバッチ計算に影響を与えることはありません。

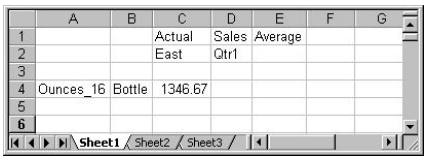
取得時に属性データを計算するため、Essbase では次のタスクが実行されます:

1. 現在のクエリーに存在する属性次元メンバーに関連付けられた基本次元メンバーを探します
2. 現在のクエリーの属性とメンバーの組合せに対して、合計、カウント、最小、最大または平均を動的に計算します
3. スプレッドシートまたはレポートに結果を表示します
4. 計算された値を破棄します。この値はデータベースに保管されません

注： Essbase では、属性データを計算するときは#MISSING 値が除外されます。

たとえば、図 44 に示すように、スプレッドシートのユーザーは、属性次元の 2 つのメンバー(「Ounces_16」と「Bottle」)と属性計算メンバー(Avg)をスプレッドシートのレポートに指定します。取得時に、Essbase では、現在のメンバーの組合せ(Actual -> Sales -> East -> Qtr1)に対してこれらの属性に関連付けられたすべての製品の平均売上高が動的に計算されます:

図 44　属性計算メンバーの取得



179 ページの「スプレッドシートの使用による属性計算メンバーへのアクセス」を参照してください。

デフォルトの属性計算メンバーの理解

属性計算次元には、属性データを計算したりレポートしたりするために使用する 5 種類のメンバー(Sum、Count、Min、Max および Avg)があります。

- Sum - 属性または属性の組合せを持つメンバーの合計を計算します。
- Count - 指定された属性または属性の組合せを持ち、それに対するデータ値が存在するメンバーの数を計算します。「Count」には、存在するデータ・ブロックを持つメンバーのみが含まれます。データ値を持つかどうかにかかわらず、ある属性を持つすべてのメンバーの数を計算するには、@COUNT 関数を @ATTRIBUTE 関数と組み合せて使用します。『Oracle Essbase テクニカル・リファレンス』を参照してください。
- Avg - 指定された属性または属性の組合せに対して、欠落していない値の数学的平均を計算します(Sum を Count で除算)。
- Min - 指定された属性または属性の組合せに対して最小データ値を計算します。
- Max - 指定された属性または属性の組合せに対して最大データ値を計算します。

注： これらの各計算では#MISSING 値は除外されます。

これらのデフォルトのメンバー名は、標準のメンバーと同じ命名規則に従って変更できます。176 ページの「属性計算次元のメンバー名の変更」を参照してください。

属性計算の例

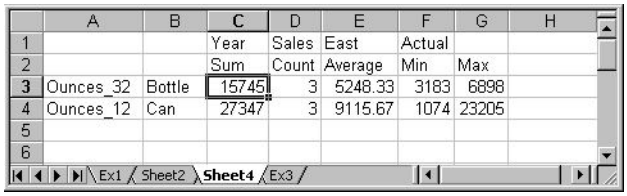
Essbase による属性データの計算例として、次に示す東部の年次売上データを検討します:

表 24　サンプル属性データ

基本次元メンバー	関連属性	属性とメンバーの組合せに対する売上高
Cola	Ounces_12, Can	23205
Diet Cola	Ounces_12, Can	3068
Diet Cream	Ounces_12, Can	1074
Grape	Ounces_32, Bottle	6398
Orange	Ounces_32, Bottle	3183
Strawberry	Ounces_32, Bottle	5664

図 45 は、計算された属性データがスプレッドシートのレポートでどのように表示されるかを示しています。属性に対して複数の属性計算メンバーを取得できます。たとえば、ボトルと缶に対して Sum、Count、Avg、Min および Max を計算できます。

図 45　属性データによるサンプル・スプレッドシート



スプレッドシートの使用による属性計算メンバーへのアクセス

Smart View で属性計算次元のメンバーにアクセスできます。スプレッドシートでは、ユーザーは次の方法のいずれかを使用して、属性計算次元メンバーを表示できます:

- シートにメンバーを直接入力します
- クエリー・デザイナからメンバーを選択します
- EssCell パラメータでメンバーを入力します

Oracle Hyperion Smart View for Office User's Guide を参照してください。

計算と取得のパフォーマンスの最適化

属性計算と取得のパフォーマンスを最適化するには、次の点を考慮してください:

- 属性計算の計算順序は、動的計算の場合と同じです。アウトラインについては、451 ページの「動的計算の計算順序」を参照してください。
- Essbase では、取得時に属性データが動的に計算されるため、属性計算は全体の(バッチ)データベース計算のパフォーマンスには影響しません。
- 基本次元メンバーに動的計算タグを付けると取得時間が短縮されます。
- クエリーに Sum メンバーと、関連する基本メンバーに 2 パス・タグが付けられた属性次元メンバーが含まれている場合、取得時間が遅くなることがあります。
- 属性の取得パフォーマンスを最大化するには、次のいずれかの方法を使用します:
  - 172 ページの「アウトラインのパフォーマンスの最適化」のヒントを使用してアウトラインを構成します。
  - データの取得前に、基本次元の最下位レベルまでドリル・ダウンします。たとえば、Smart View では、データなし操作機能を有効にして、レポートに含まれている基本次元の最下位レベルまでドリル・ダウンし、その後データを取得します。
  - 基本次元のメンバーがいくつかの属性次元に関連付けられている場合は、それらの属性に従って基本次元メンバーをグループ化することを検討します。たとえば、Sample.Basic データベースでは、すべての 8 オンスの製品をグループ化できます。

計算式での属性の使用

属性計算次元を使用して属性データを計算することに加え、標準次元または基本次元のメンバーに計算式を使用して、属性次元のメンバーに対して特定の計算を実行できます。たとえば、サイズがオンスで示された製品についてオンス当たりの収益性を判断できます。

式を属性次元のメンバーに関連付けることはできません。

注： 2 パス・メンバーに関連付けられた式で属性を使用する場合は、いくつかの制限が適用されます。表 18 および 169 ページの「属性次元での 2 パス計算の理解」で、2 パス計算に関する行を参照してください。

表 25 で、属性に対する特定の計算の実行に使用できる関数を示します:

表 25　属性に対する計算を行う関数

関数	計算のタイプ
@ATTRIBUTE	特定の属性を持つすべての基本メンバーのリストを生成します。たとえば、Bottle 属性を持つメンバーのリストを生成してから、これらのメンバーの価格を上げます。

関数	計算のタイプ
@ATTRIBUTEVAL @ATTRIBUTEBVAL @ATTRIBUTESVAL	計算される基本メンバーに関連付けられているレベル 0 属性メンバーの値を戻します。 ● 数値または日付属性次元から(@ATTRIBUTEVAL を使用) ● ブール属性次元から(@ATTRIBUTEBVAL を使用) ● テキスト属性次元から(@ATTRIBUTESVAL を使用) たとえば、計算対象の基本メンバー(たとえば、「Cola」)に対して、サイズ属性の数値(たとえば、「Ounces」の下のメンバー 12 に対して 12)を戻します。 式での@ATTRIBUTEVAL のその他の使用例は、405 ページの「属性の式の計算」を参照してください。
@TODATE	日付文字列を計算のために数値に変換します。たとえば、@TODATE を@ATTRIBUTEVAL 関数と組み合せて使用して、ある日付以降に開店した店舗の間接費を増額します。
@WITHATTR	指定した条件を満たす属性または可変属性に関連付けられている基本次元メンバーのリストを生成します。たとえば、20 オンス以上の製品のリストを生成してから、それらの製品の価格を上げます。

属性計算と共有メンバーの理解

属性計算は、レベル 0 から開始され、最初の保管済メンバーで停止します。したがって、アウトラインでアウトライン階層内の 2 つの共有メンバーの間に保管済メンバーが配置されていると、より上位の共有メンバーは計算結果に含まれないことがあります。

次の例では、属性計算が実行されると、計算はレベル 0 のメンバー 2 から開始され、最初の保管済メンバーであるメンバー A が検出された時点で停止します。したがって、メンバー 1 は計算には含まれません。

```
    Member 1 (stored)
  Member A (stored)
   Member 2 (shared)
Member B (stored)
  Member 1 (shared member whose stored member is Member 1 above)
```

属性計算での予期せぬ結果を回避するために、共有メンバーと保管済メンバーを混在させないでください。この例の場合、メンバー 2 が共有されていない場合、またはメンバー 1 にアウトライン内の他の場所に対応する共有メンバーがない場合、計算結果は予期したとおりになります。

可変属性

可変属性について

通常、製品はその製品を記述または定義する属性を持ちます。たとえば、ある製品が、製品サイズをオンスで記述する属性と、製品の種類を記述する属性を持つ

とします。このようなシナリオでは、製品は基本次元で、オンスと種類は属性次元です。

可変属性を使用すると、独立次元という 3 番目の次元との関連で 2 つの値を追跡できます。たとえば、8 オンスの製品を 1 年間追跡するとします。このシナリオでは、時間が独立次元です。この 3 番目の係数の値は可変です(そのため「可変属性」という名前です)。たとえば、製品を 1 年間、1 四半期または 1 か月間追跡できます。

注： 独立次元には、連続と個別の 2 つのタイプがあります。連続次元のメンバーは連続を示します。たとえば、週、月および四半期は、時間次元内の連続を示します。個別次元のメンバーは、連続の意味を持ちません。たとえば、市場次元のカリフォルニア、テキサスおよびオハイオには、連続に基づいた関係はありません。

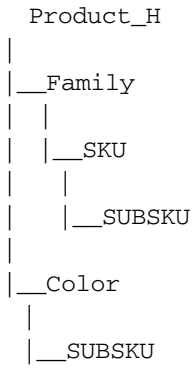
もう 1 つの例として、次のシナリオについて考えてみます: あるクライアントの営業担当者が 1 年の中ごろに変わったとします。表 26 に、半年間の顧客の売上合計と営業担当者の割当てを示します:

表 26　可変属性の例: ある期間での営業担当者の変更

3 月	4 月	5 月	6 月	7 月	8 月
4000	6000	2000	1000	1000	7000
ジョーンズ	ジョーンズ	ジョーンズ	スミス	スミス	スミス

この例では、営業担当者が可変属性です。データの取得により、営業担当者であるジョーンズが 3 月から 5 月までに総額$12,000 の製品を顧客に販売し、その後、営業担当者であるスミスが 6 月から 8 月までに総額$9,000 の製品を販売したことが示されています。可変属性を使用しない場合、営業担当者については現在の担当者であるスミスしか把握できず、すべての売上($21,000)がスミスの売上として示されます。

可変属性は、メンバーのグループ化に代わる方法を提供します。たとえば、色を使用して SKU をグループ化できるとします。このシナリオでは、属性次元「Color」は SUBSKU に関連付けられています:



Color を可変属性に設定すると、取得の結果は表 27 の値と同様になります:

表 27　可変属性としての Color

SUBSKU	SKU
赤	100
白	400
白	600
黒	200
黒	300
銀	500

可変属性は次のガイドラインに従う必要があります:

- 複数のチェーンを持つ必要がある。
- リーフ・レベルが一致する必要がある。

アウトラインでは可変属性をサポートできます。属性次元を可変属性として機能するように定義できます。また、必要な情報のタイプを反映するように可変属性を編集することもできます。

可変属性の実装

可変属性は、集約ストレージ・データベースとブロック・ストレージ・データベースでサポートされます。可変属性はデータベース・レベルで実装します。

可変属性を使用可能にして使用するには、次のワークフローを使用します:

1. アウトライン・プロパティで、可変属性を使用可能にします。
2. 基本次元のメンバー・プロパティで、「属性」タブに移動し、独立次元(可変属性が依存する次元)を識別します。

   たとえば、顧客 A の営業担当者の属性の関連付けが 5 月に変更される場合は、年が独立次元になります。
3. 独立次元のタイプ(連続または個別)を指定します。連続独立次元の一例は、時間に基づいた独立次元です。個別独立次元には連続性はありません。たとえば、市場次元では、カリフォルニア、テキサスおよびオハイオには連続に基づいた関係はありません。
4. 独立次元を可変属性に関連付けます。オプションで、範囲と関連付けモードを選択します。

   属性の関連付けが当てはまる範囲を割り当てることができます。たとえば、「ジェーンが 2007 年 7 月から 2008 年 6 月までエンジニアである」のように属性の関連付けが適用される時間範囲を割り当てることができます。

   関連付けモードは、Essbase での可変属性とその独立次元との関連付けの競合を処理する方法を指定します。使用できる関連付けモードは、「Overwrite」、「NoOverwrite」および「Extend」です。
5. アウトラインを保存し、再構築します。

6. 必要に応じて、次のメンテナンス作業を実行します:
    - 独立メンバーに新しい可変属性の関連付けを追加します(たとえば、従業員の新しい役職を追加します)。
    - 独立メンバーの関連付けを削除します。
    - 既存の独立次元のメンバーの関連付けを表示します(たとえば、会社に代理の販売管理者がいた月を表示します)。
    - 基本次元から属性次元の関連付けを解除します。

可変属性をサポートする関数

次に示すレポート・ライター・コマンドは可変属性を使用できるように設計されています。

- <PERSPECTIVE
- <WITHATTREX
- <ATTRIBUTEVA

次に示す計算関数およびコマンドには可変属性を使用できます。

- @WITHATTR
- @ISATTRIBUTE
- @ISMBRWITHATTR
- SET SCAPERSPECTIVE

次に示す MDX 関数は可変属性を使用できるように設計されています。

- AttributeEx
- WithAttrEx
- WITH PERSPECTIVE キーワード

『Oracle Essbase テクニカル・リファレンス』を参照してください。

可変属性の制限

連続独立次元は、単一の次元(年や月など)として機能する必要があります。関連のない連続独立次元はサポートされていません。

連続独立次元およびメンバーは、最後に指定する必要があります。

独立メンバーは、保管されているレベル 0 のメンバーにする必要があります。

計算スクリプトにパースペクティブが指定されていない場合は、FIX コマンドに可変属性を含めることはできません。

範囲の開始/終了を示す独立メンバーの場合:

- 削除できません
- 共有すること、またはラベルのみにすることはできません

- レベル 0 にする必要があります
- 別の次元に移動することはできません
- 他の範囲に移動することはできません

アウトラインが保存されるまでは、新たに移動または追加された独立メンバーを使用して範囲を示すことはできません。

11 Essbaseデータへのオブジェクトのリンク

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

LRO の理解

リンク・レポート・オブジェクト(LRO)を使用して、様々な種類のデータを Essbase データベース内のセルとリンクできます。この機能は、電子メールにファイルを添付することに似ています。LRO により、プランニング・アプリケーションとレポート・アプリケーションに対するサポートが改善され、データ分析機能を向上できます。

LRO は、Essbase データベース内の特定のデータ・セルに関連付けるオブジェクト(アーチファクト)です。ユーザーは、Smart View を使用して、リンク・オブジェクトを作成、取得および編集します。セルにリンクできるオブジェクトの数は無制限です。オブジェクトは Essbase サーバーに保管され、Essbase サーバーでは、適切なアクセス権を持つユーザーがそのオブジェクトを使用できます。表 28 で説明しているリンク・オブジェクトのタイプの最大サイズは、付録 A「制限」を参照してください。

表 28　リンク・オブジェクトのタイプ

オブジェクト・タイプ	説明
セル・ノート	テキスト注釈
ファイル	Microsoft Word 文書、Excel スプレッドシート、スキャン・イメージ、オーディオ・クリップおよび HTML ファイル(たとえば `mypage.htm`)などの外部ファイル。

オブジェクト・タイプ	説明

URL	例:
	
	`http://www.oracle.com`
	
	`ftp://ftp.oracle.com`

リンク・パーティション	別の Essbase データベースにリンク可能なデータ・セルのセット。

たとえば、販売管理者が最近更新した予算アイテムにセル・ノートを添付するとします。財務管理者は、この四半期の成績の補足データを含むスプレッドシートをリンクします。製品管理者は、新製品のビットマップ・イメージをリンクします。販売管理者は、Web サイトの情報にすぐにアクセスできるように、会社の Web サイトの URL をリンクします。

LRO のタイプとデータ・セルの理解

LRO はデータ・セルにリンクしており、セル内に含まれているデータにリンクしているわけではありません。このリンクは、データベース内のメンバーの組合せに基づきます。セルに対するリンクの追加や削除は、セルのコンテンツには影響しません。

ユーザーがセルにオブジェクトをリンクすると、Essbase によって、オブジェクトのタイプ、最後にオブジェクトを修正したユーザーの名前、オブジェクトの最終更新日などの情報がオブジェクト・カタログに保管されます。

Essbase での LRO の保管方法は、LRO のタイプによって異なります:

- オブジェクトがセル・ノートである場合、セル・ノートのテキストがオブジェクトの記述のパートとしてカタログ項目に保管されます。
- オブジェクトがファイルである場合、Essbase によって、そのファイルのコンテンツは Essbase サーバーのデータベース・ディレクトリに保管され、そのファイルには`.lro` 拡張子が付けられます。Essbase では、リンク・ファイルのデータ・フォーマットに対する制限はなく、ファイルタイプの確認は行われません。Essbase サーバーからファイルを取得した後のファイルの処理は、ユーザーのクライアント・コンピュータに委ねられます。
- オブジェクトが URL である場合、Essbase によって、その URL はオブジェクト記述の一部としてカタログ項目に保管されます。ユーザーがその URL を表示しようとすると、Essbase で、予備的な構文チェックが行われます。その後、デフォルトの Web ブラウザで URL の存在が確認されます。
- オブジェクトがリンク・パーティションである場合は、Essbase のパーティション化モジュールで使用できます。

LRO に関連するタスクを実行する際は、次のことに注意してください:

- Essbase では、リンク・オブジェクトの検索と取得を行うために、データベース・インデックスを使用します。データベースからすべてのデータ値を消去するとインデックスが削除されるため、リンク・オブジェクトへのリンクも削除されます。データベースを再構築すると、インデックスは保持され、リンク・オブジェクトへのリンクも保持されます。
- 共有メンバーは、データ値は共有しますが、LRO は共有しません。これは、LRO が特定のメンバーの組合せにリンクされるためです。共有メンバーは同一のメンバーの組合せを持ちません。オブジェクトを共有メンバーにリンクするには、そのオブジェクトを各共有メンバーに個別にリンクします。
- リンク・オブジェクトに関連付けられたメンバーの組合せは変更できません。オブジェクトを別のメンバーの組合せに移動するには、そのオブジェクトを削除してから、Smart View を使用してオブジェクトを目的のメンバーの組合せに再度リンクします。

LRO への権限の設定

クライアント・インタフェースから LRO を追加、編集および削除するユーザーは、アクティブなデータベース内で適切なセキュリティ権限を持っている必要があります。オブジェクトがリンク・パーティションである場合は、ユーザーはリンク・パーティションが含まれているデータベース内でも必要な権限を持っている必要があります。表 29 は、様々なタスクで必要な権限のリストです。

表 29　LRO タスクで必要な権限

タスク	権限
リンクしたオブジェクトのデータベースへの追加	読取りと書込み
既存のリンク・オブジェクトの表示	読取り
既存のリンク・オブジェクトの編集	読取りと書込み
リンク・オブジェクトの削除	読取りと書込み
ファイルへの LRO カタログのエクスポート	読取り
LRO カタログ・ファイルからの LRO のインポート	読取りと書込み

データベース内の他のデータへのユーザー・アクセスを変更せずに、ユーザーがファイルをデータ・セルにリンクすることのないように、リンク・ファイルの最大ファイル・サイズを 1 に設定します。これによってユーザーは、セル・ノートの作成、URL へのリンクまたはリンク・パーティションの表示を行うことはできますが、添付できるのは 1KB 未満の小さいファイルのみです。

➤ アプリケーションの LRO ファイルの最大サイズを設定する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「LRO ファイルのサイズ制限」を参照してください。

LRO の表示と削除

ユーザーは、Smart View を使用してセル単位で LRO を操作します。つまり、ユーザーはセルを選択して、「リンク・レポート・オブジェクト」ダイアログ・ボックスを開きます。このダイアログ・ボックスには、選択したセルにリンクされたオブジェクトが表示されます。Administration Services を使用することで LRO を表示でき、データベース全体のすべての LRO を削除できます。また、LRO をユーザー名や最終変更日などの選択条件に基づいて表示することもできます。たとえば、ある日付より古いオブジェクトをすべて削除したり、会社を退職したユーザーに属するオブジェクトを削除したりできます。

➤ データベースのリンク・オブジェクトのリストを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	LRO の管理	Oracle Essbase Administration Services Online Help
MaxL	**query database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	LISTLINKEDOBJECTS	『Oracle Essbase テクニカル・リファレンス』

➤ データベースのリンク・オブジェクトを削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	LRO の管理	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	PURGELINKEDOBJECTS	『Oracle Essbase テクニカル・リファレンス』

LRO のエクスポートとインポート

バックアップやデータの移行機能を向上させるために、データベース内のデータの交差から LRO を一度エクスポートした後、再インポートできます。

➤ データベースのリンク・オブジェクトをエクスポートおよびインポートするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	LRO のエクスポート LRO のインポート	Oracle Essbase Administration Services Online Help
MaxL	**export lro** **import lro**	『Oracle Essbase テクニカル・リファレンス』

ストレージ保持のための LRO ファイル・サイズの制限

Essbase では、サーバー上のリポジトリにリンク・ファイルが保管されるため、デフォルトではサイズに制限はありません。ファイル・サイズを制限すると、ユーザーが非常に大きなオブジェクトを保管することによってサーバー・リソースが大量に消費されるのを回避できます。アプリケーションごとに、リンク・ファイルの最大サイズを設定できます。ユーザーが制限より大きいファイルをリンクしようとすると、エラー・メッセージが表示されます。

非常に小さなファイルの場合を除いてユーザーの添付を抑止するには、1 を入力します。ファイル・サイズを 1 にすると、ユーザーは、1KB 未満のセル・ノート、URL およびファイルのみをリンクできます。

注： 最大ファイル・サイズ設定は、リンク・ファイルのみに適用され、セル・ノートや URL には影響しません。セル・ノート、URL、文字列および LRO 記述の長さは固定されています。これらのオブジェクトの最大サイズについては、付録 A「制限」を参照してください。

➤ リンク・オブジェクトのサイズを制限するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	LRO ファイル・サイズの制限	Oracle Essbase Administration Services Online Help
MaxL	**alter application**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETAPPSTATE	『Oracle Essbase テクニカル・リファレンス』

12 型付きメジャーの操作

この章の内容

型付きメジャーについて

型付きメジャーを使用すると、Essbase の分析機能を拡張できます。メジャーは、数値のほか、テキスト型や日付型の値に関連付けることもできます。

テキスト・メジャーには、表示されるどの次元メジャーでも、「テキスト」のタグが付けられます。これにより、セル値に、テキスト・ラベルの一覧表示の 1 つを含めることができます。これらのラベルは、テキスト・リストと呼ばれるマッピング・アーティファクトを使用して、アウトライン・レベルで定義されます。

日付メジャーには、メジャーが表示される次元で「日付」のタグが付けられます。日付メジャーにより、セル値をフォーマットされた日付形式で使用可能となります。

次の一般的なガイドラインは、テキスト・メジャーと日付メジャーの両方に適用されます:

- 既存のメジャーの次元(たとえば、勘定科目)に追加します。
- 集計は行われません。デフォルトでは、テキスト・メジャーと日付には集計なし演算子(^)が割り当てられています。
- メジャーが集計されるように設計されていない場合は、データをロードしたレベルと同じレベルでクエリーを行う必要があります。
- アウトラインで型付きメジャーを使用可能にした後は、アウトラインを前の状態に戻して、型付きメジャーをサポートしないようにできません。
- テキスト・メジャーと日付メジャーの機能は、集約ストレージとブロック・ストレージの両方のアプリケーションに適用されます。

テキスト・メジャーの操作

テキスト・メジャーの概要とワークフロー

テキスト・メジャーは、Essbase の分析機能を、数値データを超えてテキストベースのコンテンツまで拡張します。テキストのコンテンツの格納と分析は、セルがテキスト値の確定リストの 1つを持つ必要がある場合に便利です。たとえば、5 つの異なる色の製品が販売されることがあります。色はテキスト・メジャーであり、その値はそれらの 5 つの色のいずれかである必要があります。確定リストにない色は、Essbase では範囲外とみなされます。

テキスト・メジャーはデータベース・レベルで作成します。テキスト・メジャーは、テキスト文字列のセットの対応する数値 ID へのマッピングによって可能になります。これらのマッピングは、ユーザーが作成するデータベースレベルのテキスト・リスト・オブジェクトに含まれます。

テキスト・メジャーを使用可能にして使用するには、次のワークフローを使用します:

1. アウトライン・プロパティで、型付きメジャーを使用可能にします。
2. テキスト・リスト・オブジェクトを作成して、使用可能なテキスト値を保管し、各テキスト値を序数にマップして、Essbase でテキスト値を操作できるようにします。オプションで、欠落と範囲外を序数にマップします。

   注： 各数値には、テキスト値を 1 つのみマップできます。
3. アウトライン内(会計次元内)にテキスト・メジャーを作成し、メンバー・プロパティで次のことを行います:
   1. タイプをテキストとして定義します。
   2. テキスト・リスト・オブジェクトに関連付けます。

197 ページの「テキスト・メジャーと日付メジャーに対するデータベース操作の実行」を参照してください。

テキスト・リスト・オブジェクトとテキスト・リスト・メンバー

テキスト・メジャーに対応するセルは、最大 1024 個の有効なテキスト値の確定リストの 1 つを持つことができます。Essbase では、内部的にはテキスト値を数値として保管する必要があります。したがって、テキスト値を数値にマップする必要があります。テキスト値と数値の間のマッピングは、テキスト・リスト・オブジェクトを作成することで定義します。テキスト・リスト・オブジェクトは、テキスト値と、それぞれのテキスト値に対応する数値のリストで構成されます。

たとえば、表 30 のコンテンツの「顧客満足度レベル」というテキスト・リスト・オブジェクトを作成できます。**名前**列には、テキスト・メジャーとして可能なテキスト値が含まれ、ID 列は Essbase で使用される内部数値を表します。

表 30　テキスト・リスト・オブジェクトの例

名前	ID
欠落	#MISSING
該当なし	#OUTOFRANGE
高い	1
普通	2
低い	3

各テキスト値は、一意の数値にマップする必要があります。テキスト・リスト・オブジェクト内の整数にマップしないテキスト値は、Essbase では無効とみなされます。

最初の 2 つの ID (#MISSING と#OUTOFRANGE)は、テキスト・データが無効または空であるケースを処理するためのものです。たとえば、ユーザーが「Average」などのマップされていない値をテキスト・メジャーにロードしようとすると、セル値は更新されず、以降のクエリーでは#MISSING と表示されます。マップされていない数値セル値をユーザーがロードすると、以降のクエリーでは該当なしが戻されます。

#MISSING と#OUTOFRANGE を除き、他のすべての ID は整数である必要があります。

日付メジャーの操作

日付メジャーの概要

日付メジャーを使用して、メンバーを日付型の値に関連付けることができます。メジャー次元の日付を処理する機能は、時間次元を使用して表現するのが困難な分析のタイプで役立ちます。

たとえば、資産の減価償却を分析するアプリケーションで、一連の資本資産の取得日を追跡する必要があるとします。この会社は事業を 50 年間行っているため、取得日の範囲が実現可能な時間次元モデリングの範囲を超えて長期にわたっています(時間次元がカバーするのは 5 年のみです)。

日付メジャーは、長期間にわたる値を表現できることに加え、時間次元で取り込まれるデータ値(時や分など)よりも細分度の低いデータ値を取り込むこともできます。

日付メジャーの実装

日付メジャーは、集約ストレージ・データベースとブロック・ストレージ・データベースでサポートされます。日付メジャーはデータベース・レベルで作成します。

日付メジャーを使用可能にして使用するには、次のワークフローを使用します:

1. アウトライン・プロパティで、型付きメジャーを使用可能にします。
2. アウトライン・プロパティで、日付フォーマット(たとえば、`yyyy-mm-dd`)を選択します。アウトライン内のすべての日付メジャーで同じフォーマットを使用する必要があります。
3. アウトライン内(会計次元内)に日付メジャーを作成し、メンバー・プロパティでその日付メジャーを「日付」型として定義します。

たとえば、ASOsamp.Sample では、アウトラインの型付きメジャーを使用可能にし、日付フォーマットを選択して、「日付」型として定義された IntroDate というメジャーを追加します。

日付値は内部的には数値として保管されますが、それらの値は、Essbase にはフォーマットされた日付文字列としてロードされます。クエリーされると、日付メジャーは選択された日付フォーマットに応じて表示されます。

197 ページの「テキスト・メジャーと日付メジャーに対するデータベース操作の実行」を参照してください。

日付メジャーをサポートする関数

日付メジャーに基づいた計算に便利な MDX 関数は次のとおりです。

- DateDiff
- DatePart
- DateRoll
- FormatDate
- GetFirstDate
- GetLastDate
- ToDate
- ToDateEx
- Today

日付メジャーに基づいた計算に便利な計算関数は次のとおりです。

- @DATEDIFF
- @DATEPART
- @DATEROLL
- @FORMATDATE
- @TODATEEX
- @TODAY

DATEFORMAT レポート・ライター・コマンド。

これらの関数とコマンドの詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

テキスト・メジャーと日付メジャーに対するデータベース操作の実行

この項では、テキスト・メジャーと日付メジャーを使用する際の一般的なデータベース操作の実行方法について説明します。

テキスト・メジャーと日付メジャーのロード、消去およびエクスポート

データをテキスト・メジャーまたは日付メジャーにロードするには、数値メジャーを持つメンバーにデータをロードする場合と同じ手順に従います。入力データには、フォーマットされた日付値、またはテキスト・メジャーに関連付けられているテキスト・リスト・オブジェクトに対応するテキスト値を含める必要があります。

メンバーに関連付けられているテキスト・リスト・オブジェクトには存在しないテキスト値をロードしようとすると、Essbase で警告メッセージが表示されます。

集約ストレージ・データベースで値をロードできるのは、入力レベルのみです。この制限は、テキスト・メジャーと日付メジャーで同様に適用されます。ブロック・ストレージ・データベースでは、テキスト値と日付値はどのレベルでもロードできます。

テキスト値と日付値を集約ストレージ・データベースにロードする際は、次のガイドラインに従います。これらのガイドラインは、無効な集約をなくすために役立ちます。

- 置換モードを使用します。

  注： 置換モードは、バッファのコミット時に設定します。MaxL では、**import data** ステートメントの **override values** 構文を使用します。管理サービス・コンソールでは、「データ・ロード」ダイアログ・ボックスの「既存の値の上書き」オプションを使用します。Essbase API では、EssLoadBufferTerm の ulCommitType フィールドを使用します。

- 日付/テキスト・メジャーに関連付けられているすべての値をロードするには、単一のロード・バッファを使用します。
- aggregate_use_last 集約メソッドを使用します。

  注意 aggregate_use_last メソッドは、パフォーマンスに重大な影響を及ぼすため、大容量のデータ・ロード向きではありません。データ・ロードが 100 万セルを超える場合、数値データを(型付きメジャー・データとは)別のデータ・ロード・プロセスに分けることを検討してください。この個別データ・ロードでは、aggregate_sum をかわりに使用できます。

  aggregate_use_last は、ロード・バッファの作成時に設定します。MaxL では、**alter database** ステートメントの **initialize load_buffer** 構文の一部である PROPS ターミナルを参照してください。管理サービス・コンソールでは、「データ・ロード」ダイアログ・ボックスの「最後に集約使用」チェック・ボックスを

選択します。Essbase API では、EssLoadBufferInit の ulDuplicateAggregationMethod フィールドを使用します。

- #Missing 値を増分データ・ロード・モードでテキスト/日付メジャーにロードしないようにします。#Missing 値を増分ロードで Missing 以外の値を持つセルにロードすると、ゼロ値と置換されます。日付/テキスト・メジャーでは、ゼロ値は#Missing 値とは同じ意味を持たないことがあります。#Missing 値を日付/テキスト・メジャーにロードする必要がある場合は、完全なデータ・ロードを使用してください。

混在する(数値とテキストまたは日付)データをロードする場合は、置換モードが数値データに対して十分であることを確認するか、数値データ用に別個のデータ・ロード・プロセスを作成します。

テキスト値や日付値は、ルール・ファイルを使用しても使用しなくてもロードできます。ルール・ファイルを使用しない場合は、テキスト値を二重引用符で囲み、そのテキスト値に文字列`#Txt:`の接頭辞を付けることによって、テキスト値や日付値をメンバー名と区別する必要があります。

次に、フリーフォームのデータ・ロード・ファイルのデータ行の例を示します:

```
"100-10" "New York" "Cust Index" #Txt:"Highly Satisfied"
```

テキスト値「Highly Satisfied」には、「New York」などのメンバー名と区別するため、`#Txt:`という接頭辞が付いています。

接頭辞「#Txt」は、データ・ロードにルール・ファイルを使用しない場合の日付メジャーにも必要です。

テキスト値や日付値の消去、ロックしてから送信、およびエクスポートは、数値に対してこれらの操作を実行するのと同様に実行できます。

テキスト・メジャーと日付メジャーの集計

デフォルトでは、テキスト・メジャーには「^」(集計なし演算子)が割り当てられています。テキスト・メジャーと日付メジャーが、会計以外の次元に従って上位レベルのメンバーに集計されることはありません。

テキスト・メジャーや日付メジャーに「^」以外の演算子でタグ付けすると、その内部数値に基づき、他の次元に従って集計されます。これは、集約ストレージ・データベースにはお薦めできません。サポートされているのが+演算子のみであり、テキスト・メジャーや日付メジャーに対する集約値の妥当性が失われる可能性があるためです。さらに、Essbase では、集計中に範囲外の値は#OUTOFRANGE に変換されません。

ブロック・ストレージ・データベースの場合は、カスタム方法でテキスト・メジャーを集計する計算スクリプトを記述できます。テキスト・メジャーがランキング・メジャーを表している場合は、そのテキスト・メジャーを集計できます。たとえば、「CustomerSatisfaction」というテキスト・リストに、Excellent=5、Good=4、Fair=3、Poor=2、Bad=1 などのマッピングが含まれているとします。これらの値はレベル 0 でロードされます。子レベルで値の平均をとることによって、

値を親レベルに集計できます。たとえば、[Qtr1]の「CustomerSatisfaction」は[Jan]、[Feb]、[Mar]の値の平均です。

テキスト・メジャーと日付メジャーでのデータの取得

テキスト・メジャーや日付メジャーは、Smart View、グリッド API クライアント、レポート・ライターおよび MDX を使用して、数値メジャーと同じ方法でクエリーできます。適切なテキスト値やフォーマットされた日付値を含む、対応するセルが表示されます。

次のレポート・ライター・コマンドは、数値データを処理するために設計されており、テキスト・メジャーや日付メジャーではサポートされていません:

- RESTRICT
- TOP
- BOTTOM
- SORT*コマンド
- CALCULATE COLUMN
- CALCULATE ROW

MDX 関数の EnumValue と計算関数の@ENUMVALUE は、テキスト・メジャーのテキスト値を取得するために特別に設計されたものです。これらの関数は、内部的に保管されている数値ではなく、メンバーのテキスト値に基づいて操作を行う必要がある場合に、MDX スクリプト、計算スクリプトまたは式で役立ちます。『Oracle Essbase テクニカル・リファレンス』を参照してください

テキスト・メジャーと日付メジャーの制限

アウトラインの再構築では、テキスト・リストは再構築されません。テキスト・リスト内の数値からテキスト値へのマッピングが変更された場合、この変更は、そのテキスト・リストに関してデータベース内にすでに存在するテキスト・データに反映されます。したがって、テキスト・リストにアイテムを追加する場合は、既存のテキスト・リストのアイテムのマッピング数が変更されないように、そのアイテムをリストの一番上か一番下に追加します。

テキスト・メジャーと日付メジャーは、複数のパーティションにまたがることはありません。

共有メンバーと暗黙的な共有メンバーは、元のメンバーのテキスト型や日付型を継承します。

フォーマット文字列の操作

フォーマット文字列の概要

フォーマット文字列を使用すれば、Essbase データベース・メンバーの値(セル・コンテンツ)を数値タイプ・メジャーでフォーマットして、クエリーの実行時に、事前定義されたテキスト、日付またはその他のタイプの値として表示できます。結果として表示される値はセルのフォーマットされた値(MDXでは FORMATTED_VALUE プロパティ)です。

基礎となる実際の値は数値であり、この値は関連付けられたフォーマット済の値には影響されません。フォーマット文字列を使用すれば、実際の数値のかわりに意味のある値を表示できます。たとえば、テキスト・ベースのフォーマットされた値を使用すると、データ・セルに「High」、「Medium」および「Low」として表示できます。

テキスト型と日付型の値は、組込みのテキスト・メジャーと日付メジャーの型を使用して追加でサポートされます。フォーマット文字列では、複数の次元内のメンバーにフォーマット文字列を適用できるため、実装の柔軟性が増しますが、テキスト・メジャーと日付メジャーの場合は、単一のメジャー次元にどちらか一方を適用することしかできません。フォーマット文字列は数値次元に適用でき、次元をテキストまたは日付として入力する必要はありません。

フォーマット文字列は、集約ストレージとブロック・ストレージのいずれかのデータベースに適用できます。

フォーマット文字列は、次のメンバーに対して定義できます:

- メジャー次元内のすべてのメンバー
- 他の次元の明示的な式文字列に関連付けられたメンバー

フォーマット文字列の実装

フォーマット文字列は、集約ストレージ・データベースとブロック・ストレージ・データベースでサポートされます。フォーマット文字列はデータベース・レベルで実装します。

フォーマット文字列を使用可能にして使用するには、次のワークフローを使用します:

1. アウトライン・プロパティで、型付きメジャーを使用可能にします。
2. 会計次元で、メンバーをフォーマットするメジャーを作成し、そのメンバー・プロパティで「フォーマット文字列の関連付け」フィールドを編集して、MDX フォーマット・ディレクティブを作成します。MDX フォーマット・ディレクティブを作成する構文は、200 ページの「MDX フォーマット・ディレクティブ」を参照してください。

MDX フォーマット・ディレクティブ

フォーマット文字列は、次の構文を使用して定義します:

```
format_string_expression = MdxFormat (
string_value_expression
)
```

ここで、string_value_expression は有効な MDX 文字列値式です。『Oracle Essbase テクニカル・リファレンス』に記載されている MDX 仕様を参照してください。

ほとんどの MDX 式は、フォーマット文字列を指定するために使用できます。ただし、フォーマット文字列には、フォーマット中の現在のセル値以外のデータ・セルの値への参照を含めることができません。現在のセル値は、MDX の **CellValue** 関数を使用して参照できます。

Essbase では、無効なフォーマット文字列を含むメンバーは、フォーマット文字列が定義されていないかのように扱われます。アウトラインは無効なフォーマット文字列とともに保存できます。クエリーが無効なフォーマット文字列を持つメンバーで構成されている場合は、Essbase で、警告が生成されます。

メンバーがフォーマット文字列に関連付けられていない場合は、デフォルトのフォーマット・ルールが適用されます。デフォルトのフォーマット・ルールでは、メジャーが数値、テキストまたは日付型かに基づいて、セルがフォーマットされます。数値メジャーの場合、デフォルトのフォーマット済の値はその数値のテキスト・バージョンです。テキスト・メジャーの場合、デフォルトのフォーマット済の値は、関連付けられたテキスト・リスト・オブジェクトに基づいたテキスト値です。日付値の場合、デフォルトのフォーマットは、アウトライン・プロパティに定義されている日付フォーマット文字列に従ってフォーマットされた日付文字列です。

フォーマット文字列をサポートする関数

次の MDX 関数は、メジャーにフォーマット文字列を適用するときに便利です。フォーマット文字列は MDX 式として、集約ストレージ・データベースとブロック・ストレージ・データベースのどちらにも適用できます。

- EnumText は、内部数値に関連付けられたテキスト・リスト・ラベルを戻します。
- EnumValue は、テキスト・リスト・ラベルの内部数値を戻します。
- CellValue は、現在のセルの内部数値を戻します。
- NumToStr は、値を 10 進数文字列に変換します。

@ENUMVALUE 計算関数は、テキスト・メジャーまたはフォーマット文字列を含むブロック・ストレージ・データベースの計算スクリプトを作成するときに便利です。この関数は、内部数値に関連付けられたテキスト・リスト・ラベルを戻します。

MaxL **alter session set dml_output** ステートメントには、**set formatted_value on | off** 句が含まれています。デフォルトでは、クエリー内にはフォーマット済の値が表示されますが、このステートメントを使用するとフォーマット済の値の表示をオフにできます。

OUTFORMATTEDVALUES レポート・ライター・コマンドは、レポート内のフォーマット済のセル値を戻します。

これらの関数、コマンドおよびステートメントの詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

フォーマット文字列の制約事項

フォーマット文字列は、複数のパーティションではサポートされません。

共有メンバーと暗黙的な共有メンバーに別々のフォーマット文字列を持たせることはできません。これらのメンバーはオリジナルのメンバーのフォーマット文字列を継承します。

13 Oracleアプリケーションへのドリルスルー

この章の内容

Oracle アプリケーションへのドリルスルーの概要

Essbase では、Essbase クライアントのレポート作成インタフェース(Smart View や Oracle Hyperion Financial Reporting など)から、Oracle Enterprise Resource Planning (ERP)アプリケーションや Enterprise Performance Management (EPM)アプリケーション上の情報への、URL ベースのドリルスルー・アクセスを提供しています。

このドキュメントでは、Oracle ERP および EPM アプリケーションにホスティングされている情報をドリルスルーする機能について説明します。このドリルスルーは、Integration Services および Essbase Studio のドキュメントで説明されているドリルスルーの概念とは異なります。このドキュメントにおけるドリルスルーは、Essbase データベース・セルを他の Oracle アプリケーションに含まれている情報にリンクする機能を指します。Integration Services および Essbase Studio のドキュメントのドリルスルーは、多次元データベースのセルを、リレーショナル・ソースに保管されているトランザクション・レベルのデータなどの、詳細なデータにリンクすることを指しています。

Oracle General Ledger または Oracle Hyperion EPM Architect を使用して Essbase データベースを配置する際に、Essbase API を使用して、Essbase データベースに、Essbase データベース・セルの有効領域(エンタープライズ・レポート作成アプリケーションへのドリルスルー・アクセスが使用可能な領域)についての情報を設定します。

Essbase データベースの各 Essbase ドリル可能領域ごとに、URL によるドリルスルー・アクセスを使用可能に設定します。Essbase API を使用して、Essbase データベースに、URL 情報とドリル可能領域の情報を設定します。この URL は、Essbase クライアント・インタフェースのユーザーに対してはセルに関連付けられたリンクとして表示され、Oracle ERP および EPM アプリケーションによってホスティングされている関連情報へのアクセスを提供します。

たとえば、Smart View において、あるセルが 1 月の New York 市場における Cola の売上実績データを表しているとします。このセルの色分けは、このセルに関連付けられたレポートがあることを示しています。このセルには複数のリンクを関連付けることができ、ユーザーはスクロールして選択できます。各リンクでは URL

が使用されます。ユーザーがリンクをクリックすると、URL が検証され、ERP または EPM アプリケーションによってホストされている起動ページが、Web ブラウザに表示されます。

ドリルスルー URL の理解

ERP および EPM アプリケーションは Essbase を使用してドリルスルー URL を作成します。ドリルスルー URL は Essbase データベース・ファイルにメタデータとして保管されます。

データベース当たりのドリルスルー URL の数は、255 に制限されています。

注： ドリルスルー定義の設定、およびドリルスルー URL のターゲットとして使用する Web ページのホスティングは、ERP および EPM アプリケーションの管理者の作業となります。

ドリルスルー URL のコンポーネントの詳細は、次のトピックを参照してください:

- 204 ページの「ドリルスルー URL 名」
- 204 ページの「ドリルスルー URL XML」
- 205 ページの「ドリル可能領域のリスト」
- 205 ページの「レベル 0 ブール・フラグ」

ドリルスルー URL 名

ドリルスルー URL 名は、定義されているドリルスルー URL を管理するための識別子です。この名前は Essbase クライアントからエンド・ユーザーに表示される URL 表示名とは異なる場合があります。

ドリルスルー URL XML

ドリルスルー URL XML は、プロトコル内に構造化されている XML 情報のブロックであり、これにより Essbase が、指定されたデータベース領域を Oracle ERP および EPM アプリケーション上の情報にリンクできるようになります。この URL XML は、アプリケーションのクエリーを実行するエンド・ユーザーに対しては透過的です。URL XML は、Essbase データベースを配置した ERP または EPM アプリケーションによって指定されます。管理者による URL XML の編集はお薦めしません。ただし、Essbase は URL XML 編集用のインタフェースを備えています。XML ブロックにはドリルスルー URL の表示名の他、セルから Web インタフェースへのハイパーリンクを可能にするための URL が含まれています

ドリル可能領域のリスト

ドリル可能領域のリストは、指定された URL を使用したドリルスルーを許可するデータベース領域を定義するメンバー指定です。管理者は、1 つ以上の次元のメンバーのメンバー指定を使用してドリル可能領域のリストを定義します。セキュリティ・フィルタの定義に使用するものと同じ Essbase メンバー・セット計算言語を使用してメンバー指定を定義します。次に示す例は、Qtr1 の月について、New York 以外のすべての東部州を指定する、有効なメンバー指定です:
@REMOVE(@DESCENDANTS("Eastern Region"), "New York"), @CHILDREN(Qtr1)

ドリルスルー URL のドリル可能領域の数は、256 に制限されています。ドリル可能領域当たりの文字数は、65536 に制限されています。

レベル 0 ブール・フラグ

このフラグは、ドリル可能領域のリストで指定された領域のレベル 0 の子孫に対してのみ URL が適用されるかどうかを示します。

たとえば、ドリル可能領域 DESCENDANTS("Market"), @CHILDREN(Qtr1)に対してレベル 0 フラグが使用可能に設定されている場合、URL は、Market のすべての州について Qtr1 のすべての月にわたって適用され、また、残りの次元のすべてのレベル 0 メンバーに対して適用されます。

ドリルスルー URL の作成および管理

この項で参照する MaxL ステートメントと API の詳細は、『Oracle Essbase テクニカル・リファレンス』と『Oracle Essbase API リファレンス』をそれぞれ参照してください。

以下の MaxL ステートメントはドリルスルー URL を管理するために使用します:

- create drillthrough
- alter drillthrough
- display drillthrough
- drop drillthrough

Essbase データベース上のドリルスルー URL を管理するには、次の Essbase API 構造体および関数を使用します:

- C のメイン API 構造体:
  - ESS_DURLINFO_T
- C のメイン API 関数:
  - EssCreateDrillThruURL
  - EssDeleteDrillThruURL
  - EssGetDrillThruURL
  - EssGetCellDrillThruReports

    - EssListDrillThruURLs
    - EssMDXIsCellGLDrillable
    - EssUpdateDrillThruURL
- C グリッド API 構造体:
    - ESSG_DATA_T
- C グリッド API 関数:
    - EssGGetIsCellDrillable
- Visual Basic API 構造体:
    - ESB_DURLINFO_T
- Visual Basic API 関数:
    - EsbCreateDrillThruURL
    - EsbUpdateDrillThruURL
    - EsbDeleteDrillThruURL
    - EsbListDrillThruURLs
    - EsbGetDrillThruURL
    - EsbGetCellDrillThruReports

Administration Services でドリルスルー URL を管理するには、Oracle Essbase Administration Services Online Help の次のトピックを参照してください:

- ドリルスルー定義の管理
- 「ドリルスルー定義の編集」ダイアログ・ボックス

14 通貨換算アプリケーションの設計および作成

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

通貨換算について

Essbase の通貨換算機能では、財務データの通貨換算を行うことができます。通貨換算を行うと、各国間の比較が容易になり、使用通貨が異なる複数の場所の財務データを集計できます。

たとえば、英国の収益データ(英ポンド単位)と日本の収益データ(日本円単位)を分析する組織があるとします。この場合、それぞれの現地通貨で収益データを比較しても無意味です。2 か国の相対的寄与率を把握するには、英ポンドを日本円に換算するか、日本円を英ポンドに換算するか、英ポンドと日本円の両方を別の共通の通貨に換算する必要があります。

もう 1 つの例として、北米地域の総収益を報告するには、北米地域に含まれる各現地通貨を標準化する必要があります。北米地域の総収益を、米国、メキシコ、カナダの 3 か国の収益データの合計と想定した場合、各国の現地通貨のデータを集計しても無意味です。意味のあるデータを得るには、集計の前に、各現地通貨を共通通貨に換算する必要があります。

Essbase のインストール時には、オプションとして、Interntl と Xchgrate の 2 つのデータベースからなるサンプル通貨アプリケーションをインストールできます。サンプル通貨アプリケーションのデータベースにアクセスできない場合は、Essbase 管理者に問い合せてください。詳細は、付録 B「サンプル・アプリケーションの設定」を参照してください。

サンプル通貨アプリケーションについて

サンプル通貨アプリケーションは、第 4 章「ケース・スタディ: 単一サーバー、多次元データベースの設計」で紹介している The Beverage Company (TBC)が米国外にビジネスを拡大するビジネス・シナリオに基づいて構築されています。TBC は次の市場を新しく追加します:

- カナダ国内の 3 都市: Toronto、Vancouver、Montreal
- ヨーロッパの 4 か国: UK、Germany、Switzerland、Sweden

さらに、米国内の各地域(East、West、South、Central)のデータの集計を、U.S.という名前の新しいメンバーとして追加します。

TBC の各市場地域のデータは、現地通貨で収集されます。US ドルの値は、現地通貨の値に為替レートを適用して計算します。

TBC は、実績データを次の 2 通りの方法で分析する必要があります:

- 実績データ(Actual)を実際の為替レートで換算します。
- 為替レートによる差異を分析するために、実績データを予算為替レートで換算します。

実績データをすべて処理したら、予算データを予算為替レートで換算します。

TBC 通貨アプリケーションは、メイン・データベース(Interntl)と通貨データベース(Xchgrate)で構成されます。これらのデータベースは、Essbase サーバー上のサンプル・アプリケーションに含まれています。これらのデータベースにアクセスできない場合は、Essbase 管理者に問い合せてください。

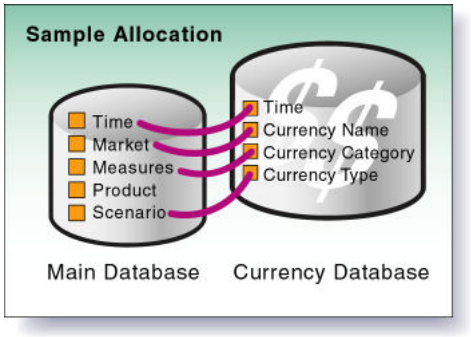
通貨アプリケーションの構造

通貨換算の必要なビジネス・アプリケーションでは、メイン・データベースが少なくとも 2 つのスライスに分割されます。一方のスライスはローカル・データの入力処理に、もう一方のスライスは共通通貨に換算された入力データのコピーを保管するために使用されます。

Essbase では、通貨換算に必要な為替レートは、別個の通貨データベースに保管されます。通貨データベースのアウトラインは、ユーザーが必要なタグを割り当てると Essbase によって自動生成されます。一般には、このアウトラインによって、メイン・データベースの 1 セクションに所定の換算率がマップされます。生成された通貨データベースは、他の Essbase データベースと同様に編集できます。

メイン・データベースと通貨データベースの関係については、図 46 を参照してください。

図 46　通貨アプリケーションのデータベース



メイン・データベース

Essbase で通貨データベースのアウトラインを自動生成するには、メイン・データベースのアウトライン内の次元およびメンバーを変更します。サンプル通貨アプリケーションのメイン・データベースは Interntl です。

メイン・データベースのアウトラインには、3 から n 次元を含めることができます。メイン・データベースには少なくとも次の次元を含める必要があります:

- 時間のタグが付けられた次元。次元に時間のタグを付けると、通貨データベース内に、メイン・データベース内の時間次元とまったく同じ次元が生成されます。Sample.Interntl データベースでは、時間のタグが付けられた次元は年次元です。
- 勘定科目のタグが付けられた次元。次元に勘定科目のタグを付け、この次元のメンバーに通貨カテゴリを割り当てると、通貨データベース内に、各通貨カテゴリのメンバーを含む次元が生成されます。カテゴリを割り当てることで、複数の勘定科目またはメジャーに、異なる為替レートを適用できるようになります。Sample.Interntl データベースでは、勘定科目のタグが付けられた次元はメジャー次元です。

  あるメンバーの子孫は各自、その祖先の通貨カテゴリ・タグを継承します。メンバーまたはメンバーのサブ分岐にも、固有のカテゴリを割り当てることができます。

  たとえば、損益(P&L)の勘定科目では、バランス・シートの勘定科目で使用されている為替レートとは異なるレートが使用されている場合があります。さらに、勘定科目によっては、換算が不要な場合もあります。たとえば、Sample.Interntl データベースの Margin%メンバーや Profit%メンバーは換算不要です。換算不要のメンバーには、換算なしタグを付けます。換算なしタグは継承されません。
- 国のタグが付けられた市場関連の次元。次元に国のタグを付け、各国に通貨名を割り当てると、通貨データベース内に各通貨を表すメンバーが生成されます。たとえば、Sample.Interntl データベース内の市場次元には、国のタグが付けられています。すべての現地通貨は、この会社の共通通貨である USD (US ドル)に換算する必要があるので、この次元の通貨名は USD です。

  複数のメンバーに同じ通貨名を割り当てることができるので、通貨名の数は、通常、次元内の合計メンバー数より少なくなります。表 31 に示すように、Sample.Interntl データベースでは、市場次元のメンバー数が 15 であるにもかか

わらず、通貨名を 6 つしか使用していません。Europe メンバーの子は、それぞれ異なる通貨を使用するので、通貨名を個別に割り当てる必要があります。これに対して、U.S.次元とその 4 つの地域メンバーは、同じ通貨を使用します。同様に、Canada メンバーとその 3 つの都市メンバーも同じ通貨を使用します。あるメンバーの子が 1 つの通貨を共有する場合、定義する必要があるのは、親メンバーの通貨名のみです。

表 31　Interntl データベースの通貨名

次元とメンバー	通貨名
市場 - 国 U.S. East West South Central	USD (US ドル)
Canada Toronto Vancouver Montreal	CND (カナダ・ドル)
Europe UK Germany Switzerland Sweden	GBP (UK ポンド) EUR (ユーロ) CHF (スイス・フラン) SEK (スウェーデン・クローナ)

メイン・データベースの通貨換算用アウトラインを準備する際、オプションの通貨パーティションを作成して、現地通貨データを保管するデータベース・スライスと換算後の通貨データを Essbase に指定できます。通貨パーティションのタグが付けられた次元には、現地通貨の値と換算後の値の両方のメンバーが含まれます。現地通貨データは、通貨換算スクリプトによって共通通貨データに換算されます。Sample.Interntl データベースでは、シナリオ次元が通貨パーティション次元になります。

通貨パーティション次元の使用方法については、216 ページの「ローカル値と換算値の保持」を参照してください。

注： 通貨換算パーティションは、「通貨換算」オプションのみに適用されます。複製パーティション、リンク・パーティション、透過パーティションを使用してデータベース間でのデータ共有を可能にする「パーティション化」オプションとは無関係です。

レポート・スクリプトを使用すると、レポートを表示した状態でデータ換算を行うレポートを作成できます。これについては、217 ページの「レポート・スクリプトでの通貨換算」を参照してください。

注： メイン・データベースのアウトラインの作成方法については、213 ページの「メイン・データベース・アウトラインの作成」を参照してください。

通貨データベース

メイン・データベース・アウトライン内のメンバーに通貨タグを割り当てることで、Essbase で自動的に通貨データベースを生成できます。サンプル通貨アプリケーションの通貨データベースは Xchgrate です。

注： 通貨データベースでは、すべてのレベル 0 メンバーが、非動的計算メンバーとして格納されている必要があります。これは、通貨データベースの生成元のデータベースでも、すべてのレベル 0 メンバーが非動的計算メンバーとして格納される必要があることを意味します。

通貨データベースは、常に次の 3 つの次元で構成されています。オプションで 4 番目の次元があります:

- 時間のタグが付けられた次元は、一般に、メイン・データベース内の時間のタグが付けられた次元と同じです。このため、通貨データベースでは、長期にわたる通貨の変動を追跡し、メイン・データベースの様々な時間スライスを正確に変換できます。Sample.Xchgrate データベースでは、時間のタグが付けられた次元は年次元です。

  通貨データベースには、メイン・データベース内の時間次元の各メンバーを定義する必要があります。メイン・データベース内の期間ごとの値は通常、通貨データベース内の各期間の為替レートに換算されます(なお、データ値は任意の期間の為替レートに換算できます)。

- 国のタグが付けられた次元には、メイン・データベース内に定義された市場(または国)の通貨名が含まれます。通貨データベース内には、メイン・データベース内に定義された各通貨名が存在している必要があります。通貨名によって、通貨換算の際の国と為替レートのマッピングが定義されます。

  Sample.Xchgrate データベースでは、国次元は CurName 次元です。CurName 次元内の通貨名については、表 32 を参照してください:

表 32　Xchgrate データベースの通貨名

次元とメンバー	別名
CurName- 国	
USD	US ドル
CND	カナダ・ドル
GBP	UK ポンド
EUR	ユーロ
CHF	スイス・フラン
SEK	スウェーデン・クローナ

- 勘定科目のタグが付けられた次元では、メイン・データベース内の勘定科目のタグが付けられた次元のメンバーに様々なレートを適用できます。メイン・

データベース内の会計次元用に定義されたカテゴリから、通貨データベースの会計次元メンバーが生成されます。たとえば、総利益(Gross Profit)と純利益(Net Profit)の換算には特定のカテゴリのレートを使用し、その他の勘定科目の換算には他のレートを使用する必要が生じることがあります。

Sample.Xchgrate データベースでは、勘定科目のタグが付けられた次元は CurCategory 次元です。この次元の勘定科目カテゴリには、損益(P&L)とバランス・シート(B/S)があります。

- 通貨データベースには、一般に、オプションの通貨タイプ次元が含まれるので、異なる通貨換算シナリオを利用できます。通常、アプリケーションの為替レートはシナリオ(実績、予算、予測など)ごとに異なります。シナリオ間でデータ換算を行うには、使用するレートのタイプを選択します。

  通貨アウトラインの生成時には、通貨タイプ次元が作成されます。通貨タイプ次元は、メイン・データベースには直接マップされません。このため、この次元内のメンバー名は、メイン・データベース内のメンバー名と一致していなくてもかまいません。

  Sample.Xchgrate データベースでは、通貨タイプ次元は CurType 次元です。CurType には、実績シナリオと予算シナリオが含まれます。

注： 通貨データベースのアウトラインの作成方法については、213 ページの「通貨換算アプリケーションの構築および換算の実行」を参照してください。

換算方法

通貨アプリケーションによって、換算の要件が異なります。Essbase では、次の 2 通りの換算方法がサポートされています:

- ローカル値の換算値による上書き。

  一部のアプリケーションでは、換算後の値のみをメイン・データベースに保管する必要があります。現地通貨の値を入力すると、換算操作により、現地通貨の値が共通通貨の値で上書きされます。これは、現地通貨のレポートや分析が不要であると想定した換算方法です。

  データが上書きされるので、換算を行うたびに現地通貨の値をロードし、データを再計算する必要があります。この方法は、単発の(継続しない)換算にのみ適しています。

- ローカル値と換算値の保持。

  ほとんどのアプリケーションでは、データを現地通貨と共通通貨(換算後の値)の両方で保管する必要があります。この方法では、現地通貨のデータのレポートと分析が可能で、データの変更や再計算の制御が比較的容易です。この方法を使用する場合は、通貨パーティションを定義します(209 ページの「メイン・データベース」を参照)。

どちらの方法でも、レポート時に通貨換算が必要になる可能性があります。レポート時の換算の場合、実際にデータベースにデータを保管することなく、様々な為替レートのシナリオを分析できます。通貨換算モジュールでは、アド・ホックな

換算を実行できます。レポート・スクリプトを使用してアド・ホックな換算を実行します。例を217 ページの「レポート・スクリプトでの通貨換算」に示します。

通貨換算アプリケーションの構築および換算の実行

通貨換算アプリケーションを構築して換算を実行するには:

1. メイン・データベースのアウトラインを作成するか、開きます。

   213 ページの「メイン・データベース・アウトラインの作成」を参照してください。

2. 通貨換算のためにメイン・データベース・アウトラインを準備します。

   214 ページの「メイン・データベース・アウトラインの準備」を参照してください。

3. 通貨データベース・アウトラインを生成します。

   214 ページの「通貨データベース・アウトラインの生成」を参照してください。

4. メイン・データベースと通貨データベースをリンクします。

   214 ページの「メイン・データベースと通貨データベースのリンク」を参照してください。

5. 通貨値を換算します。

   214 ページの「通貨値の換算」を参照してください。

6. 通貨換算を追跡します。

   218 ページの「通貨換算の追跡」を参照してください。

7. 必要に応じて、通貨換算のトラブルシューティングを行います。

   220 ページの「通貨換算のトラブルシューティング」を参照してください。

メイン・データベース・アウトラインの作成

メイン・データベースのアウトラインを作成するには、Essbase データベースのアウトラインを作成するか開いて、必要に応じて変更を加えた後、通貨換算アプリケーションで使用できるようにアウトラインを保存します。

➤ アウトラインを作成するか、既存のアウトラインを開くには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アウトラインを開いて編集する	Oracle Essbase Administration Services Online Help
MaxL	**create database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	CREATEDB	『Oracle Essbase テクニカル・リファレンス』

メイン・データベース・アウトラインの準備

メイン・データベースのアウトラインを作成するか開いた後、次元とメンバーに変更を加えて、Essbase で自動的に通貨データベースのアウトラインが生成されるようにします。209 ページの「メイン・データベース」を参照してください。

➤ メイン・データベースのアウトラインを準備する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「通貨換算のためのメイン・データベース・アウトラインの準備」を参照してください。

通貨データベース・アウトラインの生成

メイン・データベースのアウトラインの確認と保存が完了したら、通貨アウトラインを生成できます。通貨アウトラインは、メイン・データベースのアウトラインで定義した次元、メンバー、通貨名および通貨カテゴリで構成されます。通貨データベースのアウトラインは、基本的に構造化されており、生成後すぐに使用できます。ただし、完成するためにはいくらか追加が必要になることがあります。

➤ 通貨データベースのアウトラインを生成する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「通貨データベースのアウトラインの生成」を参照してください。

メイン・データベースと通貨データベースのリンク

通貨換算を実行するには、Essbase にメイン・データベースと通貨データベースのリンクを認識させる必要があります。通貨アウトラインを生成しただけでは、メイン・データベースと通貨データベースはリンクされません。データベースのリンクを作成する際、換算の計算方法とデフォルトの通貨タイプ・メンバーを指定します。

➤ メイン・データベースと通貨データベースをリンクする方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「データベースから通貨データベースへのリンク」を参照してください。

通貨値の換算

通貨換算アプリケーションを作成したら、計算スクリプト内の CCONV コマンドを使用して、現地通貨のデータ値を共通通貨に換算します。たとえば、様々な通貨のデータを USD (US ドル)に換算できます。データをオリジナルの現地通貨に換算しなおすには、CCONV TOLOCALRATE コマンドを使用します。

通貨データベースに定義したレートを使用して、メイン・データベースの一部または全体を換算できます。現地通貨の値を換算後の値で上書きするか、両方の値をメイン・データベース内に保管するかは、追跡およびレポートのニーズに応じてユーザーが指定できます。

注： 通貨換算を実行する際は、換算対象のデータが同時に別のユーザーのアクティビティ(計算、データ・ロード、同じ通貨パーティションに対する通貨換算など)によって更新中でないことを確認してください。換算対象のデータに対するアクティビティが同時に発生した場合、不正な結果になる可能性があります。この場合、Essbase からの警告メッセージは表示されません。

注： CCONV コマンドを使用して通貨換算を実行すると、換算後のデータ・ブロックは高機能計算には適さないデータとしてマークされます。このため、データベースの再計算時に、換算済のすべてのブロックが Essbase で再計算されます。

通貨換算のサンプル計算スクリプトについては、『Oracle Essbase テクニカル・リファレンス』を参照してください。

ローカル値の換算値による上書き

現地通貨の値を上書きするために、メイン・データベース内に通貨パーティションを作成する必要はありません。計算スクリプト内に CCONV コマンドを記述するだけで、データベース内のすべてのデータを換算できます:

次の計算スクリプトでは、データベース内の値が USD に換算されます:

```
   CCONV USD;
CALC ALL;
```

必要に応じて、必要な為替レートが含まれる通貨名を指定できます。次の計算スクリプトでは、通貨データベース内の Jan と定義された為替レートを使用して、データベース内の値を USD に換算できます:

```
   CCONV Jan->USD;
CALC ALL;
```

例として示されたケースでは、CALC ALL コマンドを使用する必要があります。CCONV コマンドで実行できるのは通貨換算のみで、データベース内のメンバーの集計または計算は実行されません。

次の計算スクリプトでは、Act xchg レートを使用して、換算値が元の現地通貨の値に換算されます:

```
   CCONV TOLOCALRATE "Act xchg";
CALC ALL;
```

注： 通貨パーティション次元を使用しておらず、essbase.cfg ファイル内の CCTRACK 構成設定が TRUE ではない場合、FIX コマンドは使用できません。

ローカル値と換算値の保持

現地通貨の値と換算後の値をデータベースに保持できます。メイン・データベース内に通貨パーティション次元を作成して、現地通貨の値と変換後の値を保管するメンバーを定義する必要があります(209 ページの「メイン・データベース」を参照)。通貨パーティション次元には、現地通貨の値を保管するパーティションと換算後の値を保管するパーティションが 1 つずつあります。

➤ 換算後の値を保管するパーティションに現地通貨のデータをコピーして計算を行う計算スクリプトを作成するには:

1. **DATACOPY コマンドを使用して、ローカル・データ・パーティションから換算データ・パーティションにデータをコピーします。**
2. **FIX コマンドを使用して換算データ・パーティションのみを計算し、CCONV コマンドを使用してそのデータを換算します。**

   注 : 通貨パーティション次元を使用する場合は、この次元のメンバーに対して FIX コマンドを適用して、CCONV コマンドを使用する必要があります。

3. **CALC コマンドを使用して、データベースを再計算します。**

次の例は、Sample.Interntl データベースと対応する Sample.Xchgrate 通貨データベースに基づいています。図 47 に、Sample.Interntl データベースの通貨パーティションを示します。

図 47　ローカル・データと換算データの通貨換算の計算

次の計算スクリプトでは、Actual、Budget および Actual @ Bud Xchg の各データ値に対して、3 種類の通貨換算が実行されます:

```
   /* Copy data from the local partition to the master partition (for converted
values) */

DATACOPY Act TO Actual;
DATACOPY Bud TO Budget;

/* Convert the Actual data values using the "Act xchg" rate */

FIX(Actual)
 CCONV "Act xchg"->US$;
ENDFIX

* Convert the Budget data values using the "Bud xchg" rate */

FIX(Budget)
```

```
 CCONV "Bud xchg"->US$;
ENDFIX

/* Convert the "Actual @ Bud XChg" data values using the "Bud xchg" rate */

FIX("Actual @ Bud XChg")
 CCONV "Bud xchg"->US$;
ENDFIX

/* Recalculate the database */

CALC ALL;
CALC TWOPASS;
```

次の計算スクリプトでは、Actual と Budget の各値が、元値である現地通貨値に換算されます:

```
   FIX(Actual)
 CCONV TOLOCALRATE "Act xchg";
ENDFIX

FIX(Budget)
 CCONV TOLOCALRATE "Bud xchg";
ENDFIX
CALC ALL;
```

注： CCONV コマンドを使用して通貨換算を実行すると、換算後のデータ・ブロックは高機能計算には適さないデータとしてマークされます。このため、データベースの再計算時に、換算済のすべてのブロックが Essbase で再計算されます。

データベースの計算

CALC ALL コマンドを実行して換算後にデータベースの集計を行うと、換算後の基準レートのパーティション内に有効な合計レベルのデータが生成されます。一方、現地通貨のレートのパーティションに含まれる現地通貨の値の集計は無効です。無効な集計データが生成されるのを防ぐには、定義済の同じ通貨を使用する親への集計のみを行う SET UPTOLOCAL 計算コマンドを使用します。たとえば、米国(U.S.)内のすべての都市では、通貨単位としてドルが使用されます。したがって、U.S.のすべての子は U.S.に集計されます。ただし、この集計は国レベルで停止します。それは、北米圏内に他の通貨を使用する国があるからです。

レポート・スクリプトでの通貨換算

レポート・スクリプト内で、出力通貨と通貨タイプを設定する CURRENCY コマンドを使用して通貨換算を行うことはできません。レポート・ライター・コマンドの構文および定義については、『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： Essbase では、複数の透過データベースでの同時通貨換算はサポートされていません。別々の換算処理で 2 つの透過パーティション・データベースが計算された場合、レポート内では通貨換算を実行できません。

次のサンプル・レポートには、ペセタ(Peseta)の 1 月の為替レートを使用した、第 1 四半期のコーラの予算売上高が含まれます。

```
   Illinois Sales Budget
    Jan    Feb    Mar
    ========  ========  ========
100-10  3     3     3
100-20  2     2     2
100-30  #MISSING  #MISSING  #MISSING
100    5     5     5

     Currency: Jan->Peseta->Act xchg
     Currency: Jan->Peseta->Act xchg

Illinois Sales Budget
    Jan    Feb    Mar
    ========  ========  ========
100-10  3     3     3
100-20  2     2     2
100-30  #MISSING  #MISSING  #MISSING
100    5     5     5
```

次のスクリプトを使用して、サンプル通貨換算レポートを作成します:

```
   <Page (Market, Measures, Scenario)
{SupCurHeading}
Illinois Sales Budget
 <Column (Year)
 <children Qtr1
<Currency "Jan->Peseta->Act xchg"
<Ichildren Colas
 !
{CurHeading}
Illinois Sales Budget
 <Column (Year)
 <children Qtr1
 !
```

通貨換算の追跡

換算済の通貨パーティションと換算に使用された為替レートを Essbase で追跡するかどうかを制御するには、essbase.cfg ファイル内の CCTRACK 構成設定を使用します。通貨換算の追跡には、次の利点があります:

- レポート・ライターによるレポート時の換算が可能

- CONV TOLOCALRATE コマンドを使用して、換算済通貨を元の現地通貨に換算可能
- 通貨計算時の誤った再換算による不正確なデータの発生を防止

デフォルトでは CCTRACK はオンになっています。Essbase では、どの通貨パーティションが換算済で、どの通貨パーティションが換算されていないかが追跡されます。追跡は通貨パーティション・レベルで行われます: たとえば 2 つのパーティションからなるデータベースの場合、換算済と未換算の 2 つのフラグを設定できます。Essbase では、パーティション内のメンバーの組合せに対してフラグは保管されません。

通貨パーティションを使用していて、`essbase.cfg` 内で CCTRACK が TRUE に設定されている場合(デフォルト)、単一のパーティション・メンバーに対して FIX を実行する必要があります。複数のメンバーに対して FIX を実行できません。これは、CCTRACK が通貨パーティション・メンバー・レベルで機能する設定であり、この通貨パーティション・メンバーに関連付けられているすべてのデータが換算済または未換算とマークされるからです。たとえば、Sample.Basic データベースでは、次の設定が有効になります:

```
   FIX(Actual)
 CCONV "Act xchg"->US$;
ENDFIX
```

たとえば、Sample.Basic データベースで FIX コマンドを使用してメンバー Jan と Feb の実績値のみを換算できるようにしてしまうと、データベースの同じ通貨パーティションの中に換算済のデータとそうでないデータが混在することになり、データの一貫性が失われます。

CCTRACK をオフにする理由

通貨パーティション間の通貨データの換算効率を向上させるため、CCTRACK をオフにしてみてください。たとえば、当月のデータを現地通貨のパーティションにロードする場合、DATACOPY コマンドを使用して更新データが含まれる通貨パーティション全体をコピーしてから、通貨パーティションに対して換算を実行します。

注： 現地通貨のパーティションには常に部分データをロードするものとし、DATACOPY を使用して通貨パーティション全体を換算済パーティションにコピーしてから、通貨換算を実行します。換算済パーティションに直接データをアップロードすると、正しい結果が得られません。

CCTRACK をオフにする方法

CCTRACK は、次の 3 通りの方法でオフにできます:

- 計算スクリプト内で SET CCTRACKCALC ON|OFF コマンドを使用すると、CCTRACK が一時的にオフになります。このコマンドを計算時に使用すると、通貨換算時の柔軟性と効率が向上します。

- CLEARCCTRACK 計算コマンドでは、CCTRACK によって作成された内部為替レート・テーブルを消去できます。このコマンドを FIX ステートメント内で使用すると、通貨パーティションの為替レートが消去されます。このコマンドをデータ・ロード後に使用すると、将来の通貨換算計算のために為替レート・テーブルをリセットできます。
- `essbase.cfg` ファイル内で CCTRACK を FALSE に設定します。CCTRACK を False に設定すると、追跡システムがオフになり、次のような結果になります:
    - CCONV コマンドにより、データは未換算データ(現地通貨)と見なされます。同じデータに対して、誤って CCONV コマンドを重複実行した場合、正しい結果データが得られません。
    - 同様に、通貨レポート・オプションでもデータは未換算データ(現地通貨)と見なされます。データベース内のデータがすでに換算済である場合、レポート時に再換算が行われるため、正しい結果データが得られません。
    - 通貨換算での FIX コマンドおよび DATACOPY コマンドの使用に対する制限は、適用されません。

注： 通貨換算を実行する際は、換算対象のデータが同時に別のユーザーのアクティビティ(計算、データ・ロード、同じ通貨パーティションに対する通貨換算など)によって更新中でないことを確認してください。換算対象のデータに対するアクティビティが同時に発生した場合、不正な結果になる可能性があります。この場合、Essbase からの警告メッセージは表示されません。

通貨換算のトラブルシューティング

Oracle Essbase Administration Services Online Help の「通貨換算のトラブルシューティング」を参照してください。

15 パーティション・アプリケーションの設計

この章の内容

Essbase のパーティション機能について

パーティションは、データベース内にある他のデータベースと共有の領域です。Essbase パーティション・アプリケーションは、複数のサーバー、プロセッサまたはコンピュータで使用できます。

パーティション・タイプ

表 33 は、Essbase でサポートされているパーティション・タイプの一覧です:

表 33　パーティション・タイプ

パーティション・タイプ	説明	適用先
複製	データ・ターゲットに保管されているデータ・ソースの一部のコピー。 231 ページの「複製パーティション」を参照。	ブロック・ストレージ・データベース 集約ストレージ・データベース
透過	ユーザーは、データ・ソース内のデータに、まるでこのデータがデータ・ターゲット内に保管されているかのようにしてアクセスできる。しかし実際のデータは、データ・ソース(別のアプリケーションまたは Essbase データベース)、または別の Essbase サーバー上に保管されている。 235 ページの「透過パーティション」を参照。	ブロック・ストレージ・データベース 集約ストレージ・データベース

パーティション・タイプ	説明	適用先
リンク	一方のデータベース内のセルからもう一方のデータベース内のセルにユーザーを送信する。リンク・パーティションを使用することで、データに対して異なる視点が提供される。 242 ページの「リンク・パーティション」を参照。	ブロック・ストレージ・データベース 集約ストレージ・データベース

表 34 は、使用するパーティションのタイプを選択するときに役立つ情報を示しています:

表 34　各パーティション・タイプでサポートされている機能

機能	複製	透過	リンク
最新データ		x	x
ネットワーク・トラフィックの軽減	x		x
ディスク・スペースの削減		x	x
計算速度の向上	x		
より小規模なデータベース		x	x
クエリー速度の向上	x		x
エンド・ユーザーに表示しない	x	x	
異なる次元を持つデータベースへのアクセス			x
リカバリの容易さ	x		
同期化の必要性が小さい			x
属性に基づいてデータをクエリーする機能		x	x
分散型 OLAP 対応以外のフロントエンド・ツールを使用する機能	x	x	
更新と計算を頻繁に実行することが容易かどうか		x	
データ・ターゲット側でデータを更新する機能		x	x
異なるコンテキストでのデータの表示			x
バッチ更新や簡単な集約の実行	x		

パーティションの各部分

図 48 と表 35 に示すように、パーティションには次の部分があります。

図 48　パーティションの各部分

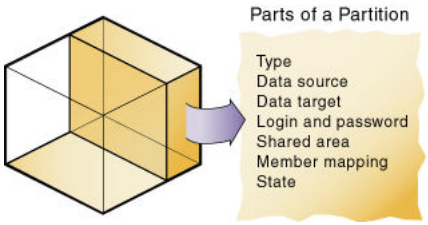


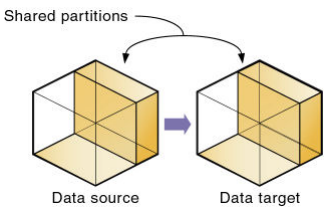
表 35　パーティションの各部分

部分	説明
パーティション・タイプ	パーティション・タイプが、複製、透過、リンクのどれであるかを示すフラグです。
データ・ソース情報	データ・ソースのサーバー、アプリケーションおよびデータベースの名前です。
データ・ターゲット情報	データ・ターゲットのサーバー、アプリケーションおよびデータベースの名前です。
ログインとパスワード	データ・ソースとデータ・ターゲットのログインおよびパスワード情報です。この情報は、2 つのデータベース間の内部要求で、管理操作およびエンドユーザー操作を実行するために使用されます。
共有領域	データ・ソースとデータ・ターゲットによって共有されている複数の領域の定義です。データベース内の隣接しない複数の部分を共有させるには、単一のパーティション内に複数の領域を定義します。この情報により、データ・ソースとデータ・ターゲットのどの部分が共有されるかが決定します。Essbase では、データ・ターゲットに適切なデータを入力して、共有領域のアウトラインの同期を保持できます。
メンバーのマッピング情報	データ・ソース内のメンバーをデータ・ターゲット内のメンバーにマッピングする方法の説明です。データ・ターゲットとデータ・ソースが、互いのメンバーや次元に異なる名前を使用している場合、Essbase ではこの情報を使用して、データ・ターゲットにデータを渡す方法が決定されます。
パーティションの状態	パーティションが最新状態になっているかどうかを確認したり、パーティションが最後に更新された日時を確認するための情報です。

データ・ソースとデータ・ターゲット

パーティション・データベースには、データ・ソース(データのプライマリ・サイト)とデータ・ターゲット(データのセカンダリ・サイト)が、それぞれ 1 つ以上含まれています。1 つのデータベースが一方のパーティションのデータ・ソース、もう一方のパーティションのデータ・ターゲットとして機能します。パーティションを定義すると、データ・ソース内のセルがデータ・ターゲット内の対応するセルにマッピングされます。

図 49　データ・ソースとデータ・ターゲット



Essbase データベースには、複数のパーティションと、他の Essbase データベースとは共有されないデータが含まれます。パーティションは、次のデータベース間に定義できます:

- 異なるアプリケーション内の異なるデータベース間。ただし、各データベースが共通の言語および Unicode 関連モードを使用している必要があります。

  アプリケーションは、同一コンピュータ上または異なるコンピュータ上に配置できます。

- 単一のブロック・ストレージ・アプリケーション内の異なるデータベース間。

  この定義はお薦めしません。各データベースが別々のアプリケーション内に存在する場合、データベースのパーティション化による利点をフルに実現できるからです。

同じ 2 つのデータベース間に各タイプのパーティションを 1 つずつのみ定義できます。たとえば、Sampeast.East データベースと Samppart.Company データベースの間には、複製パーティションを 1 つのみ作成できます。一方、East データベースや Company データベースには、他のデータベースに接続する複製パーティションを多数含めることができます。

1 つのデータベースが複数のパーティションのデータ・ソースまたはデータ・ターゲットとして機能します。多数のデータベース間でデータを共有させるには、複数のパーティションを作成します。図 50 に示すように、これらのパーティションのデータ・ソースは共通ですが、データ・ターゲットは異なります:

図 50　複数のターゲットで共有されるデータ

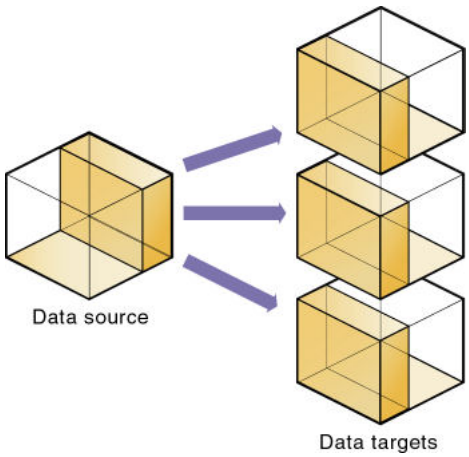


表 36 に、パーティション・タイプ別に、サポートされているデータ・ターゲットおよびデータ・ソースとして、ブロック・ストレージ・データベースと集約ストレージ・データベースの組合せを記載します:

表 36　パーティション・タイプ別サポート対象のデータ・ソースとデータ・ターゲットの組合せ

ソース	ターゲット	複製	透過	リンク
ブロック・ストレージ	ブロック・ストレージ	はい	はい	はい
集約ストレージ	ブロック・ストレージ	いいえ	はい	はい

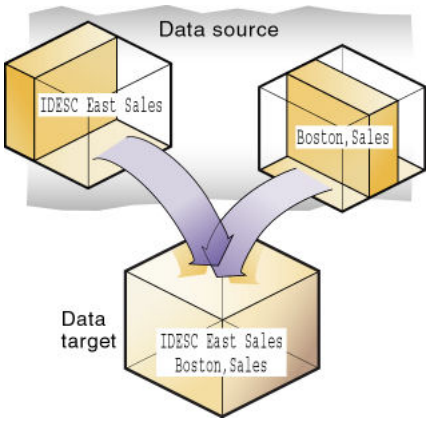
ソース	ターゲット	複製	透過	リンク
集約ストレージ	集約ストレージ	いいえ	はい	はい
ブロック・ストレージ	集約ストレージ	はい	はい	はい

オーバーラップ・パーティション

複数のデータベース内の類似データが、単一パーティション内の単一データ・ターゲットのデータ・ソースになっている場合、このパーティションはオーバーラップ・パーティションになります。

たとえば、データベース 1 の IDESC East, Sales とデータベース 2 の Boston, Sales は、データベース 3 内の IDESC East, Sales と Boston, Sales にマッピングされます。Boston は East 次元のメンバーなので、データベース 1 とデータベース 2 からデータベース 3 にマッピングされた Boston のデータがオーバーラップします。このオーバーラップの結果が、図 51 に示すようなオーバーラップ・パーティションになります:

図 51　オーバーラップ・パーティション



オーバーラップ・パーティションは、リンク・パーティション内では許可されますが、複製パーティションや透過パーティション内では許可されないため、検証中にエラー・メッセージが生成されます。

パーティション定義での代替変数

パーティション定義内で代替変数を使用することにより、そのときどきによって異なるメンバーに基づいたパーティション定義にできます。代替変数は、定期的に変更される情報のグローバル・プレースホルダとして機能します。データベース・マネージャは、各変数に割り当てられた値をいつでも変更できます。たとえば、Curmonth という名前の代替変数を定義して、この代替変数の値を年間の各月のメンバー名 Jan、Feb、Mar などに変更できます。この例では、代替変数を使用することでパーティション・サイズが縮小されます。これは、ある月のデータにアクセスする場合、パーティション定義領域にすべての月を含める必要がないからです。

領域定義またはマッピング指定に代替変数を指定するには、「テキスト・エディタの使用」または「インライン編集の使用」オプションを使用します。&Month のように、代替変数名の先頭に&を挿入します。Essbase では、パーティションの確認時に代替値が使用されます。パーティション・データを使用するプロセスを実行すると、Essbase で代替変数名が値に解決されます。代替変数名を表示するには、パーティション定義を表示します。118 ページの「代替変数の使用」を参照してください。

注： パーティション定義内で代替変数を使用する場合、アプリケーションを再起動するまで変数値はパーティション定義内で固定されたままです。たとえば、&Curmonth の値が Jan で、Feb に値を変更する場合、パーティション定義内の値はアプリケーションを再起動するまで Jan のままです。

パーティションにおける属性

ブロック・ストレージ・データベースでは、属性関数を使用して属性値によるパーティション化ができますが、属性次元のパーティション化はできません。各メンバーの特性に従って次元メンバーにアクセスするには、属性値を使用してデータベースをパーティション化します。

たとえば、Sample.Basic データベースでは、Pkg Type 属性次元をパーティション化できません。そのかわりに、Pkg Type 次元の Bottle メンバーと Can メンバー(またはそのどちらか一方)に関連付けられた製品次元のメンバーをすべて含むパーティションを作成できます。Can に関連付けられたメンバーを含むパーティションを作成した場合、缶で出荷される製品メンバー(100-10、100-20 および 300-30)のデータにのみアクセスできます。

注： ブロック・ストレージ・データベースの属性メンバーのデータを取得しても、データが存在しないという結果になる場合があります。表 18 の「存在しない保管ブロックにおける、密の動的計算メンバー」のエントリを参照してください。

@ATTRIBUTE コマンドと@WITHATTR コマンドを使用してパーティションを定義できます。

たとえば、Caffeinated 属性次元に関連付けられた製品次元のすべてのメンバーのデータを抽出するには、@ATTRIBUTE (Caffeinated)などのパーティションを作成できます。しかし、Caffeinated 属性次元はパーティション化できません。

前の例を基にすると、次のパーティションは正しくなります:

```
   Source           Target
@ATTRIBUTE(Caffeinated)  @ATTRIBUTE(Caffeinated)
```

次のパーティションは無効です:

```
   Source      Target
```

```
Caffeinated   Caffeinated
```

これらのコマンドの詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

また、第 10 章「属性の操作」も参照してください。

バージョンとエンコード方式の考慮事項

バージョン: 次のパーティション・タイプについて、パーティションの両端(ソースとターゲット)の Essbase サーバーのリリース・レベルが同じである必要があります:

- 複製
- 透過
- リンク

エンコード方式: 次のパーティション・タイプについて、パーティションの両端のアプリケーション・モード(Unicode モードまたは非 Unicode モード)が同じである必要があります:

- 複製
- 透過

パーティションの設計に関する考慮事項

この項の情報を利用して、慎重にパーティションを設計してから、パーティションを実装してください。

パーティション化の利点

パーティション化には、次の利点があります:

- ブロック・ストレージ・データベースの場合、複数のデータベースにわたるデータの同期

  Essbase により、あるパーティション内のデータ値に加えられた変更が追跡され、関連パーティション内のデータ値を更新するためのツールが提供されます。
- 複数のデータベースにわたるアウトラインの同期(集約ストレージ・データベースが透過パーティションのターゲットである場合を除く)

  Essbase により、パーティション化されたデータベースのアウトラインに加えられた変更が追跡され、関連アウトラインを更新するためのツールが提供されます。
- 異なる次元のデータベース間のナビゲート機能

  ユーザーは、新規データベースをドリルするときに、より詳細なデータにドリル・ダウンできます。

パーティション化の方法

ユーザーの要件に基づいて、次の中からパーティション化の方法を選択します:

- トップダウン方式によるアプリケーションのパーティション化。

  トップダウンのパーティション化を使用して、複数のプロセッサ、サーバーまたはコンピュータにデータベースを分割します。こうすることにより、データベースの拡張性、信頼性およびパフォーマンスを改善できます。トップダウンのパーティション化で最適な結果を得るためには、パーティション化されたデータベースごとに別々のアプリケーションを作成します。

- ボトム・アップ方式によるアプリケーションのパーティション化。

  ボトムアップのパーティション化を使用して、関連する複数のデータベース間のデータ・フローを管理します。こうすることにより、データベース内のデータの品質とアクセスを改善できます。

- 基本次元に関連付けられる属性値に従った、データベースのパーティション化(基本次元は、1 つ以上の属性次元に関連付けられる標準の次元です)。

  この方式では、次元の特性(種類、サイズなど)に基づいてデータを抽出できます。

  注： 属性次元のパーティション化はできません。226 ページの「パーティションにおける属性」を参照してください。

データベースのパーティション化のガイドライン

次の情報を利用して、データベースをパーティション化するかどうかを決定します。

- データベースをパーティション化する場合:
  - データを使用者の近くに置く必要があります。
  - 単一の障害が重大な結果を招きます。
  - 新規データをロードした後、計算の実行にかなりの時間がかかるため、複数のプロセッサまたはコンピュータに計算を分散して、パフォーマンスを改善する必要があります。
  - 異なるアプリケーション・コンテキストのデータを表示する必要があります。また、ユーザーのデータベース間のナビゲート方法を制御する必要があります。
  - ソースの異なる情報を同期する必要があります。
  - 新しい事業所を追加し、事業所固有のデータベースを有効利用できるようにする予定です。
  - 他のユーザーがデータベースにアクセスしているとき、ユーザーは待機を強いられます。
  - ユーザーにリモート・ロケーションに保管されているデータへのアクセスを許可することで、ディスク・スペースを節約する必要があります。
  - データを複製していくつかの場所に置くことで、ネットワーク・トラフィックを減少させる必要があります。

- データベースのアウトラインの制御を集中管理する必要があります。
- 集約ストレージ・データベースで、クライアントのライトバック機能を使用する必要があります。

  1035 ページの「透過パーティションの使用による集約ストレージ・データベースへのライトバックの使用可能化」を参照してください。

- データベースをパーティション化しない場合:
  - ディスク・スペース、ネットワーク帯域幅および管理リソースについて懸念があります。
  - 部門レベルの予算額を総額から求めるような複雑な配分法を適用しています。
  - すべてのデータベースを常時オンラインにしておく必要があります。

    複数のタイム・ゾーンにデータベースがある場合、データベースを常にオンラインにしておくのが困難になることがあります。理由は、ユーザー負荷がピークになる時間帯がタイム・ゾーンによって異なる可能性があるからです。この問題は、リンク・パーティションや透過パーティションを使用すると悪化しますが、複製パーティションを使用すると軽減されます。
  - データベースによって、使用言語や Unicode 関連モードが異なっています。

    Essbase では、各データベースが同じ言語を使用している場合、または各データベースが同じ Unicode モードまたは非 Unicode モードを使用している場合にかぎり、データベースをパーティション化できます。

データのパーティション化のガイドライン

パーティション化されたデータベースを設計する際は、次の情報を利用して、各パーティションに含めるデータを決定します:

- どのデータベースをデータ・ソースまたはデータ・ターゲットとして使用しますか?データを所有するデータベースをデータ・ソースに指定してください。データは、このデータ・ソースで更新されます。また、ほとんどの詳細データはここに保管されます。
- データベースの他のパートに比べ、より頻繁にアクセスされるパートがありますか?
- サイト間でどのデータを共有できますか?
- 各場所で、どのくらいの粒度のデータが必要ですか?
- データはどれくらいの頻度でアクセス、更新または計算されますか?
- ディスク・スペース、CPU およびネットワーク・リソースのうち、どのリソースを使用できますか?
- どのくらいの量のデータをネットワーク経由で転送する必要がありますか?転送時間はどのくらいかかりますか?
- データは 1 箇所に保管されていますか?それとも複数の場所に保管されていますか?

- データには 1 箇所からアクセスされますか?それとも複数の場所からアクセスされますか?
- 中央からアクセスする必要がある情報が別々のデータベースに保管されていますか?関連データ・グループはどのくらい緊密ですか?

245 ページの「パーティション・データベースの設計のケース・スタディ」を参照してください。

パーティション・データベースのセキュリティ

複製パーティション、透過パーティションまたはリンク・パーティションにアクセスするユーザーは、場合によって、複数のデータベースに保管されたデータを表示する必要があります。次の項では、ユーザーが不適切なデータを表示または変更するのを防止するセキュリティの設定方法について説明します。

エンドユーザー・セキュリティの設定

正しいフィルタを使用して、必要なエンド・ユーザーを作成します。

1. データ・ターゲット側でユーザーのアカウントを作成します。

   678 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループの管理」を参照してください。

2. データ・ターゲット側で読取りフィルタと書込みフィルタを作成し、エンド・ユーザーが表示および更新可能な内容を決定します。

   第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」を参照してください。

3. 複製パーティションを作成する場合、データ・ターゲット側の複製パーティションの変更をユーザーに許可するかどうかを決定します。更新設定(更新を許可するまたは許可しない)は、ユーザーにデータの更新を許可するユーザー・フィルタを上書きします。

   Oracle Essbase Administration Services Online Help を参照してください。

   - **create replicated partition** MaxL ステートメントを使用して複製パーティションを作成するときに、**update allow** 構文を指定しない場合、複製パーティションをデフォルトで更新できません。『Oracle Essbase テクニカル・リファレンス』を参照してください。
   - 管理サービス・コンソールを使用して複製パーティションを作成すると、複製パーティションをデフォルトで更新できます。Oracle Essbase Administration Services Online Help を参照してください。

4. リンク・パーティションを作成する場合、データ・ソース側でユーザー・アカウントを作成します。リンクされたデータベースにアクセスする際、ユーザーは、場合によって複数のデータベースに接続する必要があります。

   244 ページの「ドリルとリンク・パーティション」を参照してください。

管理者のセキュリティの設定

管理アカウントは、データ・ターゲットによって要求されるデータ・ソースの読取りおよび書込み操作をすべて実行します。たとえば、エンド・ユーザーがデータ・ターゲットのデータを要求した場合、管理アカウントはこのデータを取得します。エンド・ユーザーがデータ・ターゲットのデータを更新した場合、管理アカウントはデータ・ソースにログインして、データ・ソースのデータを更新します。

管理アカウントおよびエンド・ユーザーのフィルタを作成できます。管理アカウントのフィルタを適用することにより、データ・ターゲット側のユーザーが不適切なデータを表示または更新するのを防止できます。たとえば、中央のデータベースの管理者が特定のセルへの書込みアクセスを制限した場合、各地域の管理者がエンド・ユーザーごとに正しくセキュリティを設定する手間が省けます。

正しいフィルタを使用して必要な管理ユーザーを作成します。

1. データ・ソースとデータ・ターゲットの両方に管理アカウントを作成します。

   253 ページの「ユーザー名とパスワードの設定」を参照してください。

   Essbase では、このアカウントを使用してデータ・ソースにログインし、データ取得やアウトラインの同期化操作を行います。
2. 読取りフィルタおよび書込みフィルタを作成し、管理者が表示および更新可能なデータを決定します。

   第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」を参照してください。

   - 複製パーティションの場合は、データ・ソース側で読取りフィルタを設定して、複製時に Essbase で読み取られるデータを決定し、データ・ターゲット側で書込みフィルタを設定して、複製時に Essbase で書き込まれるデータを決定します。
   - 透過パーティションの場合は、データ・ソース側で読取りフィルタを設定して、Essbase でエンド・ユーザーに対して取得されるデータを決定し、データ・ソース側で書込みフィルタを設定して、Essbase でエンド・ユーザーに対して更新されるデータを決定します。

パーティション・データベースでのバックアップと復元、トランザクション・ロギングおよび再実行機能の使用

パーティション・データベースで、Essbase のバックアップと復元、トランザクション・ロギングおよび再実行機能を使用する場合、所定のガイドラインに従う必要があります。『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

複製パーティション

複製パーティションは、データ・ターゲットに保管されているデータ・ソースの一部のコピーです。一部のユーザーは、データ・ソース内のデータにアクセスできますが、他のユーザーはデータ・ターゲット内のデータにアクセスします。

たとえば、サンプル・アプリケーション Samppart と Sampeast では、The Beverage Company (TBC)の DBA が、East データベースと Company データベース間の複製パーティションを作成しています。この複製パーティションには、Actual、Budget、Variance および Variance%が含まれます。現在、東部のユーザーは予算データをローカルに保管しています。データを実際に本社から取得する必要がないので、応答時間が短縮され、ダウンタイムの制御やローカル・データの管理によりいっそう集中できます。245 ページの「ケース・スタディ 1: 既存のデータベースのパーティション化」を参照してください。

複製パーティション内のデータに対する変更は、データ・ソースからデータ・ターゲットに渡されます。データ・ターゲット内の複製されたデータに対する変更は、データ・ソースには戻されません。ユーザーがデータ・ターゲットでデータを変更した場合、この内容は、DBA が複製パーティションを更新する際、Essbase によって上書きされます。

複製パーティションが定義されている場合、DBA は、データ・ターゲットの複製された部分のデータが更新されないように設定できます。更新設定(更新を許可するまたは許可しない)は、セキュリティ・フィルタによるアクセス設定より優先されます。また、この設定は、データのロードや計算などのバッチ操作によって履行されます。更新設定のデフォルトの動作は、MaxL または管理サービス・コンソールを使用して複製パーティションを作成するかによって異なります。230 ページの「エンドユーザー・セキュリティの設定」を参照してください。

複製パーティションは、次の目的で使用します:

- ネットワーク・アクティビティの低減
- クエリー応答時間の短縮
- 計算時間の短縮
- システム障害からのリカバリの簡便化

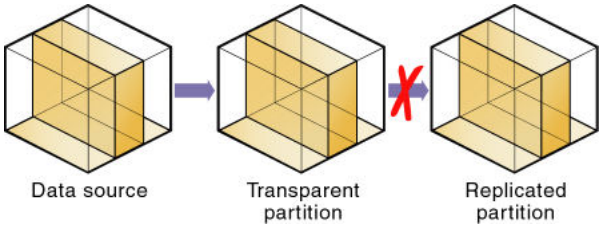
複製パーティションのルール

複製パーティションが従う必要のあるルールは、次のとおりです:

- データ・ソースおよびデータ・ターゲットのアウトラインの共有複製領域をマッピングする必要があります。ただし、共有領域がまったく同じである必要はありません。データ・ソース内の各次元およびメンバーを、データ・ターゲット内の各次元およびメンバーにマッピングする方法を Essbase に指定する必要があります。

  共有でない領域のデータ・ソースのアウトラインとデータ・ターゲットのアウトラインは、マッピングできる必要はありません。
- 複製パーティションのターゲットとして使用する領域のうち、透過パーティションをソースに持つものはありません。このため、図 52 に示すように、透過パーティションの上に複製パーティションを作成できません:

図 52　無効な複製パーティション



- 複製パーティションのデータ・ターゲット内のセルが、2 つのデータ・ソースを持つことはありません。同じパーティション内のセルは、単一のデータベースをソースにしている必要があります。複数のデータベースをソースとするセルを複製するには、データ・ソースごとに異なるパーティションを作成します。

  データ・ターゲット内のセルを、別の複製パーティションのデータ・ソースにできます。たとえば、Samppart.Company データベースに Sampeast.East データベースの複製パーティションが含まれている場合、Sampeast.East 内のセルを Sampwest.West のような 3 番目のデータベースに複製できます。

- 属性メンバーを使用して複製パーティションを定義できません。たとえば、Urban、Suburban および Rural は、市場次元に関連付けられた Market Type 属性次元のメンバーです。Urban、Suburban または Rural にパーティションを定義できません。これは、複製パーティションに、保管済のデータではなく動的なデータが含まれるからです。このため、複製パーティション内の属性のマッピングを試行した場合、エラー・メッセージが返されます。ただし、WITHATTR コマンドを使用して、属性データを複製することは可能です。

複製パーティションの長所

- データが比較的エンド・ユーザーの近くに保管されるので、データ・ターゲット内で、複製パーティションによりネットワーク・アクティビティを減少させることができます。結果として、ユーザーの取得時間が改善されます。
- すべてのユーザーは、データにより容易にアクセスできます。一部のユーザーはデータ・ソースのデータにアクセスしますが、他のユーザーはデータ・ターゲットのデータにアクセスします。
- 障害が発生しても、大きな影響はありません。データが複数の場所にあるので、1 つのデータベースで障害が発生しても、そのデータベースに接続しているユーザーが情報にアクセスできなくなるのみです。他のサイトからは、引き続きデータのアクセスおよび取得が可能です。
- ローカルの DBA は、ローカルのデータベースのダウンタイムを制御できます。たとえば、東部のユーザーは、Company データベースではなく東部固有の複製データにアクセスするので、DBA は、東部のユーザーに影響を及ぼすことなく Company データベースを停止できます。
- 各サイトには関連データのみが保持されるので、データベースのサイズを小さくできます。たとえば、東部のユーザーは、すべての地域の予算情報が含まれる会社全体の大規模なデータベースにアクセスしないで、東部の予算情報のみを複製できます。

複製パーティションの短所

- データが複数の場所に保管されるので、より多くのディスク・スペースが必要になります。
- DBA が定期的にデータをリフレッシュする必要があるので、ユーザーは最新版のデータを表示できないことがあります。

複製パーティションのパフォーマンス上の考慮事項

複製パーティションのパフォーマンスを改善するには、次のガイドラインに従ってください:

- Essbase では、動的に計算されるメンバーとそれらの子をアウトラインの検索により見つけ出して、計算の実行方法を決定する必要があるため、データ・ソース内で動的に計算されるメンバーを複製しないでください。
- データ・ソースの計算結果データを複製しないでください。かわりに、各次元について最下位の実用レベルのデータを複製し、複製の完了後にデータ・ターゲット上で計算を実行します。

  たとえば、市場次元に沿ってデータベースを複製する方法は、次のとおりです:

  - 共有領域を市場次元の最下位メンバー(East、West、South、Central など)および他の次元のレベル 0 メンバーとして定義します。
  - 複製が完了したら、データ・ターゲット側で市場の値と他の次元内の上位レベルの値を計算します。

    データ・ターゲットで計算結果データを計算できない場合もあります。その場合は、データ・ソースからデータを複製します。たとえば、データが次のいずれかの基準を満たしている場合、データ・ソースで計算結果データを計算できません:

    - 複製領域外のデータを計算するように要求する場合。
    - 該当部分のみを抽出することができない計算スクリプトをデータ・ターゲット側で計算するように要求する場合。
    - ラップトップなどの処理能力の低いコンピュータ上に複製される場合。
- 集約ストレージ・データベースがターゲット、ブロック・ストレージ・データベースがソースになっていて、2 つのアウトラインがまったく同じである場合、集約ストレージ・データベースの複製を最適化するには、次のいずれかの方法を使用します:

  - `essbase.cfg` 内の REPLICATIONASSUMEIDENTICALOUTLINE 構成設定。この設定は、サーバー、アプリケーションまたはデータベース・レベルで有効にできます。設定の構文は次のとおりです:

    ```
    REPLICATIONASSUMEIDENTICALOUTLINE [appname [dbname]] TRUE | FALSE
    ```

    `essbase.cfg` ファイルの更新の際、変更内容を有効にするには、Essbase サーバーを停止して再起動する必要があります。

- **alter database** MaxL ステートメントと **replication_assume_identical_outline** 文法。このステートメントはデータベース・レベルでのみ使用可能です。ステートメントの構文は、次のとおりです:

```
alter database
appname
.
dbname
 enable | disable replication_assume_identical_outline;
```

  alter database ステートメントを使用する場合、集約ストレージ・アプリケーションを停止して再起動する必要はありません。

最適化の方法は両方とも、ターゲットの集約ストレージ・アプリケーションのみに影響します。ソースのブロック・ストレージ・アプリケーションには影響しません。これらの方法は、ブロック・ストレージの複製には適用されません。

- 密次元に沿ったパーティション化は、疎次元に沿ったパーティション化よりも時間がかかります。Essbase では、密次元に沿ってパーティション化されたデータを複製する場合、複製操作中にデータ・ソース内の各ブロックにアクセスし、これらの各ブロックをデータ・ターゲット内に作成する必要があります。
- データ・ターゲットで動的に計算されるメンバーに、データを複製できません。Essbase では、動的計算メンバーと動的計算および保管メンバーへのデータのロードや複製は実行されません。実行時にユーザーが要求するまで、これらのメンバーにはデータが含まれないからです。複製データは、データ・ターゲットに保管されないので、Essbase では複製先の動的な密メンバーと動的な疎メンバーのどちらにも送信されません。

  パーティション全体ではなく変更されたデータ値のみを複製する方法については、267 ページの「複製パーティションの取込みまたは更新」を参照してください。

複製パーティションとポートの使用状況

複製パーティションでは、ユーザーはターゲット・データベースに接続するのみです。ターゲット・データベース上でデータが更新された場合、ソース・データベースからターゲット・データベースへのデータの複製プロセスで、ポートが 1 つ使用されます。この接続は、パーティション定義に宣言されたユーザー名(パーティション・ユーザー)に基づいています。

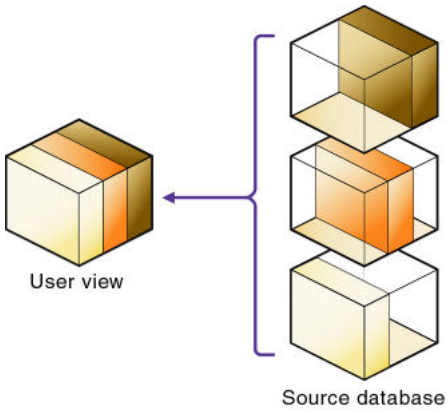
注： 短期的な複製のため、複製パーティションとポートにはほとんど問題がありません。

透過パーティション

ユーザーは、透過パーティションを使用して、リモートに保管されたデータをローカル・データベース内のデータのように操作できます。リモート・データは、デー

タ・ターゲットのユーザーから要求されるたびに、データ・ソースから取得されます。リモート・データはローカル・データベース内のデータと同様に扱えるので、ユーザーがデータの保管場所は把握する必要はありません。

図 53　透過パーティション



データはデータ・ソースから直接取得されるので、ユーザーが表示するデータは常に最新のデータです。ユーザーがデータを更新すると、更新内容が再びデータ・ソースに書き込まれます。したがって、データ・ソースの他のユーザーとデータ・ターゲットは、これらの更新にただちにアクセスできます。

透過パーティションでは、ソース・データにアクセスするユーザーが多いほど、データ・ソース側とデータ・ターゲット側の両方でパフォーマンスが低減します。

たとえば、TBC の DBA は、透過パーティションを使用して、別のコンピュータ上にあるシナリオ次元の各メンバーを計算できます。このプロセスでは、ユーザーのデータ・ビューはそのままにして、計算時間を短縮できます。245 ページの「ケース・スタディ 1: 既存のデータベースのパーティション化」を参照してください。

透過パーティションは、次の目的で使用します:

- ユーザーに最新バージョンのデータを提供する場合
- データ・ターゲット側のユーザーにデータの更新を許可する場合
- ディスク・スペースを削減する場合

透過パーティションのルール

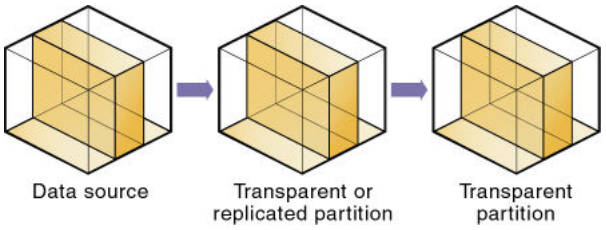
透過パーティションが従う必要のあるルールは、次のとおりです:

- データ・ソースおよびデータ・ターゲットのアウトラインの共有透過領域がまったく同じである必要はありませんが、これらに含まれる次元をマッピングする必要があります。データ・ソース内の各次元およびメンバーを、データ・ターゲット内の各次元およびメンバーにマッピングする方法を Essbase に指定する必要があります。
- 共有でない領域のデータ・ソースおよびデータ・ターゲットのアウトラインは、マッピングする必要はありません。ただし、属性の関連付けは同じである必要があります。そうでない場合、一部の取得で正しい結果が得られない可能性があります。たとえば、製品 100-10-1010 がソースの Grape Flavor 属性

と関連付けられているのに、ターゲットの Grape Flavor 属性に製品 100-10-1010 が関連付けられていない場合、New York のすべての Grape Flavor の合計売上高として、正しい結果が得られません。

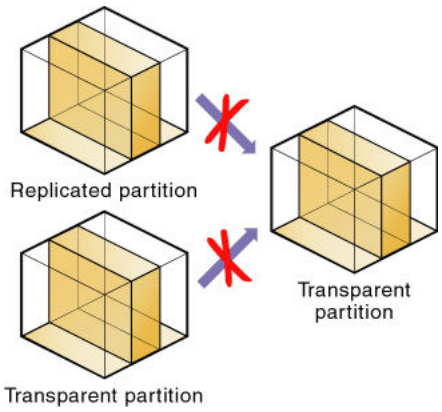
- パーティション定義には、保管済メンバーのみを含める必要があります。属性次元やメンバーを使用して透過パーティションを定義できません。たとえば、市場次元に関連付けられている Market Type 属性次元のメンバーは、Urban、Suburban および Rural です。Urban、Suburban または Rural 上にパーティションを定義できません。
- セルがデータ・ソースからターゲットの集約ストレージ・データベースにマッピングされている場合、このセルのすべての依存項目を同じパーティション定義にマッピングする必要があります。
- 複製パーティションの上に透過パーティションを作成できます。つまり、図 54 のように、複製パーティションのソースを使用して透過パーティションのターゲットを作成できるということです。

図 54　有効な透過パーティション



- 図 55 に示すように、複数のパーティションの上に透過パーティションを作成できません。これは、複数のソースからは透過パーティションのターゲットを作成できないということです。理由は、データベース内の各セルを、ローカル・ディスクまたはリモート・ディスク上の 1 箇所から取得する必要があるからです。

図 55　無効な透過パーティション



- データ・ソースとデータ・ターゲット内のメンバーに割り当てる式は、慎重に検討してください。

透過パーティションの長所

透過パーティションを使用すると、データベースの数多くの問題を解決できますが、透過パーティションが常に理想的なパーティションのタイプとは限りません。

- データを 1 つのデータベースに保管しているため、より少ないディスク・スペースで十分です。
- データ・ターゲットからアクセスされるデータは、常に最新のバージョンです。
- ユーザーがデータ・ソースのデータを更新すると、Essbase により、その変更がデータ・ターゲットに対しても適用されます。
- 個々のデータベースが小さいため、より高速に計算できます。
- データは、エンド・ユーザーやエンド・ユーザーのツールからはわからないように配布されます。
- データ・ソースまたはデータ・ターゲットのどちらからでもデータをロードできます。
- ソースに指定した集約ストレージ・データベースと、ターゲットに指定したブロック・ストレージ・データベース間に透過パーティションを作成することで、集約ストレージ・データベースのライトバック機能を有効にできます。

  1035 ページの「透過パーティションの使用による集約ストレージ・データベースへのライトバックの使用可能化」を参照してください。

透過パーティションの短所

これらの短所が無視できない場合は、かわりに複製パーティションやリンク・パーティションを使用することを検討してください。

- 透過パーティションでは、Essbase によってデータ・ソースのデータがネットワークを通ってデータ・ターゲットに転送されるので、ネットワーク・アクティビティが増大します。ネットワーク・アクティビティが増大すると、ユーザーの取得時間が長くなります。
- データ・ソースにアクセスするユーザーが増加するため、取得時間が長くなります。
- データ・ソースの障害は、データ・ソースとデータ・ターゲットの両方のユーザーに影響を及ぼします。このため、ネットワークとデータ・ソースは、データ・ソースとデータ・ターゲットのどちらのユーザーが必要とするときでも使用できる必要があります。
- 一部の管理操作は、ローカル・データにしか実行できません。たとえば、データ・ターゲットをアーカイブする場合、Essbase では、データ・ソースではなくデータ・ターゲットのみがアーカイブされます。次の管理操作は、ブロック・ストレージ・データベース内のローカル・データのみに影響を及ぼします:
  - CLEARDATA 計算コマンド
  - DATACOPY 計算コマンド
  - EXPORT コマンド

- VALIDATE コマンド
- BEGINARCHIVE コマンドおよび ENDARCHIVE コマンド
- Administration Services 内の再構築操作

- 透過パーティション上で計算を実行すると、Essbase ではローカル・データと透過の依存項目の最新の値を使用して計算が行われます。データ・ソースとデータ・ターゲットのアウトラインの差異が大きすぎて正確な計算結果が得られないので、Essbase では透過の依存項目の値は再計算されません。すべてのパーティションを計算するには、個々のパーティションに対して CALC ALL コマンドを発行し、各パーティションの新しい値を使用して、最上位で CALC ALL コマンドを実行します。

  次に例を示します:

  - データ・ターゲットのアウトラインには、East、West、South および Central のメンバーからなる市場次元が含まれます。
  - データ・ソースのアウトラインには、New York と New Jersey のメンバーからなる East 次元が含まれます。

  データ・ターゲットのアウトラインの計算を試みる場合は、East をレベル 0 メンバーと想定します。しかし、データ・ソースでは、East は New York と New Jersey の合計によって計算されます。一方、データ・ターゲットでの計算では、この情報は認識されず、データ・ソース内の New York および New Jersey に加えられた変更は反映されません。正確な計算を行うには、データ・ソース内で East を計算した後、データ・ターゲットを計算します。
- データ・ソースのメンバーに割り当てられた式は、データ・ターゲットで定義された式や集計と矛盾する計算結果を算出する場合があり、これは逆の場合も当てはまります。

透過パーティションのパフォーマンス上の考慮事項

透過パーティションのパフォーマンスを改善するには、パーティションの作成時に次のガイドラインについて検討してください:

- 透過パーティション内の密次元に沿ってパーティション化を行う場合、パフォーマンスがきわめて低くなる可能性があります。これは、密次元がデータ・ブロックの構造とコンテンツを特定するために使用されるからです。データベースをターゲットの密次元のみに沿ってパーティション化する場合、Essbase では、透過パーティション内およびブロックのローカル部分のディスク I/O でリモート・データのネットワーク呼出しを実行して、データ・ブロックを作成する必要があります。

  パフォーマンスを向上させるには、領域定義内に 1 つ以上の疎次元を含ませて、必要なブロック数を疎メンバーとの組合せに限定することを検討してください。
- 透過パーティションを次元の属性値に基づかせると、属性は疎次元に関連付けられているため、取得時間が長くなることがあります。この場合、属性のレベルより上位のレベルでパーティション化を実行すると、取得時間を短縮

できます。たとえば、Sample.Basic データベースの製品次元で、子 100-10、200-10、300-10 (レベル 0)が属性に関連付けられている場合、親 100、200、300 (レベル 1)をパーティション化すると、取得パフォーマンスが改善されます。

- データ・ターゲットからデータ・ソースにデータをロードすると、パフォーマンスが大幅に低下します。可能であれば、ローカルでデータ・ソースにデータをロードします。
- ユーザーはネットワークを使用してデータにアクセスするため、取得時間が長くなります。
- 透過パーティションがターゲットの場合、次の構成設定を使用することを検討してください:
  - データ・ソースから透過パーティション・ターゲット(ブロック・ストレージ・データベースまたは集約ストレージ・データベース)に送信される要求の場合、`essbase.cfg` ファイル内の ENABLE_DIAG_TRANSPARENT_PARTITION 構成設定を使用して、トランザクションの応答時間をロギングできます。これらのメッセージのロギングは、応答時間がきわめて長くなりトラブルシューティングが必要になった場合に役立ちます。
  - 透過パーティション・ターゲットが集約ストレージ・データベースの場合、MAX_REQUEST_GRID_SIZE および MAX_RESPONSE_GRID_SIZE 構成設定を使用して、要求グリッドと応答グリッドの最大サイズを指定できます。
- 基本次元をパーティション化すると、パフォーマンスが大幅に低下します。
- 241 ページの「透過パーティション計算のパフォーマンス上の考慮事項」を参照してください。

透過パーティションの計算

透過パーティション上で計算を実行すると、Essbase ではローカル・データと透過の依存項目の最新の値を使用して計算が実行されます。リモート・データに依存するローカル・データを計算するときは、Essbase でボトムアップの計算が実行されます。ボトムアップの計算は、ターゲット・データベース上で計算機キャッシュが適切に使用されている場合にかぎり実行できます。976 ページの「ボトムアップ計算の使用」を参照してください。

計算機キャッシュに割り当てるメモリーを増やすと、透過パーティションでの計算パフォーマンスが大幅に改善されます。計算が開始されると、アプリケーション・ログ・ファイルに、ターゲット・データベース上で計算機キャッシュが有効であるかどうかを示すメッセージが書き込まれます。ターゲット・データベース上で計算機キャッシュを使用すると、計算時にデータ・ソースから要求されるブロック数が減少します。要求されるブロック数が減少すると、ネットワーク経由でブロックを転送することにより発生するネットワーク・トラフィックが減少します。920 ページの「計算機キャッシュのサイズ設定」を参照してください。

透過パーティション計算のパフォーマンス上の考慮事項

ネットワークを使用して、データ・ターゲットが個々の依存データ・ブロックを取得して計算を実行する必要がある場合、データ・ターゲット上のデータを計算すると、パフォーマンスが大幅に低下することがあります。

Essbase で、トップダウンのメンバー式を含むデータ・ターゲットの一部に対してトップダウンの計算が実行される場合にも、透過計算のパフォーマンスは低下することがあります。データ・ターゲットにトップダウンのメンバー式が含まれない場合、Essbase では、より実行速度が速いボトムアップの計算がデータ・ターゲットに対して実行されます。

Essbase でデータ・ソースに対する計算が実行される場合、常にボトムアップの計算を実行できます。トップダウンの計算とボトムアップの計算の比較については、976 ページの「ボトムアップ計算の使用」を参照してください。

次の代替計算アプローチの使用を検討してください:

- ターゲット・パーティションの計算スクリプトでリモート・データを使用しないことがわかっている場合は、計算スクリプト内で SET REMOTECALC OFF 計算コマンドを使用して、ソース・パーティションからの取得を停止できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。
- 動的計算メンバーまたは動的計算および保管メンバーを透過データの親として使用し、取得時にただちにデータが計算されるようにします。このプロセスにより、バッチ処理時間が短縮されます。Essbase では、この計算はユーザーが要求した場合にかぎり実行されます。
- 下位レベルの透過データと上位のローカル・データ間に複製レイヤーを使用します。

次のパフォーマンス戦略を検討してください:

- 計算機キャッシュ領域内に、パーティションを完全に確保します(920 ページの「計算機キャッシュのサイズ設定」を参照)。つまり、パーティション定義に含まれるすべての疎メンバーが計算機キャッシュ内に含まれるようにする必要があります。たとえば、Sample.Basic データベース内で、パーティション定義に@IDESC(East)が含まれる場合、East のすべての子孫を計算機キャッシュに含める必要があります。
- 計算機キャッシュを使用可能にし、十分なメモリー量を計算機キャッシュに割り当てます。
- パーティションを定義するメンバーに対しては、複素数式を使用しないでください。たとえば、Sample.Basic 内で、New York または New Jersey (どちらも East の子)に対して複素数式を割り当てた場合、Essbase では強制的にトップダウンの計算方式が採用されます。976 ページの「ボトムアップ計算とトップダウン計算」を参照してください。

透過パーティションとメンバー式

メンバー式が異なる点を除いて、データ・ターゲットとデータ・ソースのアウトラインがまったく同じである場合は、パーティション定義により意図したとおりの計算結果が出ることを確認してください。

たとえば、データ・ソースとデータ・ターゲットの両方のアウトラインに、North、South、North の子、South の子をメンバーとして持つ市場次元が含まれているとします。データ・ターゲット上で、市場は、データ・ソース上の North、South およびその子メンバーのデータから計算されます。データ・ソース上のこれらのメンバーのいずれかにメンバー式が含まれる場合、データ・ターゲット上の市場の計算結果に影響が及びます。この計算結果は、メンバー式が含まれない、データ・ターゲット上の North および South メンバーから計算された市場メンバーの値とは異なっている可能性があります。

データ・ソースとデータ・ターゲット内のメンバーに割り当てる式により、意図したとおりの結果が出ることを確認してください。

透過パーティションとポートの使用状況

一意のユーザーとマシンの組合せに対して、ポート 1 つが使用されます。ユーザーが、同じユーザー名を使用して、1 台のサーバー上に複数の透過パーティションを定義した場合、1 つのポートのみが使用されます。

透過パーティション内で、ユーザー(user1)がターゲット内のソース・データにアクセスする領域をドリルする場合、user1 はパーティション定義内に宣言したユーザー名(パーティション・ユーザー)を使用して、ソース・データベースからデータにアクセスします。このアクセスにより、追加のポートが使用されます。理由は、異なるユーザー(user1 とパーティション・ユーザー)がアプリケーションに接続しているからです。

2 番目のユーザー(user2)がターゲット・データベースに接続し、ドリル・ダウンしてソース・データにアクセスする場合、user2 もパーティション定義内に宣言したユーザー名(パーティション・ユーザー)を使用します。パーティション・ユーザーはすでにソース・データベースに接続しているので、user2 が同じソース・データベースにアクセスしているかぎり、パーティション・ユーザーの追加ポートは不要です。

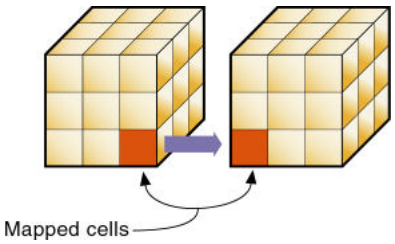
リンク・パーティション

リンク・パーティションは、データ・セルを使用して 2 つのデータベースに接続します。データ・ターゲット内のリンク・セルをクリックすると、2 番目のデータベース(データ・ソース)のドリルが実行され、そのデータが表示されます。たとえば、Smart View を使用している場合、リンク・パーティションを起動すると、リンクされたデータベースの次元を表示する新しいスプレッドシートが開きます。ここから、リンクされたデータベースの次元にドリル・ダウンできます。

複製パーティションや透過パーティションの場合と異なり、リンク・パーティションでは、データ表示がターゲット・データベースと同じ次元に制限されていません。リンク先のデータベースには、接続元のデータベースとは異なる次元が含ま

れていてもかまいません。リンク・パーティションでは、ソースからターゲットへの物理的なデータ転送は行われません。そのかわり、ターゲットのデータ・セルまたはセル範囲により、ソースのセルまたはセル範囲へのリンク・ポイントが提供されます。

図 56　リンク・パーティション



ユーザーが特権付きデータを表示するのを防ぐには、データ・ソースおよびデータ・ターゲット上にセキュリティ・フィルタを設置します。230 ページの「パーティション・データベースのセキュリティ」を参照してください。

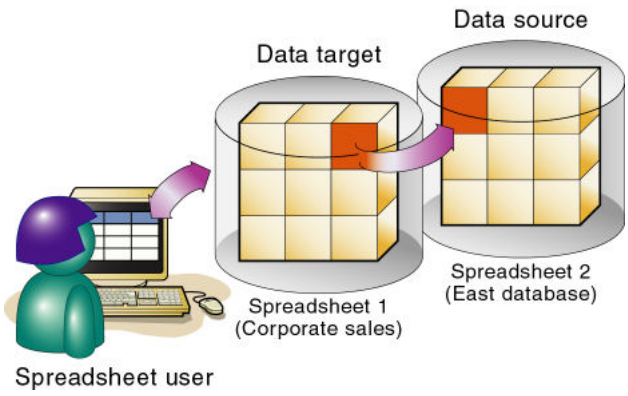
各リンク・データベースのパフォーマンスを最適化する以外に、リンク・パーティションのパフォーマンスに関する考慮事項はありません。

たとえば、TBC が大企業に成長した場合、複数の事業所が設けられる可能性があります。一部のデータ(収益と販売など)は各事業所に存在します。全社の収益と販売を一目で把握できるように、TBC はこのデータを中央のデータベースに保管できます。DBA は事業所データベースを会社のデータベースにリンクできます。249 ページの「ケース・スタディ 3: 2 つのデータベースのリンク」を参照してください。

このようなケースでは、ユーザーは次のようなタスクを実行できます:

- データ・ターゲット側のスプレッドシートに全社レベルの全体的な収益と販売を表示します。
- East など、個々の事業所をドリルして、新しいスプレッドシートを開きます。
- 新しいスプレッドシートをドリル・ダウンし、詳細データを取得します。

図 57　リンク・パーティションのソースとターゲット



リンク・パーティションの場合、ユーザーが最初に表示するスプレッドシートはデータ・ターゲットに接続され、ユーザーがドリルすることによって開くスプレッドシートは、データ・ソースに接続されます。これは、データ・ターゲットから

データ・ソースに移動する複製データベースや透過データベースの設定とは正反対です。

データベースを異なる次元に接続するには、リンク・パーティションを使用します。

リンク・パーティションの長所

- 別のコンテキストでデータを表示できます。これは、多次元の複数のデータベース間をナビゲートできるということです。
- アウトラインの共有部分がほとんどないため、データ・ソースのアウトラインとデータ・ターゲットのアウトラインを密接に同期する必要はありません。
- ユーザーは単一のデータ・セルを使用して複数のデータベースをナビゲートできます。たとえば、Accounting データベース内の Total Profit セルを、各事業部のデータベースの Profit セルにリンクできます。
- Essbase はデータ・ターゲットを介さずデータベースに直接アクセスするので、パフォーマンスが改善される可能性があります。

リンク・パーティションの短所

各データベース上でユーザー・アカウントを作成するか、目的のデータベースに対するデフォルトのアクセスを使用します(ゲスト・アカウントなどを使用)。244 ページの「ドリルとリンク・パーティション」を参照してください。

ドリルとリンク・パーティション

ユーザーがリンク・パーティション内のリンク・セルをクリックすると、スプレッドシートが開き、リンク・データベースが表示されます。このプロセスをドリルと呼びます。次の方法により、ドリル・アクセスを容易化できます:

- 各データベース上に各ユーザーのアカウントを作成します。たとえば、Mary が Company データベースと East データベースのデータにアクセスする場合、Company データベースと East データベース上に Mary 用として同じログインとパスワードを持つアカウントを作成します。

  678 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループの管理」を参照してください。
- ユーザーがターゲット・データベースにアクセスするときに使用できるデフォルト・アカウントを作成します。たとえば Company という名前のデータ・ソースと East という名前のデータ・ターゲットからデータにアクセスする場合、適切な権限を持つ East のゲスト・アカウントを作成します。リンク・パーティションを作成するときは、このゲスト・アカウントのログインとパスワードをデフォルト・ログインとして使用します。

ユーザーが、データ・ターゲットのデータをドリルするとき、Essbase では、次の手順を使用して、データ・ターゲットにアクセスするユーザーのログを取ります:

1. ユーザーがデータ・ターゲット上に同じ名前とパスワードのアカウントを持っているかどうかを確認します。持っている場合、ユーザーはそのアカウントを使用して Essbase にログインできます。
2. ユーザーがパーティションを作成したとき、データ・ターゲット上にデフォルト・アカウントを指定したかどうかを確認します。デフォルト・アカウントを指定している場合、ユーザーはそのアカウントを使用して Essbase にログインできます。
3. 新しいログインとパスワードの入力を求めるログイン・ウィンドウを開きます。ユーザーが有効なログインとパスワードを入力すると、そのアカウントを使用して Essbase にログインできます。

リンク・パーティションとポートの使用状況

Essbase では、リンク・パーティションにアクセスする際、エンド・ユーザー(user1)のログイン情報を使用してソース・データベースへの接続が試行されます。user1 がソース・データベースにアクセスできない場合は、Essbase によりリンク・パーティションのデフォルトのユーザー名とパスワードが検索されます。これらのデフォルトが指定されていない場合、user1 はソース・データベースにアクセスするためのログイン情報を入力するように求められます。ポートの使用状況は、様々なソースおよびターゲット・データベースにアクセスするために使用されているユーザー名の数(とこれらのデータベースが同一サーバー上にあるか、別々のサーバー上にあるか)によって異なります。

パーティション・データベースの設計のケース・スタディ

次の項では、データベースをパーティション化する例について説明します:

ケース・スタディ 1: 既存のデータベースのパーティション化

Sample.Basic データベースの基になっている架空の清涼飲料水製造会社 TBC は、創業時から集中管理データベースを使用していました。ところが、東部地区での事業が成長したため、このソリューションでは対応できなくなりました。東部地区に対するネットワークは、大規模なデータ・フローを処理できませんでした。ユーザーは、常に意志決定に必要なデータを待機していました。ある日、ネットワークがダウンし、東部地区のユーザーはデータにアクセスできなくなりました。

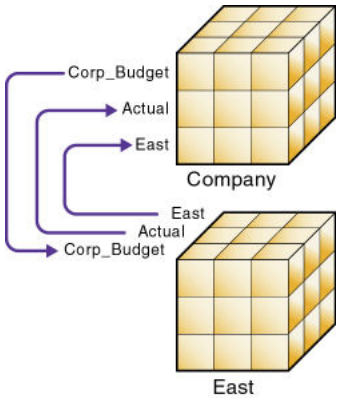
このことが契機となり、会社全体のデータベースにアクセスするのではなく、東部地区固有のデータに直接アクセスできるようにする必要があるという合意が得られました。さらに、TBC は、予算設定情報の保管場所を変更することを決定しました。会社全体の予算設定情報は本社に残しますが、東部地区の予算設定方法は東部地区のデータベースに移動することになりました。

その結果、大規模な集中管理データベースを、2 つの小規模なデータベース(Company および East)にパーティション化する要求が出されました。

この例では、Company データベースを含む Samppart サンプル・アプリケーションと、East データベースを含む Sampeast サンプル・アプリケーションを基にしています。

図 58 は、パーティション化されたデータベースのサブセットです。矢印は、データ・ソースからデータ・ターゲットへのフローを示します。Company データベースは、Corp_Budget メンバーのデータ・ソースであると同時に、East および East Actual メンバーのデータ・ターゲットです。East データベースは、East および Actual メンバーのデータ・ソースであると同時に、Corp_Budget メンバーのデータ・ターゲットです。

図 58　データ・ソースからデータ・ターゲットへのデータ・フロー



➤ この例に基づいてパーティションを作成するには:

1　**パーティション化するデータを決定します。**

Sample.Basic データベースには、5 つの標準次元(年、メジャー、製品、市場およびシナリオ)があります。

- 東部地区にその地区のデータベースのコンテンツを管理する権限を多く与えるために、市場次元の East メンバーに沿ってデータベースをパーティション化します。
- シナリオ次元の Actual メンバーと Corp_Budget メンバーに沿ってデータベースをパーティション化します。

2　**データ・ソースとデータ・ターゲットを選択します。**

- Corp_Budget については、Company をソースとして使用し、East をターゲットとして使用します。本社に会社の予算があるため、Company がソースになります。
- Eastern Region と Actual については、East がソース、Company がターゲットになります。これは、東部地区が、市場情報と実績情報を更新する必要があるためです。

3　**使用するパーティションのタイプを決定します。**

- East については、データ・ターゲット(Company)が最新データを必要とするため、透過を使用します。
- Corp_Budget については、データ・ターゲット(East)が最新データを必要とするため、透過を使用します。

- East Actual については、データ・ターゲット(Company)が最新データを必要としないため、複製を使用します。

4 次の作業を実行してパーティション・データベースを作成します。

- Sampeast アプリケーションを作成します。
- Company アウトラインを切り取り、East アウトラインに貼り付けて、East データベースを作成します。次に、余分なメンバー(South、West および Central)を削除し、East をプロモートします。
- 必要に応じて、既存のデータ・ソース、ルール・ファイル、計算スクリプト、レポート・スクリプトおよびアウトラインを編集します。
- パーティションを作成します。
- データを新規パーティションにロードします。

会社全体のデータベースをパーティション化することで、ユーザーと DBA には次のような利点があります:

- データに対してユーザーの競合がほとんどないうえ、そのデータをローカルにアクセスしているため、応答時間が以前より速くなりました
- DBA はローカルのデータベースのダウンタイムを制御できるので、メンテナンスが簡単になりました
- ユーザーは東部と会社の両方の予算に接続できるため、より多くのデータにアクセスできるようになりました
- 会社の予算と東部の予算が同期化されたため、同一のデータが使用できるようになり、データの品質が上がりました。

ケース・スタディ 2: 既存の関連データベースの接続

TBC には、Inventory、Payroll、Marketing、Sales などの複数のデータベースがあるとします。Sample.Basic データベースを表示するユーザーは、複数のデータベースでの情報の共有と、他のデータベースへの移動機能を必要としています。この場合、DBA は関連データを同期化する必要があります。すべてのデータベースを 1 つのデータベースにまとめるのは実際的ではありません。次に、その理由を示します:

- 多くのユーザーが 1 つのデータベースにアクセスすると、パフォーマンスが低下します。
- データベース管理のためにダウンタイムを作成することができません。
- データベースは集中管理されるため、誰も固有のデータを管理できなくなります。
- 多くのデータが互いに関連を持たないため、非常に疎なデータベースになります。

一方、データベースを接続した場合、次の利点があります:

- すでに完結した仕事を利用可能

- データの同期が可能

注： この例は、Essbase には含まれていません。

➤ 複数のデータベースにアクセスするには:

1 **接続するデータを決定します。まず、Inventory データベースに接続します。**

- Inventory データベースのメジャー次元から Sample.Basic データベースのメジャー次元に、Opening_Inventory メンバーと Ending_Inventory メンバーを複製します。
- Inventory データベースのメジャー次元にある Number_On_Hand、Number_Shipped および Number_Returned メンバーを、Sample.Basic データベースに複製することは避けてください。
- ユーザーが必要に応じて詳細なメジャーを表示できるように、Inventory データベースにリンクを追加します。
- Sample.Basic データベース内の Payroll、Marketing および Sales データベースのデータを含むパーティションを作成します。

2 **データ・ソースとデータ・ターゲットを選択します。**

Opening_Inventory メンバーと Ending_Inventory メンバーの場合、Inventory データベースがデータ・ソースで、Sample.Basic データベースがデータ・ターゲットになります。

3 **使用するパーティションのタイプを決定します。**

ネットワーク接続が遅いため、Opening_Inventory メンバーと Ending_Inventory メンバーの複製パーティションを使用します。

4 **Payroll、Marketing および Sales データベースを接続します。データベースごとに手順手順 1 から手順 3 を繰り返します。**

5 **次の作業を実行してパーティション・データベースを作成します:**

- 既存のデータ・ソース、ルール・ファイル、計算スクリプト、レポート・スクリプトおよびアウトラインを編集します。
- パーティションを作成します。
- 必要に応じて、データを新規パーティションにロードします。

Sample.Basic データベースがパーティション化されることで、ユーザーと DBA には次のような利点があります:

- DBA はローカルのデータベースのダウンタイムを制御できるので、メンテナンスが簡単になりました。
- ユーザーは新しいデータベースにリンクできるため、より多くのデータにアクセスできるようになりました。
- 複数のデータベースが同期化されたため、同一のデータが使用できるようになり、データの品質が上がりました。

ケース・スタディ 3: 2 つのデータベースのリンク

TBC には、Sample.Basic と TBC.Demo の 2 つのメイン・データベースがあるとします。2 つのデータベースのアウトラインはよく似ていますが、TBC.Demo には次の 2 つの次元が追加されています:

- Channel (製品の販売地域を説明)
- Package (製品の出荷形態を説明)

Sample.Basic の DBA は、その他のユーザーも Sample.Basic に対するチャネル情報の追加を要求していることを認識します。ただし、チャネル情報に関するデータを所有していないため、対応が困難です。かわりに、すでにチャネル情報が含まれている TBC.Demo データベースへのリンクを許可することにしました。

注 : この例は、Essbase には含まれていません。

➤ 2 つのデータベースをリンクするには:

1 **リンクするデータを決定します。**

DBA は、Sample.Basic の製品次元を TBC.Demo の製品次元にリンクすることを決定しました。

このため、ユーザーは、TBC.Demo をドリルして Channel と Package の情報を表示できるようになりました。

2 **データ・ソースとデータ・ターゲットを選択します。**

ユーザーは Sample.Basic データベースで開始しているため、データ・ターゲットとみなされます。ユーザーは TBC.Demo に移動するため、データ・ソースとみなされます。

注 : この設定は、ユーザーがデータ・ターゲットからデータ・ソースに移動する点で、複製データベースや透過データベースとは逆になります。

3 **使用するパーティションのタイプを決定します。**

データベースに異なる次元があるため、リンク・パーティションを使用します。

4 **パーティションを作成します:**

- Sample.Basic の製品メンバーから TBC.Demo の製品次元へのリンクを確立します。TBC.Demo の追加次元である Channel と Product を Sample.Basic 内の Void にマッピングする点に注意してください。255 ページの「次元数の異なるデータ・キューブのマッピング」を参照してください。
- TBC.Demo 上に、Sample.Basic から接続するユーザーに Channel 次元と Package 次元へのアクセスを提供するゲスト・アカウントを設定します。アカウントの作成に関する一般情報については、674 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループへの権限の付与」を参照してください。リンク・パーティションにアカウントを割り当てる方法については、251 ページの「パーティション・タイプの選択」を参照してください。

データベースがリンクされることで、ユーザーと DBA には次のような利点があります:

- ユーザーはより多くのデータにアクセスできます。
- Sample.Basic の DBA は、TBC.Demo を維持する必要がありません。リンクに問題がないかどうかを定期的に確認するだけで済みます。

16 パーティションの作成および管理

この章の内容

パーティションの作成プロセス

パーティションを作成すると、パーティション内の各データベースでは、パーティション定義ファイルを使用して、パーティションのデータ・ソース、データ・ターゲット、共有する領域などパーティションに関するすべての情報が記録されるようになります。パーティションを作成するには、データベース・マネージャ以上の権限が必要です。

パーティションの作成が完了したら、パーティションが含まれるデータベースをロードし、計算します。パーティションのロードと計算を実行するには、既存のルール・ファイルと計算スクリプトの変更が必要になる可能性があります。第 17 章「データ・ロードおよび次元構築の理解」と第 22 章「Essbase データベースの計算」を参照してください。

パーティション・タイプの選択

作成するパーティションのタイプを決定します:

- 複製

- 透過
- リンク

221 ページの「パーティション・タイプ」を参照してください。

データ・ソースとデータ・ターゲットの設定

データ・ソースとデータ・ターゲットを定義します。その一環として、ソース・アプリケーションおよびデータベース、ターゲット・アプリケーションおよびデータベース、さらにそれぞれの場所を指定します。パーティション・タイプごとのサポートされているブロック・ストレージおよび集約ストレージのデータ・ソースとデータ・ターゲットの組合せについては、223 ページの「データ・ソースとデータ・ターゲット」を参照してください。

➤ データ・ソースとデータ・ターゲットを設定するには:

1 **データ・ソースおよびデータ・ターゲットのアプリケーションとデータベースの名前を指定します。**

Oracle Essbase Administration Services Online Help の「パーティションの接続情報の指定」を参照してください。

2 **データ・ソースとデータ・ターゲットが置かれている Essbase サーバーの場所を指定します。**

Oracle Essbase Administration Services Online Help の「パーティションの接続情報の指定」を参照してください。

データ・ソースやデータ・ターゲットの名前にネットワーク別名を使用する場合は、使用しているシステム上のすべてのコンピュータに別名を適用する必要があります。それ以外の場合は、完全サーバー名を使用します。

システム上のすべてのコンピュータに別名を適用するには、使用しているオペレーティング・システムの `hosts` ファイルを編集します(このためには root の権限か管理権限が必要です):

- Windows: `%WINDIR%/system32/drivers/etc/hosts`
- UNIX: `/etc/hosts`

次の構文を使用して、`hosts` ファイルにエントリを追加します:

```
IP_address

hostname
.
domainname

alias
 [
alias
]
```

別名を 1 つ指定する例は、次のとおりです:

```
172.234.23.1 myhost.mydomain abcdefg.hijk.123
```

別名を複数指定する例は、次のとおりです:

```
172.234.23.1 myhost.mydomain abcdefg.hijk.123 lmnopqrs.tuvw.456
```

注： localhost をソース・サーバーおよびターゲット・サーバーの別名として使用しないでください。

3 **オプション: データ・ソースまたはデータ・ターゲットを説明するノートを入力します。**

Oracle Essbase Administration Services Online Help の「パーティションの接続情報の指定」を参照してください。

4 **オプション: 変更できるアウトラインを指定します。**

デフォルトでは、アウトラインの同期の際、データ・ソースのアウトラインへのすべての変更内容はデータ・ターゲットのアウトラインに上書きされます。反対に、データ・ターゲットのアウトラインへの変更内容でデータ・ソースのアウトラインが上書きされるように指定することもできます。

263 ページの「アウトラインの同期化」を参照してください。

ユーザー名とパスワードの設定

データ・ソースとデータ・ターゲットで使用する Essbase のユーザー名とパスワードを指定する必要があります。データ・ソースとデータ・ターゲットで共通のユーザー名とパスワードを使用する必要があります。Essbase では、このユーザー名とパスワードを使用して、次の作業が行われます:

- 複製パーティションおよび透過パーティションのデータ・ソースとデータ・ターゲット間でデータを転送します。エンド・ユーザーが特権付きデータを表示できないように、ローカル・セキュリティ・フィルタが適用されます。
- あらゆるパーティション・タイプのデータベース・アウトラインを同期化します。

230 ページの「パーティション・データベースのセキュリティ」を参照してください。

➤ データ・ソースとデータ・ターゲットにユーザー名とパスワードを設定する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「パーティションの接続情報の指定」を参照してください。

パーティション領域の定義

パーティション内のデータ・ターゲットと共有するように、データ・ソース内の領域を定義または編集できます。領域はデータベース内のサブキューブであり、1 つ以上の領域で構成されるパーティションです。たとえば、東部地区の実績データの最下位レベルにあるすべてのメジャーを 1つの領域ととらえることができます。

複製領域を定義するときは、データ・ソースとデータ・ターゲットのセル数を揃えてください。2 つのパーティションの形状は一致している必要があります。たとえば、データ・ソース内の領域に 18 個セルがある場合、これらの値を配置するため、データ・ターゲット内の領域にも 18 個セルが必要です。属性次元のセルはセル数にはカウントされません。

パーティション領域定義内で代替変数を使用すると、異なる時間の異なるデータを共有できる柔軟性が得られます。225 ページの「パーティション定義での代替変数」を参照してください。

注： 領域定義を作成するときは、別名ではなくメンバー名を使用します。別名は Essbase では有効ですが、パーティションでは使用できません。

➤ パーティション領域を定義する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「パーティションの領域の定義」を参照してください。

メンバーのマッピング

パーティションを作成するには、Essbase で、データ・ソースのすべての共有メンバーをデータ・ターゲットのメンバーにマッピングできる必要があります。パーティションのメンテナンスの要件(特にパーティションがメンバー属性に基づいている場合)を減少させるため、データ・ソースのメンバー名とデータ・ターゲットのメンバー名を同一にすることをお薦めします。

データ・ソースとデータ・ターゲットのメンバー数が同じで、同じメンバー名を使用している場合は、Essbase で自動的にメンバーのマッピングが行われます。ユーザーは、パーティションを検証し、保存して、テストするだけで済みます。Essbase でメンバーのマッピングが自動的に行われない場合は、手動でマッピングする必要があります。

次のいずれかの方法を使用して、データ・ソースのメンバーをデータ・ターゲットのメンバーにマッピングします:

- 手動でメンバー名を入力するか、選択します。重複するメンバー名を入力するときは、修飾メンバー名を二重引用符で囲んで入力します。たとえば、`"[State].[New York]"`のように入力します。

- 外部のデータ・ファイルからメンバーのマッピングをインポートします。
- 領域固有マッピングを作成します。

注： マッピングの指定では、メンバー名の代替変数を使用できます。225 ページの「パーティション定義での代替変数」を参照してください。

メンバーをマッピングする方法については、Oracle Essbase Administration Services Online Help の「パーティションでのグローバル・マッピングの定義」を参照してください。

異なる名前を持つメンバーのマッピング

データ・ソースのアウトラインとデータ・ターゲットのアウトラインに含まれるメンバーが異なる場合や、各アウトラインに含まれるメンバーの名前が異なる場合は、データ・ソースのメンバーをデータ・ターゲットのメンバーにマッピングする必要があります。次の例では、1 番目と 2 番目のメンバー名は同じですが、3 番目のメンバー名が異なっています:

Source

Target

```
Product    Product
 Cola       Cola
Year       Year
 1998       1998
Market     Market
 East       East_Region
```

データ・ソースの East はデータ・ターゲットの East_Region に対応することがわかっているので、East を East_Region にマッピングします。すると、データ・ターゲット内の East_Region の参照がすべてデータ・ソース内の East をポイントするようになります。たとえば、データ・ソース内の Cola、1998、East のデータ値が 15 の場合、データ・ターゲット内の Cola、1998、East_Region のデータ値は 15 になります。

次元数の異なるデータ・キューブのマッピング

データ・ソースとデータ・ターゲットに含まれる次元数が異なっている場合があります。ここでは、データ・ソースのアウトラインにデータ・ターゲットのアウトラインより多くの次元が含まれている例を紹介します:

Source

Target

```
Product    Product
 Cola      Cola
Market     Market
 East      East
Year
 1999
 1998
 1997
```

年次元のメンバー 1997 は、データ・ターゲットの Void にマッピングできます。最初にデータ・ターゲットと共有するデータ・ソースの領域を定義します:

Source

Target

```
@DESCENDANTS(Market), 1997   @DESCENDANTS(Market)
```

次に、データ・ソースのメンバーをデータ・ターゲットの Void にマッピングします:

Source

Target

```
1997     Void
```

Void は自動で表示されます。Void を手動で入力するとエラーになる可能性があります。

追加の次元のメンバーが、領域定義に 1 つも含まれていない場合には、パーティションの検証を試みたときに、エラー・メッセージが表示されます。

注： 追加の次元のメンバーをマッピングすると、結果のパーティションにはマッピングされたメンバーのデータのみが反映されます。この例の場合、年次元には 1999、1998、1997 の 3 つのメンバーがあります。データ・ソースのメンバー 1997 をデータ・ターゲットにマッピングすると、結果のパーティションに 1997 年の製品データと市場データが反映されます。1998 年と 1999 年の製品および市場データは抽出されません。

次の例は、データ・ターゲットが、データ・ソースよりも多くの次元を含むケースを示しています:

Source

Target

```
Product    Product
 Cola      Cola
      Market
```

```
       East
Year     Year
 1997     1997
```

このようなケースでは、最初にデータ・ソースとデータ・ターゲットの共有領域を定義します:

Source

Target

```
@IDESCENDANTS(Product)  @IDESCENDANTS(Product), East
```

これで、データ・ターゲットの市場次元のメンバー East を、データ・ソースの Void にマッピングできます:

Source

Target

```
Void     East
```

データ・ターゲット内の市場次元のメンバー East が、ターゲット領域の定義に含まれていない場合、パーティションの検証時にエラー・メッセージが返されます。

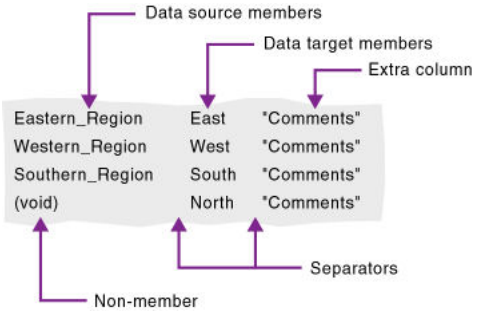
共有メンバーのマッピング

共有メンバーを使用して複製パーティションまたは透過パーティションを作成するときは、マッピングに含まれる実際のメンバー名を使用します。Essbase では、データ・ソースの実際のメンバーがマッピングされます。

メンバー・マッピングのインポート

テキスト・ファイルからメンバー・マッピングをインポートできます。マッピング・ファイルには拡張子.txt を付ける必要があります。サンプルのメンバー・ファイルには、次のすべての項目が含まれている必要があります(追加の列を除く):

図 59　メンバーのマッピングのインポート・ファイル



- データ・ソースのメンバー列(左) - データ・ソース内のメンバー名のリストです。メンバー名にスペースが含まれる場合、メンバー名を二重引用符で囲む必要があります。
- データ・ターゲットのメンバー列(中央) - データ・ターゲット内のメンバー名のリストです。メンバー名にスペースが含まれる場合、メンバー名を二重引用符で囲む必要があります。
- 非メンバー列(左) - 欠落したメンバーです。データ・ソース内の追加メンバーをデータ・ターゲット内の Void にマッピングする場合や、データ・ターゲット内の追加メンバーをデータ・ソース内の Void にマッピングする場合に使用します。
- 区切り記号(列の間) - 列を区切るタブまたはスペースです。
- 追加の列(右) - ファイルには、メンバー名を持たない追加の列を含めることができます。

➤ メンバー・マッピングをインポートする方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「パーティションのメンバー・マッピングのインポート」を参照してください。

メンバーに関連付けられた属性のマッピング

データ・ソースの属性次元とメンバーをデータ・ターゲットに正確にマッピングして、パーティションが有効であることを確認します。

注： 複製パーティション内の属性次元メンバーのマッピングはできません(232 ページの「複製パーティションのルール」を参照)。しかし、透過パーティション内とリンク・パーティション内の属性のマッピングは可能です(226 ページの「パーティションにおける属性」を参照)。

次の例のデータ・ソースのアウトラインには、メンバー 100 (Cola)を持つ製品次元が含まれています。子 100-10 と 100-20 は Caffeinated 属性次元のメンバー TRUE に、子 100-30 は Caffeinated 属性次元のメンバー FALSE に関連付けられています。

データ・ターゲットのアウトラインには、メンバー 200 (Cola)を持つ製品次元が含まれています。子 200-10 と 200-20 は With_Caffeine 属性次元のメンバー Yes に、子 200-30 は With_Caffeine 属性次元のメンバー No に関連付けられています。

最初に、データ・ソースからデータ・ターゲットへ共有される領域を定義します:

```
Source

Target

@DESCENDANTS(100)   @DESCENDANTS(200)
@DESCENDANTS(East)   @DESCENDANTS(East)
```

次に属性をマッピングします:

Source

Target

```
100-10        200-10
100-20        200-20
100-30        200-30
Caffeinated      With Caffeine
Caffeinated_True   With_Caffeine_True
Caffeinated_False   With_Caffeine_False
```

属性 Caffeinated_True を属性 With_Caffeine_No にマッピングすると、検証中にエラー・メッセージが返されます。データ・ソースのカフェイン入りコーラをデータ・ターゲットのカフェイン入りコーラに関連付ける必要があります。

属性次元または属性メンバーがデータ・ソースのアウトライン内に存在する場合、データ・ターゲットのアウトライン内には存在できません。属性次元または属性メンバーがデータ・ターゲットのアウトライン内に存在する場合、データ・ソース内のアウトライン内には存在できません。例:

Source

Target

```
Caffeinated
 True
 False
```

このケースでは、次の選択肢があります:

- データ・ターゲットのアウトライン内に Caffeinated 属性次元とそのメンバーを作成し、これらを製品次元と関連付けます。これで、これらの属性をデータ・ソースからデータ・ターゲットにマッピングできます。
- データ・ソース内の Caffeinated 属性次元を、データ・ターゲット内の Void にマッピングします。

属性の詳細は、第 10 章「属性の操作」を参照してください。パーティション内の属性の一般情報については、226 ページの「パーティションにおける属性」を参照してください。

拡張領域固有マッピングの作成

標準のメンバー・マッピングを使用して、データ・ソースのすべてのメンバーを、データ・ターゲットの対応箇所にマッピングできる場合は、拡張領域固有マッピングを実行する必要はありません。

ただし、Essbase で、より細かいレベルでメンバーのマッピングの方法を制御する必要がある場合は、領域固有マッピングを使用する必要が生じることがあります。領域固有マッピングでは、特定の領域マップのコンテキストでのみ、ある領域のメンバーを別の領域のメンバーにマッピングできます。

領域間マッピングでは、次の処理を実行できます:

- データの取得先ごとに、データを別々にマッピングします。
- データ・ソースの複数のメンバーを、データ・ターゲットの単一のメンバーにマッピングします。

Essbase では、データ・ソース内の複数のメンバーをデータ・ターゲット内の単一のメンバーにマッピングする方法を決定できません。このため、ユーザーは、データのサブセットに 1 つのマッピング・ルールを適用できるまで、そのデータの分割方法を論理的に決定する必要があります。次に、このルールを領域固有マッピングのコンテキストで使用して、メンバーのマッピングを行います。

➤ 領域固有マッピングを作成する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「パーティションでの領域固有のメンバー・マッピングの定義(オプション)」を参照してください。

例 1: 拡張領域固有マッピング

データ・ソースおよびデータ・ターゲットには、次の次元およびメンバーが含まれます:

Source

Target

```
Product    Product
  Cola       Cola
Market     Market
  East       East
Year       Year
  1998       1998
  1999       1999
       Scenario
         Actual
         Budget
```

データ・ソースにはシナリオ次元がありません。かわりに、過去のデータは実績データ、未来のデータは予測データまたは予算データと見なされます。

データ・ソースの 1998 はデータ・ターゲットの 1998, Actual に、データ・ソースの 1999 はデータ・ターゲットの 1999, Budget に対応することがわかっています。したがって、たとえばデータ・ソースの Cola, East, 1998 のデータ値が 15 の場合、データ・ターゲットの Cola, East, 1998, Actual のデータ値は 15 になります。

マッピングできるのはメンバーの組合せではなくメンバーなので、1998 を 1998, Actual に単純にマッピングはできません。領域(1998 と 1998, Actual)を定義し、この領域の領域固有マッピングルールを作成します。

また、データ・ソースには Actual メンバーと Budget メンバーがないので、これらのメンバーをデータ・ターゲットの Void にマッピングする必要があります。

例 2: 拡張領域固有マッピング

データ・ソースとデータ・ターゲットの構造が非常に異なっていても、同じタイプの情報を持つ場合、拡張領域固有マッピングを使用できます。

この方法は、たとえば、データ・ソースとデータ・ターゲットが、次の次元とメンバーを含む場合に機能します:

Source

Target

```
Market    Customer_Planning
 NY       NY_Actual
 CA       NY_Budget
        CA_Actual
        CA_Budget
Scenario
 Actual
 Budget
```

データ・ソースの NY と Actual はデータ・ターゲットの NY_Actual に、データ・ソースの NY と Budget はデータ・ターゲットの NY_Budget に対応することがわかっています。したがって、たとえばデータ・ソースの NY, Budget のデータ値が 28 の場合、データ・ターゲットの NY_Budget のデータ値は 28 になります。

マッピングできるのはメンバーの組合せではなくメンバーなので、NY, Actual を NY_Actual に単純にマッピングはできません。領域(NY と Actual、および NY_Actual)を定義し、この領域の領域固有マッピングルールを作成します。

データ・ターゲットは、NY メンバーと CA メンバーを持たないため、データ・ソースからデータ・ターゲットに向かうときに、次元性が完全であるよう、これらのメンバーをデータ・ターゲットの Void にマッピングすることが必要です。

パーティションの検証

パーティションを作成したら、使用する前に検証して、正確であることを確認します。データベース・マネージャ以上の権限が必要です。検証が完了したら、パーティション定義を保存します。必要に応じて、既存のパーティションを編集することもできます。

Essbase では、パーティション定義の検証時に、Essbase サーバー上で、データ・ソースとデータ・ターゲットについて次の項目が確認されます:

- 領域の定義が有効です(構文エラーがない)。
- 指定されたデータ・ソースのメンバーが有効であることと、データ・ターゲット内の有効なメンバーにマッピングされていること。
- すべての接続情報(すなわち、サーバー名、データベース名、アプリケーション名、ユーザー名およびパスワード情報)が正しいこと。
- リンク・パーティションの場合、提供したデフォルトのユーザー名とパスワードが正しいこと。

- 複製パーティションおよび透過パーティションの場合、複製ターゲットが複製ターゲットとオーバーラップしないこと。複製ターゲットが透過ターゲットとオーバーラップしないこと。透過ターゲットが透過ターゲットとオーバーラップしないこと。
- 複製パーティションおよび透過パーティションの場合、パーティションのセル数が、データ・ソースとデータ・ターゲット上で同じであること。
- 複製パーティションと透過パーティションの場合、領域の次元性が、データ・ソースとデータ・ターゲットで一致すること。
- 結果が完全であることを検証するには、属性値に基づく透過パーティションを検証する必要があります。結果が不完全な場合でも、Essbase では、エラー・メッセージが表示されません。

検証後、パーティションを保存すると、パーティション定義は、データ・ソース・サーバーとデータ・ターゲット・サーバーの 2 つの異なる .ddb ファイルに保存されます。

➤ パーティションを検証するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**パーティションの検証**	Oracle Essbase Administration Services Online Help
ESSCMD	VALIDATEPARTITIONDEFFILE	『Oracle Essbase テクニカル・リファレンス』
MaxL	**create partition**	『Oracle Essbase テクニカル・リファレンス』

パーティションの保存

パーティション定義の検証を完了した後、パーティション定義を次のいずれかの場所に保存できます:

- データ・ソース・サーバーとデータ・ターゲット・サーバーの両方。パーティション定義は .ddb ファイルに保管されます。
- クライアント・マシン。パーティション定義は .ddb ファイルに保管されます。

注: マッピング・エラーがあるパーティションも保存できますが、マッピング・エラーを修正しないかぎり、そのパーティションを使用して行う操作は失敗します。

➤ パーティション定義を保存する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「パーティションの保存」を参照してください。

パーティション管理のプロセス

次の項では、パーティションを管理する方法について説明します。

パーティションのテスト

パーティションをテストするには:

- Smart View またはその他のツールを使用して、データ・ターゲットを表示し、ユーザーに正しいデータが表示されていることを確認します。
- リンク・パーティションをテストする場合、Essbase によって、望みのデータベースにリンクされていることと、デフォルトのユーザー名とパスワードが正しく機能していることを確認します。

アウトラインの同期化

データベースをパーティション化する際、Essbase では、データ・ソースのアウトライン内の各次元およびメンバーを、データ・ターゲットのアウトライン内の適切な次元およびメンバーにマッピングできる必要があります。2 つのアウトラインを相互にマッピングした後、Essbase ではデータ・ソース内のデータをデータ・ターゲットから使用可能にできます。ただし、このためには、アウトラインが同期していて、パーティション定義が最新である必要があります。

一方のアウトラインに変更を加えると、2 つのアウトラインの同期が失われます。アウトラインが同期していない場合、Essbase によって複製パーティションと透過パーティションにできるかぎりの変更が加えられますが、Essbase ではデータ・ソース内のデータをデータ・ターゲット内で使用可能にできない場合があります。

Essbase では、ユーザーによるブロック・ストレージのアウトラインの変更が追跡されます。また、ブロック・ストレージのアウトラインの同期を確保するツールが提供されます。

注： Essbase では、集約ストレージのアウトラインを自動的には同期できません。ソースおよびターゲットのアウトラインに、手動で同じ内容の変更を加える必要があります。

ソースのアウトラインとターゲットのアウトラインの設定

ブロック・ストレージのアウトラインを同期化する前に、どのアウトラインがソースのアウトラインであり、どのアウトラインがターゲットのアウトラインであるかを決める必要があります。

- ソースのアウトラインは、アウトラインの変更元になるアウトラインです。
- ターゲットのアウトラインは、アウトラインの変更が適用されるアウトラインです。

デフォルトでは、ソースのアウトラインは、データ・ソースと同じデータベースのアウトラインになります。これは、アウトラインとデータの変更のフローが同じ向きになるということです。たとえば、East データベースがデータ・ソース、Company データベースがデータ・ターゲットの場合、デフォルトのソースのアウトラインは East です。

データ・ターゲットのアウトラインをソースのアウトラインとして使用することもできます。アウトラインの構造(次元、メンバーおよびプロパティ)を全社レベルで集中管理し、アウトラインのデータ値を地区レベル(たとえば East)で管理している場合、この方法をお薦めします。管理者は、Company アウトラインに変更を加え、これらの変更をアウトラインの同期時に各地区のアウトラインに適用できます。

- ソースのアウトラインの共有領域に変更を加えた場合、その内容をアウトラインの同期時にターゲットのアウトラインに適用できます。
- ターゲットのアウトラインに変更を加えた場合、その内容をアウトラインの同期時にソースのアウトラインに適用できません。ターゲットのアウトラインの変更をソースのアウトラインに適用する場合は、アウトライン・エディタで変更します。第 7 章「データベース・アウトラインの作成および変更」を参照してください。

Essbase により、ターゲットのアウトラインに可能なかぎりの変更が更新されます。Essbase がすべての変更を適用できない場合は、アプリケーション・ログでの詳細の確認を要求する警告メッセージが表示されます。アウトラインの同期に関するメッセージの先頭には OUTLINE SYNC という文字列が付加されます。817 ページの「Essbase サーバー・ログおよびアプリケーション・ログの表示」を参照してください。

➤ ソースのアウトラインを設定する方法については、252 ページの「データ・ソースとデータ・ターゲットの設定」を参照してください。

ブロック・ストレージ・アウトラインの同期化

➤ ブロック・ストレージ・アウトラインを同期化するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アウトラインの同期化	Oracle Essbase Administration Services Online Help
MaxL	**refresh outline**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	GETPARTITIONOTLCHANGES APPLYOTLCHANGEFILE RESETOTLCHANGETIME PURGEOTLCHANGEFILE	『Oracle Essbase テクニカル・リファレンス』

注： マルチバイト文字の非 Unicode モードのアウトラインの同期化では、MaxL シェルで実行される ESSCMD または MaxL ステートメントなどの非 Unicode クライアントしか使用できません。

注： アウトラインに含まれている動的計算メンバーに多数(約 100 以上)の子がある場合、そのアウトラインに対してアウトライン同期を実行できません。

変更の追跡

この項では、ソースのアウトラインを変更し、ソースのアウトラインとターゲットのアウトラインを同期化する手順について説明します。

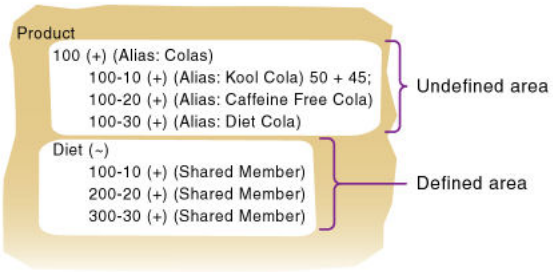
- ソースのアウトラインに変更を加えると、Essbase では次の処理が行われます:
  1. 変更を essxxxx.chg という名前の変更ログに記録します。xxxxx はパーティションの数です。ソースのアウトラインに複数のパーティションが存在する場合、Essbase によってパーティションごとに変更ログが作成されます。
  2. パーティション定義(.ddb)ファイル内にそのパーティションのアウトライン変更タイムスタンプが作成されます。.ddb ファイルには、ソースのアウトラインに対して定義されたパーティションごとに個別のタイムスタンプが記録されます。
- アウトライン・ソースから変更を戻すと、Essbase では次の処理が行われます:
  1. ターゲットのアウトラインの.ddb ファイル内にある最後に更新されたタイム・スタンプと、ソースのアウトラインのバックアップ(.dbb)ファイルの内にある最後に更新されたタイム・スタンプが比較されます。2 つのアウトラインが異なるタイム・ゾーンのサーバー上に存在している場合でも、Essbase ではソースのアウトライン上にある最後に更新された時刻を使用して、アウトラインの同期化の終了時にターゲットのタイム・スタンプが更新されます。
  2. ソースのアウトラインが前回の同期の後で変更された場合、Essbase によってソースのアウトラインの変更ログから変更内容が取得され、ターゲットのアウトラインの変更ログに適用されます。ソースのアウトライン上とターゲットのアウトライン上では、変更ログの名前が異なる場合があります。
- ターゲットのアウトラインに適用する変更を選択すると、Essbase では次の処理が行われます:
  1. ターゲットのアウトラインに変更が適用されます。
  2. ターゲットのアウトラインの.ddb ファイルにあるタイムスタンプがソースのアウトラインの時刻で更新されます。

注意 変更を適用しないように選択した場合、これらの変更は後から適用できません。

アウトライン同期中の共有メンバーの更新

パーティション領域に実際のメンバーまたは共有メンバーが 1 つ以上定義されていれば、ソースのアウトラインに含まれる実際のメンバーとその共有メンバーはターゲットのアウトラインに適用されます。図 60 に示すように、パーティション定義は@IDESC("Diet")です。親 100 と子(100-10、100-20、100-30)はパーティション領域内に定義されていません。親 Diet とその子(100-10、100-20、100-30)はパーティション領域内に定義されています。Diet の子は実際のメンバーの共有メンバーです。

図 60　共有メンバーとアウトライン同期



未定義のパーティション領域で実際のメンバーを変更(実際のメンバー 100-10 への別名の追加など)すると、その変更は、定義されたパーティション領域の共有メンバーに関連付けられているため、ターゲットのアウトラインに適用されます。

その逆も当てはまります。共有メンバーがパーティション定義領域になく、その実際のメンバーがそこに存在する場合、未定義領域での共有メンバーへの変更は、ターゲットのアウトラインに適用されます。

定義済のパーティション領域内に実際のメンバー(または共有メンバー)を 1 つ以上持たないメンバーに変更を加えた場合、その変更はターゲットのアウトラインに適用されません。たとえば、図 60 では、親 100 は未定義のパーティション領域にあり、定義済パーティション領域内に関連する共有メンバーを持たないので、親 100 への変更はターゲットのアウトラインに適用されません。

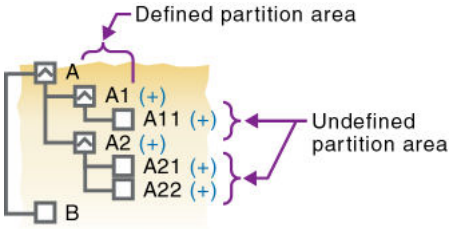
パーティション領域内に共有メンバーがある場合、その親も含めることをお薦めします。この例の場合、親 Diet の子が共有メンバーで定義済パーティション領域内にあるので、親 Diet もアウトラインに含めます。

アウトラインの同期時には、暗黙の共有メンバーも共有メンバーと同様に扱われます。パーティション領域内に実際のメンバーまたは暗黙の共有メンバーが 1 つ以上定義されていれば、ソースのアウトライン内の実際のメンバーとその暗黙の共有メンバーは、ターゲットのアウトラインに適用されます。

図 61 の例で、パーティション定義@CHILD("A")を使用する場合、A1 と A2 は定義済パーティション領域内、A11、A21 および A22 は未定義のパーティション領域内にあります。A11 (暗黙の共有メンバー)は未定義のパーティション領域にありますが、その親である A1 は定義済パーティション領域内にあるので、A11 への変更はターゲットのアウトラインに適用されます。子 A21 と A22 はパーティション領域内に定義されておらず、定義済パーティション領域内のメンバーに関連付けられていないので、これらの子メンバーへの変更はターゲットのアウトラインに適用されません。

ここでも、その逆が当てはまります。A1 がパーティション領域で定義されておらず、その暗黙的な共有メンバーが定義されている場合は、A1 に対する変更はターゲットのアウトラインに適用されます。

図 61　暗黙的な共有メンバーとアウトライン同期



複製パーティションの取込みまたは更新

管理者は、複製パーティション内のデータを定期的に更新する必要があります。複製パーティションを更新する頻度は、最新データに対するユーザーのニーズによって決まります。Essbase では、複製パーティションの更新のタイミングを決定できるように、データ・ソースの前回の変更時刻とデータ・ターゲットの前回の更新時刻が追跡されています。この情報はデータ・ソース側で保存されています。データの複製を担当するのは、データ・ソース・サイトまたはデータ・ターゲット・サイトの管理者です。

Essbase では、パーティション内のどのセルが変更されたかも追跡されています:

- 高速- 前回の複製以降にデータが変更されたセルのみ
- 低速- すべてのセル

  特定の条件では、低速の更新を使用します。たとえば、ターゲットで失われたデータをリカバリするため、すべてのセルを更新する場合などです。

次のガイドラインに従います:

- 前回の複製以降ソース・データが変更されていないときは、ユーザーがすべてのセルを更新しないかぎり、複製によってターゲット・データが更新されることはありません。
- デフォルトでは、Essbase で#MISSING のセルが複製されます。#MISSING のセルが複製されないようにするには、`essbase.cfg` ファイル内の DISABLEREPLMISSINGDATA 構成設定を使用します。『Oracle Essbase テクニカル・リファレンス』を参照してください。
- データ・ソース上のデータ・ブロックを削除した場合は、変更されたセルのみが更新されるように設定している場合でも、Essbase によりデータ・ターゲット上のすべてのデータ・セルが更新されます。次のいずれかの方法で、データ・ソース側のデータ・ブロックを削除できます:
  - 計算スクリプト内で CLEARDATA コマンドを使用
  - データ・ロード時にルール・ファイル内で「組合せの消去」を使用
  - Administration Services で CLEAR UPPER、CLEAR INPUT または RESETDB コマンドを発行
  - レベル 0 のデータまたは入力データのみを記録するデータベースを再構築
  - 疎メンバーの削除

次の複製ができます:

- データ・ソースに接続されたすべてのデータ・ターゲットの複製。

  たとえば、Sampeast.East データベースに接続されたすべてのデータ・ターゲットを複製する場合、Samppart.Company データベース内の Budget、Actual、Variance および Variance %メンバーが Essbase によって更新されます:

- データ・ターゲットに接続されたすべてのデータ・ソースからの複製。

  たとえば、Samppart.Company データベースに接続されたすべてのデータ・ソースからデータを複製する場合、Essbase により Sampeast.East データベースから Budget、Actual、Variance および Variance %メンバーが取得され、これらのデータにより Samppart.Company データベースが更新されます。

➤ 複製パーティションを更新するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データの複製	Oracle Essbase Administration Services Online Help
MaxL	**refresh replicated partition**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	GETUPDATEDREPLCELLS GETALLREPLCELLS PUTUPDATEDREPLCELLS PUTALLREPLCELLS	『Oracle Essbase テクニカル・リファレンス』

パーティションの編集および削除

パーティションを編集するときは、パーティションを作成するときと同じインタフェースを使用します。

パーティションを削除すると、Essbase により、データ・ソース・サーバーとデータ・ターゲット・サーバー上の .ddb ファイルからパーティション定義が削除されます。

➤ パーティションを編集または削除する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「「パーティションの作成」または「パーティションの編集」ウィンドウを開く」および「パーティションの削除」を参照してください。

パーティション情報の表示

➤ パーティションに関する情報を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	「パーティションの作成」または「パーティションの編集」ウィンドウを開く	Oracle Essbase Administration Services Online Help

ツール	トピック	場所
MaxL	**display partition**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	PRINTPARTITIONDEFFILE	『Oracle Essbase テクニカル・リファレンス』

パーティション化と SSL

保護モード(SSL)でパーティション化する場合は、次の考慮事項が適用されます:

- パーティションのソースとターゲットのセキュリティ・プロトコルは同じである必要があります(両方ともが SSL を使用するか、どちらも SSL を使用しない)。
- Essbase が SSL 接続を使用できるようにするには、ENABLESECUREMODE を TRUE に設定する必要があります。
- CLIENTPREFERREDMODE を SECURE に設定することを検討します。

  CLIENTPREFERREDMODE が設定されていないか、または、FALSE に設定されているが、ENABLESECUREMODE が TRUE に設定されている場合は、`:secure` を HOST-NAME 文字列に追加することにより、MaxL でパーティションを安全に作成およびリフレッシュできます。たとえば、

```
login esbuser esbpassword on “localhost:6423:secure”;
```

パーティションのトラブルシューティング

次の表では、パーティションの使用時に起こる一般的な問題をリストします。

表 37　パーティションの問題のトラブルシューティング

状態	考えられる原因	解決策
複数のデータ・ターゲットに複製するとき、複製されないことがあります。	データ・ソースといずれかのデータ・ターゲットとの接続が複製操作中に切断されました。	複製操作を再試行します。使用できないデータベースがある場合は、使用可能なデータベースにのみ複製します。
データ・ターゲットにすべての情報が届いていません。	データ・ソースのアウトラインとデータ・ターゲットのアウトラインがマッピングできなくなっています。	データ・ソースのアウトラインとデータ・ターゲットのアウトラインを同期化して、やり直してみます。
新規のパーティションまたは最近変更したパーティションを検証して保存しましたが、正しく動作しません。	パーティションに依存関係の循環がある可能性があります。データベース A がデータベース B のソースである場合、データベース B を同じスライスのデータベース A のソースにすることはできません。	パーティションの定義を編集して、依存関係の循環を取り除きます。

状態	考えられる原因	解決策
ポートが不足しています。	ポートを使用してパーティションが他のデータベースに接続しています。	ポートを新規に購入します。
パーティションにアクセスしても、接続できません。	接続先のデータベースを含むアプリケーションを、誰かが削除、名前変更または移動しています。	問題が発生しているパーティションを編集して、新しいアプリケーション名または場所を指定します。
パーティション・データベースを相互に接続できません。	ホスト名が一致していない可能性があります。hosts ファイルを使用してローカル・マシンに別名を指定したか確認してください。 252 ページの「データ・ソースとデータ・ターゲットの設定」を参照してください。	ホスト名がサーバー間で同期化されていることを確認します。
Essbase によってユーザーの編集内容が上書きされます。	パーティションを更新するたびに上書きする複製パーティション側のデータをユーザーが変更しています。	ユーザーの更新を許可しないようにパーティションを設定するか、ユーザーにデータが消えた理由を説明します。
Administration Services には、アウトラインの変更は反映されません。すなわち、変更されたアウトラインがあっても、同期されているアウトラインとしてリストされます。	• ソースのアウトラインではなくターゲットのアウトラインが変更された可能性があります。Essbase では、ソースのアウトラインにしか変更が適用されません。 • アウトラインの変更が、定義済パーティションに影響を及ぼすようになっていることを確認します。Essbase では、どのパーティションにも影響を及ぼさないアウトラインに対する変更は、適用されません。	パーティション定義を検討します。
データが混在しています。	パーティションが正しく設定されていない可能性があります。	パーティションをチェックして、必要なデータをパーティション化していることを確認します。
ソースのアウトライン内で移動したメンバーまたは次元が、同期化されたターゲットのアウトライン内に正しく反映されません。	パーティション内にない以前の次元を移動した場合、アウトラインの同期化の実行時に変更がターゲットのアウトラインに適用されないことがあります。パーティション内にない以前のメンバーを移動した場合も同様の問題が発生することがあります。	パーティション間でメンバーと次元の移動が行われないように、アウトラインの構造を設定します。それが不可能な場合は、ソースのアウトライン内で移動したメンバーや次元が反映されるようにターゲットのアウトラインを変更します。

第 I I I 部

次元の構築およびデータのロード

次元の構築およびデータのロードの内容：

17 データ・ロードおよび次元構築の理解

この章の内容

この章の情報は、ブロック・ストレージ・データベースおよび集約ストレージ・データベースに適用されます。集約ストレージ・データベースではいくつかのルール・ファイル・オプションとデータ・ソースの要件が異なるため、1052 ページの「集約ストレージ・データベースの準備」も参照してください。

はじめに

Essbase データベースには、次元、メンバーおよびデータ値が含まれます。

- データ・ロードは、Microsoft Excel スプレッドシートや SQL データベースなどのデータ・ソースから Essbase データベースにデータ値を追加するプロセスです。データ・ソースが完全にフォーマットされていない場合、データ値をロードするにはルール・ファイルが必要になります。
- 次元構築は、データ・ソースとルール・ファイルを使用して Essbase データベースのアウトラインに次元とメンバーをロードするプロセスです。アウトライン・エディタを使用して手動で次元とメンバーを追加することもできます。

データ・ロードおよび次元構築のプロセス

Essbase データベースにデータ値や次元およびメンバーをロードするには:

1. データ・ソースを設定します。

   ルール・ファイルを使用しない場合は、Essbase の外部のデータ・ソースを設定する必要があります。274 ページの「データ・ソース」を参照してください。

2. 必要な場合は、ルール・ファイルを設定します。

280 ページの「ルール・ファイル」を参照してください。

3. データ・ロードまたは次元構築を実行します。

   第 20 章「データ・ロードまたは次元構築の実行およびデバッグ」を参照してください。

データ・ソース

データ・ソースには、Essbase データベースにロードする情報が含まれています。データ・ソースに含まれる情報は、次のとおりです:

- データ値
- メンバー情報(メンバー名、メンバーの別名、式、集計プロパティなど)
- 世代名とレベル名
- 通貨名とカテゴリ
- データ・ストレージ・プロパティ
- 属性
- UDA

次の項では、各種のデータ・ソースの構成要素について説明します。

サポートされるデータ・ソース

Essbase では、次のデータ・ソース・タイプがサポートされます:

- テキストのバックアップまたは外部ソースにあるテキスト・ファイル(フラット・ファイル)
- SQL データ・ソース
- Essbase エクスポート・ファイル(エクスポート・ファイルの場合、ロードにルール・ファイルは不要)
- Microsoft Excel スプレッドシート・ファイル
- スプレッドシートの監査ログ・ファイル
- Oracle Business Intelligence Enterprise Edition

注: Essbase でスプレッドシート・ファイルを使用してデータのロードやアウトラインの構築を行うには、使用するツールにかかわらず、スプレッドシート・ファイルが Windows コンピュータ上に置かれている必要があります。UNIX コンピュータ上のスプレッドシート・ファイルや、FTP 経由で UNIX コンピュータに転送されたスプレッドシート・ファイルはサポートされません。Essbase 管理サーバーが UNIX コンピュータにインストールされている場合、クライアント側のスプレッドシート・ファイルからのデータ・ロードや次元の構築はサポートされません。

ルール・ファイル、データ・ロードおよび次元の構築のエラーを避けるには、Microsoft Excel データ・ソース・ファイルでフォーマットを削除します。たとえば、Excel で色を「自動」、「塗りつぶしなし」に設定し、太字や斜体などのフォント設定を削除します。

データ・ソース内のアイテム

図 62 に示すように、データ・ソースはレコード、フィールドおよびフィールド区切り記号からなります。

- レコードとは、関連フィールドの行を構造化したものです。
- フィールドとは、個々の値のことです。
- 区切り記号は、フィールドが完結しており、レコード内の次の文字から別のフィールドになることを示しています。

Essbase では、データ・ソースの読取りは上から下、左から右へと行われます。

図 62　レコードとフィールド

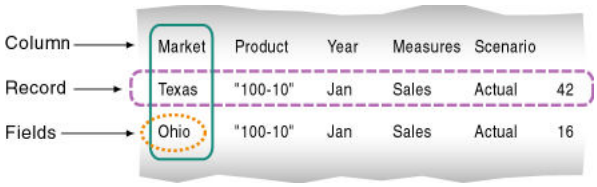
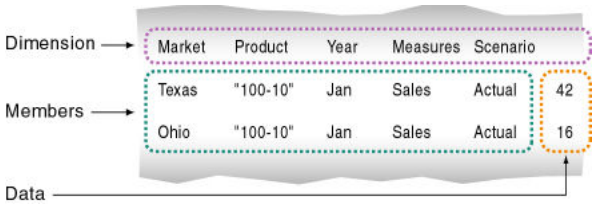


図 63 に示すように、データ・ソースには、次元フィールド、メンバー・フィールド、メンバーの組合せフィールドおよびデータ・フィールドがあります。

図 63　フィールドの種類



- 次元フィールドは、市場など、データベースの次元を識別します。次元フィールドを使用して、Essbase にデータ・ソース内の次元の順序を指示します。図 63 では、たとえば、「Market」、「Product」、「Year」、「Measures」および「Scenario」の次元フィールドがあります。「Market」列のフィールド(「Texas」など)は「Market」次元のメンバー、「Product」列のフィールド(「100-10」など)は「Product」次元のメンバーになります。次元フィールドはデータ・ソース内に設定することもできますが、通常はルール・ファイル内に定義します。
- メンバー・フィールドは、指定した次元のメンバーまたはメンバーの組合せを識別します。メンバー・フィールドを使用して、Essbase に新しいデータ値をマッピングするメンバーや、アウトラインに追加するメンバーを指示します。図 63 では、たとえば、「Texas」、「100-10」、「Jan」、「Sales」および「Actual」がメンバー・フィールドになります。

- データ・フィールドには、データベースのメンバーの交差にロードされるデータ値(数値)が含まれます。各データ値は、次元の交差にマッピングされている必要があります。図 63 では、たとえば、「42」は「Texas」、「100-10」、「Jan」、「Sales」および「Actual」の交差に対応するデータ値です。

  ヘッダーと個々のレコード内に情報を指定できます。次の例では、100 は、Jan、Actual、Cola、East および Sales の交差に対応するデータ値です。200 は、Jan、Actual、Cola、West および Sales の交差に対応するデータ値です。

```
     Jan, Actual
Cola  East  Sales  100
Cola  West  Sales  200
Cola  South  Sales  300
```

データ・フィールドは、データ・ロードに対してのみ使用されます。次元構築では、データ・フィールドは無視されます。

次の項では、データ・ソース内の各アイテムについて説明します。

有効な次元フィールド

データ・ロードでは、データ・ソースで Essbase データベース内のすべての次元が識別されない場合、ルール・ファイルで欠落した次元を識別する必要があります。たとえば、Sample.Basic データベースには年次元があります。複数の地域から月番号を持つ複数のデータ・ソースを生成する場合、月自体はデータ・ソース内に指定されていないことがあります。この場合は、データ・ソースのヘッダーまたはルール・ファイルに月を指定する必要があります。304 ページの「ヘッダー・レコードの定義」を参照してください。

次元フィールドには、有効な次元名を含める必要があります。次元構築を実行しない場合は、その次元がすでにデータベース内に存在している必要があります。次元構築を実行する場合、新しい次元名を指定できますが、この名前はルール・ファイルに指定する必要があります。

有効なメンバー・フィールド

メンバー・フィールドには、有効なメンバーの名前または別名を含めることができます。図 63 では、たとえば、「Texas」と「Ohio」は「Market」次元の有効なメンバーです。空のメンバー・フィールドは、以前のレコードのメンバー名を継承します。データ・ソースの各メンバー・フィールドとデータベースのメンバーのマッピングを Essbase に指示する必要があります。

有効なメンバー・フィールドにするには、次の基準を満たしている必要があります:

- メンバー・フィールドには、有効なメンバー名またはメンバー・プロパティを入力するか、継承させる必要があります。292 ページの「データ・ソースを使用したメンバー・プロパティの操作」を参照してください。次元構築を実行しない場合、そのメンバーはすでにアウトライン内に存在している必要があります。次元構築を実行する場合は、新しいメンバーを指定できます。

- データ・ソースまたはルール・ファイルのいずれかで、各メンバー・フィールドをどの次元にマッピングするかを指定する必要があります。
- メンバー・フィールドは、1 つのメンバー名(年次元のメンバーである Jan など)またはメンバーの組合せ(年次元およびシナリオ次元のメンバーである Jan、Actual など)にマッピングできます。
- ファイル区切り記号と同じ文字が含まれるメンバー名は二重引用符で囲む必要があります。たとえばデータ・ソースの区切り記号がスペースの場合、"New York"のようにスペースが含まれるメンバー名は二重引用符で囲みます。ルール・ファイルなしでデータ・ロードを実行する場合は、その他の文字が含まれるメンバー名も引用符で囲む必要があります。281 ページの「ルール・ファイルを必要としないデータ・ソース」を参照してください。

ルール・ファイルを使用しない場合、空の次元およびメンバー・フィールドが有効になります。Essbase では、ルール・ファイルなしでデータ・ロードを実行する際に空の次元またはメンバー・フィールドが検出された場合、その次元またはメンバーの列で最後に検出された次元またはメンバー名が使用されます。

注： データ・ロードのデータ・ソース内の各レコードの処理中、Essbase では、メンバー・フィールドに指定されたメンバーが次元フィールドに指定された次元に属しているかどうかは確認されません。Essbase によって、レコード内のメンバーの組合せで識別されるデータ・セルにデータ値がロードされます。図 63 では、たとえば、2 番目のレコードで「Jan」と「Sales」が反転した場合(「Texas, '100-10', Sales, Jan, Actual, 42」)、Essbase によって正しいデータ・セルに「42」がロードされます。ただし、次元の参照方法としてルール・ファイルが設定されているフィールドは例外です。

有効なデータ・フィールド

次元構築を実行する場合、この項はスキップします。次元構築時は、データ・フィールドは無視されます。

データ・ソースまたはルール・ファイルには、Essbase に各データ値の配置先を判断させるために十分な情報が含まれている必要があります。データ・フィールドには、データベース内の交差に入るデータ値が含まれます。図 63 では、たとえば、「42」はデータ・フィールドです。これは、1 月のテキサスにおけるコーラ(100-10)のドル建ての売上高です。

Essbase では、表 38 に示すように、データ・フィールドには間にスペースや区切り記号のない数字と修飾子、およびテキスト文字列#MI と#MISSING が入ります。

表 38　有効なデータ・フィールド修飾子

有効な修飾子	例
通貨記号: • ドル($) • ユーロ € • 円(¥)	$12 は有効な値です。 $ 12 は、ドル記号と 12 の間にスペースがあるため無効な値です。

有効な修飾子	例
負の値を示す、丸かっこで囲まれた数字	(12)
数字の前のマイナス符号。数字の後ろのマイナス符号は無効。	-12
小数点	12.3
桁数の多い数字(カンマありまたはカンマなし)	1,345,218 と 1345218 は有効な値です。
欠落値または不明な値を表す#MI または#MISSING	310 ページの「空のフィールドへのテキストの配置」を参照してください。

データ・ソースに各次元のメンバー・フィールドとデータ値を含むフィールドが 1 つ含まれる場合、このデータ値を含むフィールドをデータ・フィールドとしてルール・ファイル内に定義する必要があります。たとえば、次のデータ・ソースを Sample.Basic データベースに読み込むには、最後のフィールドをデータ・フィールドとして定義します。

```
   Jan  Cola  East  Sales  Actual  100
Feb  Cola  East  Sales  Actual  200
```

➤ データ・フィールドを定義するには、『Oracle Essbase Administration Services オンライン・ヘルプ』のデータ・フィールドとしての列の定義に関する項を参照してください。

データ・ソース内にデータ値の空白フィールドが含まれる場合、#MI または #MISSING で置き換えます。データ・フィールドに値が入っていない場合(または値が#MISSING の場合)、Essbase ではデータベース内の既存のデータ値は変更されません。Essbase によって現在の値が空の値で置き換えられることはありません。

有効な区切り記号

フィールド間は区切り記号で区切る必要があります。ルール・ファイルなしでデータをロードする場合、フィールドの区切り記号としてスペースを使用する必要があります。

ルール・ファイルを使用する場合は、区切り記号として次のいずれかを使用できます:

- タブ(デフォルト)
- スペース
- 改行
- キャリッジ・リターン
- カンマ

余分な区切り記号(ルール・ファイルを使用しない場合)

ルール・ファイルなしでロードされたデータ・ソースでは、Essbase により余分な区切り記号は無視されます。次の例では、フィールドがスペースで区切られています。Essbase ではフィールド間の余分なスペースは無視されます。

```
   East Cola Actual Jan Sales 10
East Cola Actual Feb Sales 21
East Cola Actual Mar Sales 30
```

余分な区切り記号(ルール・ファイルを使用する場合)

ルール・ファイルを使用してロードされたデータ・ソースでは、Essbase によって余分な区切り記号が空フィールドとして読み取られます。たとえば、ルール・ファイルを使用して次のファイルを Sample.Basic データベースにロードしようとした場合、失敗します。Essbase では、最初のレコードの East と Cola の間の余分なカンマが余分なフィールドとして読み取られます。このため、Essbase では Cola がフィールド 3 に入ります。しかし、次のレコードでは、Cola はフィールド 2 に入っています。Essbase ではこの場合、Cola はフィールド 3 に入る必要があると見なされるので、データ・ロードが停止します。

```
   East,,Cola,Actual,Jan,Sales,10
East,Cola,Actual,Feb,Sales,21
East,Cola,Actual,Mar,Sales,30
```

問題を解決するには、データ・ソースから余分な区切り記号を削除します。

有効なフォーマット用文字

Essbase では、データ・ソース内の一部の文字がフォーマット用文字としてのみ認識されます。Essbase では表 39 にない文字は無視されます。

表 39　有効なフォーマット用文字

フォーマット用文字	説明
==	二重の下線などを表す複数の等号
--	一重の下線などを表す複数のマイナス符号
_ _	複数のアンダースコア
==	複数の IBM PC グラフィック二重下線(ASCII 文字 205)
_ _	複数の IBM PC グラフィック一重下線(ASCII 文字 196)

無視されるフィールドは、データ・ロードまたは次元構築に影響を与えません。

たとえば、Essbase では次のデータ・ソース内の等号は無視され、その他のフィールドのみが正常にロードされます。

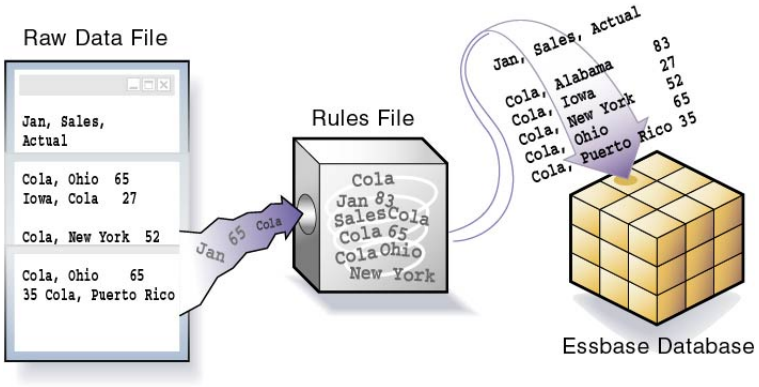
```
   East Actual "100-10"
```

```
   Sales  Marketing
   =====  =========
Jan  10    8
Feb  21    16
```

ルール・ファイル

ルールには、Essbase でのデータ・ソース処理時に、データ値または次元およびメンバーに対して実行される操作が定義されます。ルールを使用して、データ値の Essbase データベースへのマッピングや、次元およびメンバーの Essbase アウトラインへのマッピングを実行できます。

図 64　ルール・ファイルを使用したデータ・ソースのロード



ルールはルール・ファイルに保管されます。ルール・ファイルには、使用する構築方法、データ値やメンバーをソートするかランダムな並び順のままにするか、およびデータ値やメンバーをロードする前に変換する方法が定義されます。次元ごとに別々のルール・ファイルを作成することをお薦めします。

Essbase により、データ・ソースのデータ値またはメンバーが読み込まれ、ルール・ファイルに指定されたルールに基づいて変更が加えられます。さらに、変更されたデータ値がデータベースに、変更されたメンバーがアウトラインにロードされます。Essbase ではデータ・ソースは変更されません。ルール・ファイルは、同じルール・セットを必要とする任意のデータ・ソースで再利用できます。

次元構築ルール・ファイルを作成した後、次元更新プロセスを自動化する場合があります。付録 E「ESSCMD の使用」を参照してください。

ルール・ファイルが必要な場合および不要な場合

ルール・ファイルは、データ・ソースがデータベースに完全にマッピングされないか、または次のいずれかのタスクを実行する場合に必要となります:

- SQL データ・ソースからのデータのロード
- 次元の構築

- 次元およびメンバーをデータベースに追加する場合
- データベース内の既存の次元およびメンバーを変更する場合

- 次のいずれかの場合を含む、データの変更:
  - データ・ソース内のフィールドまたは文字列を無視する場合
  - フィールドを移動、結合、分割または作成して、フィールドの順序を変更する場合
  - 文字列を変更して、データ・ソース内のデータをデータベースにマッピングする場合
  - データ値をスケーリングしたり、データ値をデータ・ソース内の既存のデータ値に追加して、データ・ソース内のデータ値を変更する場合
  - 欠落している値に対してヘッダー・レコードを設定する場合
  - 無効なレコードを除外してデータ・ロードを続行する場合

データ・ロードを実行し、データ・ソースがデータベースに完全にマッピングされる場合、ルール・ファイルは不要です。281 ページの「ルール・ファイルを必要としないデータ・ソース」を参照してください。

注： ルール・ファイルを使用する場合、ルール・ファイル内の各レコードのフィールド数が一致する必要があります。321 ページの「データ・ソース内の欠落フィールドの処理」を参照してください。

ルール・ファイルを必要としないデータ・ソース

次元構築を実行する場合、この項はスキップします。

データ・ソース内のデータ値をデータベースにロードするために必要なすべての情報がデータ・ソース内に含まれている場合、このデータ・ソースを直接、フリーフォームのデータ・ロードでロードできます。

データ値を正常にロードするには、Essbase により、データ値の前に、各次元のメンバーが 1 つずつ検出される必要があります。たとえば、図 63 では、「Texas」、「100-10」、「Jan」、「Sales」および「Actual」の各メンバーとともに、データ値「42」が Essbase によってデータベースにロードされます。各次元のメンバーが指定される前に Essbase でデータ値が検出された場合、データ・ソースのロードは停止します。

完全にマッピングするには、データ・ソースに次の項目のみが含まれ、他の項目が含まれないようにします:

- 各次元の有効なメンバーを 1 つ以上。メンバー名に次のいずれかの文字が含まれる場合、メンバー名を引用符で囲む必要があります:
  - スペース
  - 数字(0-9)
  - ダッシュ(マイナス符号、ハイフン)
  - プラス符号

- アンパサンド(&)

  ルール・ファイルを使用しないでデータ・ロードを実行する場合、Essbaseにより無効なメンバー・フィールドが検出されると、データ・ロードが停止します。Essbaseでは、無効なフィールドより前の読込み済のフィールドはすべてデータベースにロードされます。つまり、データ値の一部のみがロードされることになります。833ページの「次元構築およびデータ・ロードのエラー・ログのロード」を参照してください。

- 1つ以上の有効なデータ値。277ページの「有効なデータ・フィールド」を参照してください。

  データ・ソースにデータ値の空白フィールドが含まれる場合、空白フィールドを#MIまたは#MISSINGで置き換えます。そうしないと、データ値が正常にロードされない可能性があります。

- 有効な区切り記号。278ページの「有効な区切り記号」を参照してください。

データ・ソース内のフィールドが、Essbaseに読み取られる順にフォーマットされている必要があります。レコードの最も簡単なフォーマットでは、次に示すように、各次元のメンバー1つとデータ・フィールド1つを含めます:

```
   Sales "100-10" Ohio Jan Actual 25
Sales "100-20" Ohio Jan Actual 25
Sales "100-30" Ohio Jan Actual 25
```

正常にフォーマットされていないデータ・ソースは、ロードされません。データ・ソースをテキスト・エディタで編集して、問題を修正できます。多くの修正(複数のフィールドおよびレコードの移動など)が必要な場合は、ルール・ファイルを使用してデータ・ソースをロードすることを検討してください。280ページの「ルール・ファイル」を参照してください。

次の項では、フリーフォームのデータ・ソースのより複雑なフォーマット方法について説明します。

メンバー・フィールド範囲のフォーマット

次元構築を実行する場合、この項はスキップします。

メンバー名を次元内の範囲として表現できます。たとえば、SalesとCOGSはメジャー次元内の範囲になります。メンバー名の範囲を使用して、一連の値を処理できます。

データ・ソースには、同時に複数の次元の範囲を含めることができます。次の例では、JanとFebが年次元の範囲、SalesとCOGSがメジャー次元の範囲になります。

```
Actual  Texas  Sales  COGS
     Jan  Feb  Jan  Feb
"100-10"  98   89   26   19
"100-20"  87   78   23   32
```

Sales は先頭の 2 列に対して、COGS は末尾の 2 列に対して、それぞれ定義されています。

次の項では、他のタイプの範囲について説明します。

範囲の自動設定

次元構築を実行する場合、この項はスキップします。

Essbase により、間にデータ・フィールドを挟まないで同じ次元のメンバーが複数検出された場合、その次元の範囲が設定されます。この範囲は、Essbase で同じ次元の別のメンバー名が検出されるまで有効です。同じ次元の別のメンバー名が検出されると、この範囲は Essbase によって新しいメンバー範囲で置き換えられます。

次の例には、年次元の Jan から Feb の範囲が含まれています。Essbase で Mar のような別のメンバー名が検出されるまで、この範囲は有効です。Essbase で Mar が検出されると、範囲が Jan、Feb、Mar に変わります。

```
   Texas  Sales
          Jan  Feb  Mar
Actual   “100-10"  98   89   58
     “100-20”   87   78   115
```

範囲外のデータ値の処理

次元構築を実行する場合、この項はスキップします。

Essbase によってメンバー名の範囲が検出されると、対応するデータ値の範囲が存在するものと見なされます。データ値がメンバー範囲にない場合、データ・ロードは停止します。Essbase により、無効なフィールドの前に読み込まれたすべてのデータがデータベースにロードされます。このため、部分的なデータ・ロードになります。

次の例では、データ・フィールドの数が、定義されたメンバー範囲内のメンバー・フィールドの数より多くなっています。データ・ロードは、10 データ・フィールドに達した時点で停止します。Essbase により 100 および 120 のデータ・フィールドがデータベースにロードされます。

```
   Cola  Actual  East
    Jan  Feb
Sales  100  120  10
COGS   30   34   32
```

ロードを再開する方法については、833 ページの「次元構築およびデータ・ロードのエラー・ログのロード」を参照してください。

範囲内の重複メンバーの解釈

次元構築を実行する場合、この項はスキップします。

ソース・データ内に範囲を構築して、Essbase で範囲が正しく解釈されるようにします。ある範囲内に同じメンバーが 1 回以上含まれる場合、Essbase では重複は無視されます。

次の例に示すのは、Actual、Budget、Sales、および Budget と 2 つの範囲(Actual から Budget と Sales から COGS)の重複メンバーです。Essbase では Actual、Budget、Sales および COG の重複インスタンスは無視されます(たとえば、例の 2 行目の、2 番目の Actual と 2 番目の Actual は無視されます):

```
Cola  East
   Actual  Budget  Actual  Budget
   Sales  Sales  COGS  COGS
Jan  108   110   49   50
Feb  102   120   57   60
```

最初の範囲の最初のメンバーである Actual では、Essbase により 2 番目の範囲の各メンバー(Sales と COGS)にデータ値がマッピングされます。Essbase では次に、最初の範囲の 2 番目の値である Budget でも、同様に 2 番目の範囲の各メンバーに値がマッピングされます。結果として、Essbase ではファイルは次のように解釈されます:

```
   Cola  East
   Actual     Budget
   Sales  COGS  Sales  COGS
Jan  108   110   49   50
Feb  102   120   57   60
```

複数の範囲の読取り

次元構築を実行する場合、この項はスキップします。

Essbase では、ファイルのスキャンの際、一番最後に検出されたデータ値の範囲が最初に処理されます。この例には、Actual と Budget、Sales と COGS の 2 つの範囲が含まれています。左から右、上から下へファイルが読み取られる際、Essbase では最初に Actual と Budget の範囲が検出され、次に Sales と COGS の範囲が検出されます。Sales と COGS の範囲は 2 番目に検出されるので、Essbase ではデータベースの Sales と COGS の部分にあるデータ・フィールドが最初に配置されます。

列のフォーマット

次元構築を実行する場合、この項はスキップします。

ファイルには、フィールドの列を含めることができます。Essbase では、対称列または非対称列のデータのロードがサポートされます。

対称列

次元構築を実行する場合、この項はスキップします。次元構築を実行するには、ルール・ファイルが必要です。

対称列の下には、同数のメンバーがあります。次の例では、各次元列の下に、メンバーの列が 1 つあります。たとえば、製品の下に 1 列(100-10 と 100-10)、市場の下に 1 列(Texas と Ohio)があります。

```
Product   Measures  Market   Year  Scenario
"100-10"  Sales     Texas    Jan   Actual    112
"100-10"  Sales     Ohio     Jan   Actual    145
```

Jan および Feb の下には同数のメンバーがあるため、次のファイルにおける列も対称列です。

```
                    Jan         Feb
              Actual   Budget   Actual   Budget
"100-10"  Sales  Texas  112     110     243     215
"100-10"  Sales  Ohio   145     120     81      102
```

非対称列

次元構築を実行する場合、この項はスキップします。

非対称列では、その下のメンバー数が異なっています。次の例の Jan 列と Feb 列は非対称列です。これは、Jan の下の列数が 2 列(Actual と Budget)、Feb の下の列数が 1 列(Budget)のためです:

```
              Jan    Jan    Feb
             Actual  Budget  Budget
"100-10"  Sales  Texas  112    110    243
"100-10"  Sales  Ohio   145    120    81
```

ファイル内に非対称列が含まれている場合は、各列に適切なメンバー名のラベルを付加します。

この例は、Actual と Budget に Jan ラベルが付加されているので有効です。Essbase では、両方の列が Jan にマッピングされることがわかります。

次の例は、列ラベルが不十分なので無効です。Actual 列と Budget 列に Jan ラベルを付加する必要があります。

```
              Jan         Feb
             Actual  Budget  Budget
"100-10"  Sales  Texas  112    110    243
"100-10"  Sales  Ohio   145    120    81
```

セキュリティとマルチユーザーに関する考慮事項

Essbase では、複数ユーザーが同時にデータベースの読取りおよび更新を実行できます。このためユーザーは、データベースを使用しながら、次元構築、データ・ロード、データベースの計算などを動的に実行できます。マルチユーザー環境では、第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」で説明するセキュリティ・システムを使用して、Essbase によりデータが保護されます。

- セキュリティの問題

  セキュリティ・システムにより、無許可のユーザーによるデータベースの変更が防止されます。データベースへの書込みアクセスを持つユーザーのみが、データ値のロード、またはデータベースへの次元およびメンバーの追加を実行できます。書込みアクセスはグローバルに、またはフィルタを使用して提供できます。

- マルチユーザー・データ・ロードの問題

  複数のユーザーがデータベースに接続しているときも、データ値をロードできます。Essbase では、マルチユーザーの問題を処理するため、ブロックのロック・スキームが採用されています。ユーザーがデータ値をロードすると、Essbase により次の処理が行われます:

  - ロード先ブロックがロックされます。これにより、このブロックへの書込みができなくなります。

    Essbase のトランザクション設定については、第 51 章「データの整合性の確保」を参照してください。トランザクション設定では、ロックされたブロックに対する読取り専用アクセスを他のユーザーが取得するかどうかを特定したり、ロックされたブロックが解放されるまで Essbase が待機する時間を決定します。

  - そのブロックが更新されます。

    更新が完了した時点で Essbase によってブロックが解除されるか、またはデータ・ロード全体が完了した後にブロックが解除されるかについては、862 ページの「データ・ロック」を参照してください。

- マルチユーザーの次元構築の問題

  他のユーザーがデータベースの読取りや書込みを実行している間は、次元を構築できません。次元の構築が完了すると、Essbase によりアウトラインが再構築され、再構築の操作が完了するまでデータベースがロックされます。

18 ルール・ファイルの使用

この章の内容

関連項目:

- 第 17 章「データ・ロードおよび次元構築の理解」
- 第 20 章「データ・ロードまたは次元構築の実行およびデバッグ」

データ・ロード・ルール・ファイルの作成プロセス

データ・ロードのルール・ファイルを作成するには:

1. 次元構築とデータ・ロードに同じルール・ファイルを使用するかどうかを決定します。

   289 ページの「データ・ロード・ルール・ファイルと次元構築ルール・ファイルの結合」を参照してください。

2. ルール・ファイルを作成します。

   289 ページの「ルール・ファイルの作成」を参照してください。

3. データ・ソースに対するファイル区切り記号を設定します。

   290 ページの「ファイル区切り記号の設定」を参照してください。

4. ルール・ファイルの各フィールドをデータ・ソースにマッピングし、フィールドのプロパティを定義します。

   297 ページの「データ・ロード・フィールド・プロパティの定義」を参照してください。

5. 必要に応じてレコード、フィールドおよびデータ操作を設定し、ロード時にデータ・ソースのデータを変更します。

   第 19 章「ルール・ファイルを使用したレコード、フィールドおよびデータへの操作の実行」を参照してください。

6. ルール・ファイルを検証および保存します。

   297 ページの「次元構築操作の設定についての説明」を参照してください。

次元構築ルール・ファイルの作成プロセス

次元構築のルール・ファイルを作成するには:

1. 次元構築とデータ・ロードに同じルール・ファイルを使用するかどうかを決定します。

   289 ページの「データ・ロード・ルール・ファイルと次元構築ルール・ファイルの結合」を参照してください。

2. ルール・ファイルを作成します。

   289 ページの「ルール・ファイルの作成」を参照してください。

3. データ・ソースに対するファイル区切り記号を設定します。

   290 ページの「ファイル区切り記号の設定」を参照してください。

4. 次元を作成する場合は、次元に名前を付けます。

   290 ページの「新しい次元の名前付け」を参照してください。

5. 構築方法を選択します。

   291 ページの「構築方法の選択」を参照してください。

6. 必要に応じて、構築するメンバーや次元のプロパティを変更または設定します。

   291 ページの「メンバーや次元のプロパティの設定および変更」を参照してください。

7. 必要に応じてレコードやフィールドの操作を設定し、ロード時にデータ・ソースのメンバーを変更します。

   第 19 章「ルール・ファイルを使用したレコード、フィールドおよびデータへの操作の実行」を参照してください。

8. フィールド・タイプの情報(フィールド・タイプ、フィールド番号、次元など)を設定します。

   293 ページの「フィールド・タイプ情報の設定」を参照してください。

9. ルール・ファイルを検証および保存します。

   297 ページの「次元構築操作の設定についての説明」を参照してください。

データ・ロード・ルール・ファイルと次元構築ルール・ファイルの結合

ルール・ファイルの構築前に、データ・ロードのみ、次元構築のみ、またはデータ・ロードと次元構築の両方にルール・ファイルを使用するかどうかを決定する必要があります。ルール・ファイルを作成した後、2 つのルール・ファイルに分割できません。同様に、2 つのルール・ファイルを 1 つのファイルにマージすることもできません。

データ・ソースのロードおよび新しい次元の構築を同時に行う場合は、データ・ロードと次元構築の両方に対して同じルール・ファイルを使用します。

次のいずれかの場合には、データ・ロードと次元構築のそれぞれに対して個別のルール・ファイルを使用します:

- 最初からアウトラインを作成する場合
- データ・ロード時と次元構築時で異なるフィールド操作を実行する場合
- データ・ロードまたは次元構築のルール・ファイルを、個別に再利用する場合
- 次元のみでデータ値を含まないデータ・ソースを使用する場合

ルール・ファイルの作成

Essbase では、ルール・ファイルに基づいて、データ・ロード時または次元構築時にデータ・ソースおよびアウトラインに対して加える変更が認識されます。

注： ルール・ファイル内のレコード・サイズは、64 KB に制限されています。

➤ ルール・ファイルを作成するには:

1 **Essbase サーバー上でルール・ファイルを作成する場合、サーバーに接続します。**

クライアント上でルール・ファイルを作成する場合、サーバーに接続する必要はありません。

2 **データ準備エディタを開きます。**

Oracle Essbase Administration Services Online Help の、ルール・ファイルの作成、またはルール・ファイルを開くことに関する項を参照してください。

新規ルール・ファイルまたは既存のルール・ファイルを開いた状態で、データ準備エディタを起動できます。データ準備エディタが起動したら、エディタを正しいモードに切り替えます。Oracle Essbase Administration Services Online Help の「データ準備エディタについて」を参照してください。

3 **データ・ソースを開きます。**

データ準備エディタでは、テキスト・ファイル、スプレッドシート・ファイル、SQL データ・ソースなどのデータ・ソースを開くことができます。データ準備エディタにデータ・ソースが表示されるので、変更の必要な点を確認できます。

- テキスト・ファイルおよびスプレッドシート・ファイルを開く方法については、Oracle Essbase Administration Services Online Help の、データ・ファイルを開くことに関する項を参照してください。
- SQL データ・ソースを開く方法については、Oracle Essbase Administration Services Online Help の、SQL データベースを開くことに関する項を参照してください。

  SQL データ・ソースを開くには、SQL インタフェースを使用します。サポートされている環境、インストールおよびサポートされているデータ・ソースへの接続に関する情報は、Oracle Essbase SQL Interface Guide を参照してください。詳細は、Essbase 管理者に問い合せてください。

  データ・ソース名(DSN)の代替変数を定義できます。SQL データ・ソースを開き、DSN として使用する値の代替変数を選択できます。たとえば、Payroll_detail という名前の代替変数を作成してから、データ・ソース名の代替変数として Payroll_detail を指定するルール・ファイルを作成できます。データ・ロードまたは次元構築の実行前に、Payroll_detail の値として、使用するデータ・ソースの名前(たとえば Oracle または IBM DB2 データベース)を設定する必要があります。データ・ロードまたは次元構築を実行すると、Essbase サーバーによって検出された代替変数の値が使用されます。118 ページの「代替変数の使用」を参照してください。

  注： SQL データ・ソースを開いたとき、ルール・フィールドには、デフォルトで SQL データ・ソース列の名前が入ります。この名前が Essbase の次元名と同じでない場合は、フィールドを次元にマッピングします。310 ページの「フィールド名の変更」を参照してください。

ファイル区切り記号の設定

ファイル区切り記号は、データ・ソース内のフィールドを区切る文字(複数可)です。デフォルトでは、ルール・ファイルのフィールドはタブで区切られます。ファイル区切り記号としては、カンマ、タブ、スペース、固定幅の列、カスタム値などを設定できます。使用可能なカスタム値は、標準 ASCII 文字セットの番号 0 から 127 までの文字です。通常、ファイル区切り記号の設定は、データ・ソースを開いた直後に行います。

注： SQL データの場合、ファイル区切り記号を設定する必要はありません。

➤ ファイル区切り記号の設定方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「ファイル区切り記号の設定」を参照してください。

新しい次元の名前付け

ルール・ファイルに次元を作成しない場合は、この項はスキップします。

次元を作成する場合、ルール・ファイル内で次元に名前を付ける必要があります。1202 ページの「次元、メンバーおよび別名の命名規則」を参照してください。

属性次元を作成する場合、基本次元は、アウトラインかルール・ファイルですでに定義されている疎次元である必要があります。第 10 章「属性の操作」を参照してください。

- 新しい次元に名前を付ける方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の、ルール・ファイルを使用した次元の命名に関する項を参照してください。

構築方法の選択

次元構築を実行しない場合、この項はスキップします。

新しい次元を構築したり既存の次元にメンバーを追加したりする場合は、作成または変更する次元ごとに、構築方法を指定する必要があります。各構築方法については、331 ページの「構築方法の理解」を参照してください。

- 構築方法を選択する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の、構築方法の指定に関する項を参照してください。

メンバーや次元のプロパティの設定および変更

次元構築を実行しない場合、この項はスキップします。

次元構築を実行する場合、アウトライン内のメンバーや次元のプロパティを設定または変更できます。選択した次元の全メンバーに影響を及ぼす変更、選択した次元のみに影響を及ぼす変更、ルール・ファイル内の全次元に影響を及ぼす変更があります。

データ準備エディタで、メンバーや次元のプロパティを設定または変更できます。また、データ・ソースでメンバー・プロパティを変更することもできます。

データ準備エディタを使用した次元プロパティとメンバー・プロパティの設定

次元構築を実行しない場合、この項はスキップします。

➤ 次元プロパティを設定する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元プロパティの設定」を参照してください。

➤ メンバー・プロパティを設定する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「メンバー・プロパティの設定」を参照してください。

データ・ソースを使用したメンバー・プロパティの操作

次元構築時に次の処理を実行することにより、新規および既存のメンバーのプロパティを変更できます:

- データ・ソース内のフィールドにメンバー・プロパティを含める
- データ・ソース・フィールドを空にしてプロパティをデフォルト値にリセットするか、式または UDA を削除する

管理サービス・コンソールの次の次元構築オプションで、データ・ソース・プロパティ・フィールドの値を関連付けられたメンバーに適用するかどうかを制御します:

- プロパティの変更可能
- 式の変更可能
- UDA の変更可能

データ・ソース内で、プロパティによって変更されるメンバーが含まれるフィールドの直後のフィールドにプロパティを挿入します。たとえば、`Margin%`メンバーが親にロール・アップされず、共有されないように指定するには、次のように設定します:

1. ~プロパティと N プロパティを Margin%フィールドの後に配置します(~プロパティは、メンバーがその親にロール・アップされないことを示し、N プロパティは、メンバーが共有されないことを示します)。例:

   ```
   Margin%  Margin%  ~ N Sales
   ```

2. プロパティ・フィールドのフィールド・タイプを Property と設定します。

   293 ページの「フィールド・タイプ情報の設定」を参照してください。

式、UDA または属性を削除したり、プロパティをデフォルト値にリセットしたりするときは、次の追加手順を実行します:

- 管理サービス・コンソールで「フィールド・プロパティ」ダイアログ・ボックスを開き、「次元構築プロパティ」タブの「プロパティ」フィールドで「フィールドが空の場合に削除」オプションを選択します(「次元構築」ダイアログ・ボックスで適切な次元プロパティが選択されていない場合、このオプションは無視されます)。

- データ・ソース内のフィールドは NULL または空のままにしておきます。

表 40 に、ブロック・ストレージのアウトライン・メンバーにプロパティを適用するためにデータ・ソース内で使用するすべてのメンバー・コードを一覧します(集約ストレージのアウトライン・メンバーに適用できるプロパティのリストについては、1053 ページの「集約ストレージ次元構築のルール・ファイルに関する相違点」を参照)。

表 40　メンバー・プロパティ・コード

コード	説明
%	集計の現在合計値のパーセンテージとして表す
*	集計の現在合計値で乗算
+	集計の現在合計値に加算
-	集計の現在合計値から減算
/	集計の現在合計値で除算
~	集計から除外
^	すべての次元のすべての集計から除外
A	平均タイム・バランス・アイテムとして処理(会計次元にのみ適用)
B	タイム・バランスのゼロまたは#MISSING のデータ値を除外(会計次元にのみ適用)
E	支出アイテムとして処理(会計次元にのみ適用)
F	タイム・バランス期首アイテムとして処理(会計次元にのみ適用)
L	タイム・バランス期末アイテムとして処理(会計次元にのみ適用)
M	タイム・バランスから#MISSING のデータ値を除外(会計次元にのみ適用)
N	データ共有不可
O	ラベルのみのタグ(データ保管なし)
S	メンバーを保管済メンバー(非動的計算メンバーで、ラベルのみメンバーではない)として設定
T	2 パス計算を要求(会計次元にのみ適用)
V	動的計算および保管として作成
X	動的計算として作成
Z	タイム・バランスからゼロのデータ値を除外(会計次元にのみ適用)

フィールド・タイプ情報の設定

次元構築を実行しない場合、この項はスキップします。

次元構築では、データ・ソース内の各フィールドがアウトライン・メンバーを説明する列の一部になります。フィールドには、次の情報を含めることができます:

- メンバー名
- メンバー・プロパティ
- 属性の関連付け

Essbase でこの情報を処理するには、フィールド・タイプの設定時に次の情報を指定する必要があります:

- フィールド・タイプ

  世代フィールド、別名フィールドなど、その列に適切なフィールドのタイプ。フィールド・タイプはデータ・ソースと構築方法に依存します(331 ページの「構築方法の理解」を参照)。

- 次元

  列のメンバーが属する次元。

- 世代番号またはレベル番号

  その列のメンバーの世代番号またはレベル番号。

➤ フィールド・タイプ情報の設定方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールド・タイプの設定」を参照してください。

フィールド・タイプと有効な構築方法

表 41 に、フィールド・タイプと有効な構築方法を一覧します。

表 41　フィールド・タイプと有効な構築方法

フィールド・タイプ[1]	フィールドのコンテンツ	有効な構築方法
別名	別名 **注：**「メンバー更新次元構築」が「未指定の削除」に設定されていて、新規メンバーのデータ・ソースに削除されたメンバーの別名の値が含まれる場合、別名の値は新規メンバーに割り当てられません。	世代、レベルおよび親子参照 295 ページの「フィールド・タイプを割り当てるためのルール」を参照してください。
プロパティ	メンバー・プロパティ。 表 40 を参照してください。	
式	数式	
通貨名	(ブロック・ストレージ・アウトラインのみ)通貨名	
通貨カテゴリ	(ブロック・ストレージ・アウトラインのみ)通貨カテゴリ	
UDA	UDA	

フィールド・タイプ[1]	フィールドのコンテンツ	有効な構築方法
属性の親	属性次元における、次のフィールドの属性メンバーに対する親メンバーの名前	
特定属性次元の名前	指定した属性次元のメンバー。このメンバーは、選択された基本次元の指定の世代またはレベルに関連付けられます。	
世代	指定した世代内のメンバーの名前	世代参照
重複する世代	共有メンバーを子とするメンバーの名前	
重複する世代の別名	共有メンバーの別名	
レベル	レベル内のメンバーの名前	レベル参照
共有するレベル	共有メンバーを子とするメンバーの名前	
共有するレベルの別名	共有メンバーの別名	
親	親の名前	親子参照
子	子の名前	

[1] フィールド・タイプの名前が重複で始まる場合(重複する世代、重複するレベルの別名など)、このフィールド・タイプは第 8 章「重複メンバーのアウトラインの作成および使用」で説明する重複メンバー名とは関係がありません。

フィールド・タイプを割り当てるためのルール

表 42 に、構築方法に基づいて有効なフィールド・タイプを選択するルールを一覧します。

表 42 構築方法に基づいてフィールド・タイプを割り当てるためのルール

構築方法	フィールド・タイプを割り当てるためのルール

世代	● GEN 番号が 2 から開始していない場合は、指定された世代の最初のメンバーが、アウトライン内に存在する必要があります。
	● GEN 番号は連続した範囲になっている必要があります。たとえば GEN 3 と GEN 5 がある場合、GEN 4 も定義する必要があります。
	● DUPGEN フィールドを GEN フィールドの直後に置きます。
	● DUPGENALIAS フィールドを DUPGEN フィールドの直後に置きます。
	● GEN フィールドを、次元内で連続してグループ化します。例:
	`GEN2,PRODUCT GEN3,PRODUCT GEN4,PRODUCT`
	● 関連付けられている基本フィールドの後ろに属性の関連付けフィールドを配置し、関連付けられた基本次元メンバーの世代番号を指定します。例:
	`GEN2,PRODUCT GEN3,PRODUCT OUNCES3,PRODUCT`
	世代番号は、フィールドが値を提供するアウトライン内のメンバーの世代に対応している必要があります。たとえば、`GEN3,PRODUCT` 内の `3` は、フィールド内の値が製品次元の第 3 世代のメンバーであることを示します。`ALIAS2,POPULATION` 内の `2` は、フィールド内の値が人口次元の第 2 世代のメンバーであることを示します。
	注： 世代構築メソッドを使用して重複メンバー次元を作成する場合、世代の最大数は 20 です。

レベル	● DUPLEVEL フィールドを LEVEL フィールドの直後に置きます。 ● DUPLEVELALIAS フィールドを DUPLEVEL フィールドの直後に置きます。 ● 各レコードにレベル 0 メンバーが含まれている必要があります。親の異なる新しいレコード上でレベル 0 メンバーが繰り返される場合、ユーザーが「移動可能」メンバー・プロパティを選択しないかぎり、Essbase によってこのレコードは却下されます。Oracle Essbase Administration Services Online Help の、メンバー・プロパティの設定および変更に関する項を参照してください。 ● レベル・フィールドを、次元内で連続してグループ化します。 ● それぞれのロールアップに対応したフィールドを逐次的な順序で配置します。 ● 単一レコードを使用して、プライマリやセカンダリのロールアップを表現します。 ● 関連付けられている基本フィールドの後ろに属性の関連付けフィールドを配置し、関連付けられた基本次元メンバーのレベル番号を指定します。例: `LEVEL3,PRODUCT OUNCES3,PRODUCT LEVEL2,PRODUCT` ● レベル番号は、フィールドが値を提供するアウトライン内のメンバーのレベルに対応している必要があります。たとえば、LEVEL3,PRODUCT の 3 は、フィールド内の値が製品次元のレベル 3 のメンバーであることを示します。`ALIAS2,POPULATION` 内の `2` は、フィールド内の値が人口次元のレベル 2 のメンバーであることを示します。
親子	フィールド・タイプが「親」または「子」の場合は、「番号」テキスト・ボックスに 0 (ゼロ)を入力します。
属性次元名	世代またはレベル番号は、アウトライン内の関連基本メンバーの世代またはレベルに対応している必要があります。たとえば、`OUNCES3,PRODUCT` 内の `3` は、フィールド内の値が Ounces 属性次元のメンバーであり、同じソース・データ・レコード内の製品次元の第 3 世代メンバーに関連付けられていることを示します。

必要に応じて、フィールドを適切な位置に移動します。307 ページの「フィールドの移動」を参照してください。

➤ フィールドを移動する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールドの移動」を参照してください。

次元構築操作の設定についての説明

次元構築を実行しない場合、この項はスキップします。

ルール・ファイル内に、データ・ソースの読取り後に実行する操作を定義します:

- Essbase でデータ・ソース内のすべてのメンバーを処理および追加した後、メンバーをソートするかどうか
- メンバーを既存のアウトラインに追加するか、未指定のメンバーをアウトラインから削除するか

  未指定のメンバーは、構築方法が世代参照、レベル参照および親子参照の場合にのみ削除できます。

注： メンバーを削除すると式のないレベル 0 動的計算メンバーになる場合は、アウトラインは無効です。

データ・ロード・フィールド・プロパティの定義

ルール・ファイルの各フィールドを、対応するアウトライン・メンバー、またはデータ・フィールドか無視するフィールドにマッピングする必要があります。その他のフィールド特性も適用できます。

メンバー・アウトラインが重複している場合、Essbase でのフィールドのマッピングの方法(レベル参照、世代参照または次元参照)を指定する必要があります。

- レベル参照と世代参照: データ・ソースでは重複メンバーを一意に識別するための複数のフィールドが重複メンバー次元に含まれます。
  - 次元内のフィールドがデータ・ソース内においてボトムアップ順で編成されている場合は、レベル参照方法を使用します。
  - 次元内のフィールドがデータ・ソース内においてトップダウン順で編成されている場合は、世代参照方法を使用します。例:

```
                gen2,Market, gen3,Market, Product, Year, Measures, Scenario, *
                data
                *
State,"New York","100-10",Jan,Sales,Actual,42
City,"New York","100-20",Jan,Sales Actual,82
State,Texas,"100-10",Jan,Sales,Actual,37
```

- 次元参照: アウトライン内の複数の次元に重複したメンバー名が含まれる場合、たとえば Other などのメンバー名が複数の次元で意味を持つ場合は、次元参照方法を使用できます。次元参照方法を使用するようにフィールドを設定すると、そのフィールド内のメンバーがどの次元に属しているかがわかります。次元参照方法が指定されているフィールドでは、フィールド内のメンバーが

そのフィールドに指定された次元に属していることが Essbase により確認されます。

➤ データ・ロードの世代、レベルまたは次元を指定する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールド名のマッピング」を参照してください。

レコード、フィールドおよびデータの操作の実行

ルール・ファイルを使用すると、データ・ソースに変更を加えることなく、レコード、フィールドおよびデータ値の操作を実行してからデータベースにロードできます。第 19 章「ルール・ファイルを使用したレコード、フィールドおよびデータへの操作の実行」を参照してください。

検証、保存および印刷

ルール・ファイルを検証して、ルール・ファイル内のメンバーおよび次元がアウトラインにマッピングされていることを確認します。検証では、データ・ソースが適切にロードされるかどうかは確認できません。

➤ ルール・ファイルを検証する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の、ルール・ファイルの検証に関する項を参照してください。

ルール・ファイルが有効な場合、データ・ロードや次元構築を実行できます。第 20 章「データ・ロードまたは次元構築の実行およびデバッグ」を参照してください。

ルール・ファイルが無効である場合については、ルール・ファイルのタイプごとに適切なトピックを参照してください:

- データ・ロード: 298 ページの「データ・ロード・ルール・ファイルが有効であるための要件」
- 次元構築: 299 ページの「次元構築ルール・ファイルが有効であるための要件」

➤ ルール・ファイルを保存する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の、ルール・ファイルの保存に関する項を参照してください。

データ・ロード・ルール・ファイルが有効であるための要件

データ・ロード・ルール・ファイルを検証する場合、次の質問すべてに対する回答が「はい」である必要があります。

- ルール・ファイルが正しいアウトラインに関連付けられていますか?

Oracle Essbase Administration Services Online Help の、ルール・ファイルの検証に関する項を参照してください。

- データ・ソース内の各レコードには、各次元に対するメンバーが 1 つのみ存在していますか?

  275 ページの「データ・ソース内のアイテム」を参照してください。

- すべてのメンバー名および次元名のスペルは正しいですか?
- メンバーに数字またはファイル区切り記号が含まれている場合、これらのメンバーはすべて引用符で囲まれていますか?

  276 ページの「有効なメンバー・フィールド」を参照してください。

- データ・ソース内に余分な区切り記号はありませんか?

  279 ページの「余分な区切り記号(ルール・ファイルを使用する場合)」を参照してください。

- 各データ・フィールドのマッピング先メンバーは、ルール・ファイル内で正しいスペルで記述されていますか?

  310 ページの「フィールド名の変更」を参照してください。

- ファイル区切り記号は正しく配置されていますか?

  278 ページの「有効な区切り記号」を参照してください。

- フィールド名の中のメンバーは、有効なメンバーですか?

  309 ページの「フィールドのマッピング」を参照してください。

- 次元名は、フィールド名とヘッダーに使用されたりせずに、単一のフィールドのみで使用されていますか?

  単一のデータ値は、1 セットのメンバーに対してのみマッピングできます。

- 1 つのフィールドのみが、データ・フィールドとして定義されていますか?

  312 ページの「データ・フィールドとしての列の定義」を参照してください。

- 関連付けられたアウトライン内で、UDA が符号の反転のために使用されていますか?

  314 ページの「フィールド符号の反転」を参照してください。

次元構築ルール・ファイルが有効であるための要件

次元構築ルール・ファイルを検証する場合は、次の質問すべてに対する回答が「はい」である必要があります。

- ルール・ファイルが正しいアウトラインに関連付けられていますか?

  Oracle Essbase Administration Services Online Help の、ルール・ファイルの検証に関する項を参照してください。

- 各レコードには、各次元に対するメンバーが 1 つのみ存在していますか?

  275 ページの「データ・ソース内のアイテム」を参照してください。

- すべてのメンバー名および次元名のスペルは正しいですか?
- メンバーに数字またはファイル区切り記号が含まれている場合、これらのメンバーはすべて引用符で囲まれていますか?

  276 ページの「有効なメンバー・フィールド」を参照してください。
- データ・ソース内に余分な区切り記号はありませんか?

  279 ページの「余分な区切り記号(ルール・ファイルを使用する場合)」を参照してください。
- リファレンス番号が連続していますか?

  295 ページの「フィールド・タイプを割り当てるためのルール」を参照してください。
- 繰返しの世代が存在していませんか?

  295 ページの「フィールド・タイプを割り当てるためのルール」を参照してください。
- 各フィールド・タイプは、構築方法に対して有効ですか?

  294 ページの「フィールド・タイプと有効な構築方法」を参照してください。
- すべてのフィールドの順序は正しいですか?

  295 ページの「フィールド・タイプを割り当てるためのルール」を参照してください。
- 各子フィールドには、親フィールドがありますか?
- すべての次元名が、アウトラインまたはルール・ファイル内に存在しますか?
- ルール・ファイルのヘッダー・レコードおよびデータ・ソースのヘッダー・レコードの両方に指定されている次元はありますか?

  次元は、ルール・ファイルのヘッダーかデータ・ソースのヘッダーのいずれかに指定できますが、両方には指定できません。304 ページの「ヘッダー・レコードの定義」を参照してください。

ルール・ファイルのコピー

権限に従って、ルール・ファイルを任意の Essbase サーバー上のアプリケーションやデータベースにコピーできます。アプリケーションの移行の一環として、ルール・ファイルを複数のサーバーにコピーすることもできます。

➤ ルール・ファイルをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ルール・ファイルのコピー	Oracle Essbase Administration Services Online Help
MaxL	**alter object**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYOBJECT	『Oracle Essbase テクニカル・リファレンス』

ルール・ファイルの印刷

データ・ロード・ルール・ファイルまたは次元構築ルール・ファイルのすべてのコンテンツとプロパティを印刷できます。印刷するプロパティと設定を指定することもできます。

➤ ルール・ファイルを印刷する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「ルール・ファイルの印刷」を参照してください。

19 ルール・ファイルを使用したレコード、フィールドおよびデータへの操作の実行

この章の内容

関連項目:

- 第 17 章「データ・ロードおよび次元構築の理解」
- 第 18 章「ルール・ファイルの使用」

レコード上での操作の実行

レコード・レベルで操作を実行できます。たとえば、特定のレコードをデータベースにロードする前に除外できます。次の項を参照してください。

レコードの選択

選択条件を設定して、Essbase でデータベースにロードするレコードや、次元の構築に使用するレコードを指定できます。選択条件は文字列と数値の条件です。Essbase でレコードをロードするためには、レコード内の 1 つ以上のフィールドがこの条件を満たしている必要があります。レコード内に選択条件を満たさないフィールドが 1 つでもある場合、そのレコードは Essbase でロードされません。選択条件は 1 つ以上定義できます。たとえば、データ・ソースから 2003 年の予算データのみをロードする場合、最初のフィールドが Budget で、2 番目のフィールドが 2003 であるレコードのみをロードする選択条件を作成します。複数のフィールドに対して選択条件を定義する場合は、Essbase での条件の組合せを指定できます。304 ページの「複数の選択条件および除外条件の結合」を参照してください。

➤ レコードを選択する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「レコードの選択」を参照してください。

レコードの除外

除外条件を設定することにより、Essbase で無視されるレコードを指定できます。除外条件は文字列と数値の条件で、レコード内の 1 つ以上のフィールドがこの条件を満たしていると、Essbase でそのレコードは除外されます。選択条件は 1 つ以

上定義できます。レコード内にこの除外条件を満たすフィールドがなければ、Essbase でそのレコードはロードされます。たとえば、データ・ソースから実績データを除外して予測データのみをロードする場合、最初のフィールドが Actual のレコードを除外するような除外条件を作成します。

➤ レコードを除外する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「レコードの除外」を参照してください。

複数の選択条件および除外条件の結合

複数のフィールドに選択条件や除外条件を定義するときは、Essbase で、これらのフィールドにおけるルールの組合せ方(条件に論理演算子 AND と OR のどちらを関連付けるか)を指定できます。ブール・グループから AND を選択した場合、フィールドはすべての条件を満たす必要があります。OR を選択した場合、フィールドはいずれか 1 つの条件を満たしていればよいことになります。グローバルなブール設定は、ルール・ファイル内のデータ・ロード・フィールドと次元構築フィールドのすべての選択条件または除外条件に適用されます。

注： 選択条件と除外条件が同じレコードに適用されると(1 つのレコードに対して選択条件と除外条件の両方を定義した場合)、そのレコードは除外されます。

➤ 複数のフィールドの選択条件と除外条件を組み合せる方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「選択条件や除外条件の結合」を参照してください。

表示するレコードの設定

データ準備エディタで、Essbase に表示されるレコードの数と最初のレコードを指定できます。最初のレコードを指定すると、Essbase ではそれより前のレコードはすべてスキップされます。たとえば、開始レコードとして 5 と指定した場合、レコード 1 から 4 までは表示されません。

注： Essbase では、スキップするレコードをカウントする際、ヘッダー・レコードがデータ・レコードと同様に扱われます。

➤ 表示するレコードの設定方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の、レコード表示の設定に関する項を参照してください。

ヘッダー・レコードの定義

データ・ソースには、次の項目を含めることができます:

- データ・レコード: メンバー・フィールドとデータ・フィールドが含まれるレコード
- ヘッダー・レコード: データ・ソースのコンテンツとデータ・ソースからデータベースに値をロードする方法が記述されたレコード

ルール・ファイルには、データ・ソースのデータを変換してデータベースにマッピングするレコードが含まれます。その情報の一部として、ルール・ファイルにはヘッダー・レコードも含まれます。たとえば、Sample.Basic データベースには年次元があります。異なる地区から月番号付きの複数のデータ・ソースが届く場合、月自体はデータ・ソース内に指定されていない可能性があります。月を指定するには、ヘッダー情報を設定する必要があります。

ヘッダー・レコードは、次のいずれかの方法で作成できます:

- ルール・ファイル内にヘッダー情報を定義します。

  ルール・ファイルのヘッダーは、データのロード中または次元の構築中にのみ使用され、これによってデータ・ソースが変更されることはありません。ルール・ファイル内に設定されたヘッダー情報は、ルール・ファイルでデータ・ソース内のヘッダー・レコードを同時に指している場合は使用されません。
- データ・ソース内にヘッダー情報を定義し、ルール・ファイル内でヘッダー・レコードを指します。

  データ・ソース内にヘッダー情報を配置すると、フォーマットの異なる複数のデータ・ソースに対して同じルール・ファイルを使用できます。データ・ソースのフォーマットは、ルール・ファイル内ではなく、データ・ソース・ヘッダー内で指定されるためです。

  データ・ソースに 1 つ以上のヘッダーを追加するときは、ルール・ファイルにデータ・ソース内のヘッダーの位置も指定する必要があります。ルール・ファイルを使用して、ヘッダー情報をデータ・レコードではなくヘッダー・レコードとして読み取るように Essbase に指示します。各ヘッダー・レコード内のヘッダー情報のタイプも指定できます。

  データ・ソース内で定義されているヘッダー情報は、ルール・ファイル内で定義されているヘッダー情報よりも優先されます。

➤ ルール・ファイルにヘッダーを定義する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の、ルール・ファイルのデータ・ソース・メンバーの指定に関する項を参照してください。

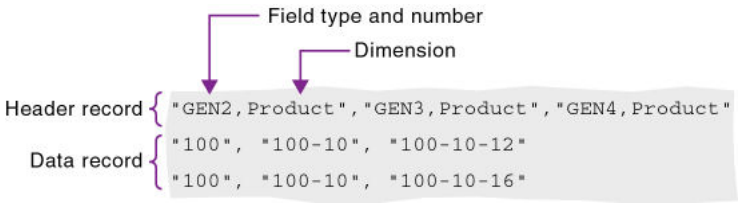
➤ データ・ソース内にヘッダーを定義する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「データ・ソースのヘッダーの設定」を参照してください。

データ・ソース・ヘッダー

ヘッダー情報をデータ・ソースの先頭レコードに追加し、ルール・ファイルでヘッダー・レコードの場所を指定して、次元を動的に構築できます。

ヘッダー・レコードには、各フィールドのフィールド定義が並んでいます。フィールド定義には、フィールドのタイプ、フィールド番号、フィールドのロード先の次元名が含まれます。図 65 に、ヘッダー・レコードのフォーマットを示します:

図 65　3 つのフィールド定義を有するヘッダー・レコード



ファイル区切り記号がカンマの場合、各フィールド定義を引用符(" ")で囲みます。

データ・ソース内にヘッダー情報を設定した後、ルール・ファイル内にヘッダー情報の場所を指定する必要があります。ルール・ファイルがデータ・ソース内のヘッダー情報を参照している場合は、Essbase ではフィールドのタイプと次元を決定する際、ルール・ファイル内の情報ではなくデータ・ソース内の情報が使用されます。

有効なデータ・ソース・ヘッダー・フィールド・タイプ

次の有効なフィールド・タイプは、大文字で記述する必要があります:

- GEN、DUPGEN および DUPGENALIAS
- LEVEL、DUPLEVEL および DUPLEVELALIAS
- PARENT、CHILD
- PROPERTY
- ALIAS
- FORMULA
- CURNAME
- CURCAT
- UDA
- ATTRPARENT
- 属性次元の名前(CAFFEINATED など)

設定するフィールド・タイプごとにフィールド番号が必要です。フィールド・タイプが属性次元の名前である場合、9 より大きいフィールド番号は使用できません。293 ページの「フィールド・タイプ情報の設定」を参照してください。

フィールドに対する操作の実行

フィールド・レベルで操作を実行できます。たとえば、レコード内の新しい位置にフィールドを移動できます。次の項を参照してください。

フィールドの無視

データ・ソースの指定した列のすべてのフィールドを無視できます。フィールドはデータ・ソース内に存在しますが、Essbase データベースにはロードされません。たとえば、Sample.Basic データベースには 5 つの標準次元(年、製品、市場、メジャーおよびシナリオ)があります。データ・ソース内にこのいずれかの次元のメンバーでない追加フィールド(Salesperson フィールドなど)があった場合、そのフィールドを認識しないように Essbase に指示できます。

➤ ある列内のすべてのフィールドを無視する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールドの無視」を参照してください。

文字列の無視

トークンと呼ばれるある文字列に一致するデータ・ソース内の任意のフィールドを無視できます。文字列の値に基づいてフィールドを無視するように設定した場合、そのフィールドは、特定の列内に限らず、データ・ソース内のどこにあっても無視されます。たとえば、データ・ソースがコンピュータによって生成されたテキスト・フォーマットのレポートである場合、ページ間の水平線や見出しの囲み枠が、特別な ASCII 文字で生成されていることがあります。こうした特別な ASCII 文字をトークンとして定義し、無視する対象に指定できます。

➤ 文字列のすべてのインスタンスを無視する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「文字列の一致に基づいたフィールドの無視」を参照してください。

フィールドの整列

ルール・ファイル内のフィールドの順序を、データ・ソース内のフィールドの順序と異なるように設定できます。データ・ソースは変更されません。次の項を参照してください。

フィールドの移動

ルール・ファイルを使用して、フィールドを別の場所に移動できます。たとえば、データ・ソース内の最初のフィールドがデータ・ロード時または次元構築時に 3 番目のフィールドになるように指定できます。

フィールドを移動すると、フィールドがマージされたように見える場合があります。たとえば次のように、空のセルが含まれるフィールドをレコードの最終フィールドの位置に移動した場合、このフィールドはその左のフィールドにマージされます。

```
   1<tab>2<tab>3
1<tab>2<tab>(null)
```

マージされないようにする場合は、空のセルを区切り記号に置き換えます。

➤ フィールドを移動する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールドの移動」を参照してください。

フィールドの結合

複数のフィールドを 1 つのフィールドに結合できます。新しいフィールドには、結合された最初のフィールドの名前が付けられます。たとえば、データ・ソースに製品番号フィールド(100)と製品ファミリ・フィールド(-10)がある場合、これらのフィールドを Sample.Basic データベースにロードする前に結合する必要があります(100-10)。

フィールドを結合する前に、結合する順にフィールドの位置を移動します。Oracle Essbase Administration Services Online Help の「フィールドの移動」を参照してください。

➤ フィールドを結合する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールドの結合」を参照してください。

フィールド結合によるフィールドの作成

結合されたフィールドを新しいフィールド内に配置して、フィールドを結合できます。この方法では、元のフィールドは変更されません。フィールドの作成は、データ・ソースのフィールドを連結してメンバーを作成する必要がある場合に便利です。

たとえば、データ・ソースに製品番号フィールド(100)と製品ファミリ・フィールド(-10)がある場合、これらのフィールドを Sample.Basic データベースにロードする前に結合する必要があります(100-10)。ただし、データ・ソース内の 2 つの既存フィールドを維持する場合は、結合を使用してフィールドを作成できます(100-10)。この場合、データ・ソースは、100、-10、100-10 の 3 つのフィールドで構成されることになります。

フィールドを結合する前に、結合する順にフィールドの位置を移動します。Oracle Essbase Administration Services Online Help の「フィールドの移動」を参照してください。

➤ 既存のフィールドを結合してフィールドを作成する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「結合を使用したフィールドの作成」を参照してください。

フィールドのコピー

元のフィールドを変更しないで、フィールドのコピーを作成できます。たとえば、1 回の次元構築で、マルチレベルの属性次元を定義し、属性に基本次元のメンバーを関連付ける場合、フィールドの一部をコピーします。345 ページの「マルチレベルの属性次元の扱い」を参照してください。

➤ フィールドをコピーするには、1 つのフィールドを選択し、結合を使用して新しいフィールドを作成します。『Oracle Essbase Administration Services オンライン・ヘルプ』の「結合を使用したフィールドの作成」を参照してください。

フィールドの分割

1 つのフィールドを 2 つに分割できます。たとえば、Sample.Basic データベースのデータ・ソースに、UPC100-10-1 という値が含まれているフィールドがある場合、フィールドから UPC を分割して無視できます。その後、製品番号の 100-10-1 のみがロードされます。

➤ フィールドを分割する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールドの分割」を参照してください。

追加のテキスト・フィールドの作成

2 つのフィールド間にテキスト・フィールドを 1 つ作成できます。結合するフィールド間にテキストを挿入するテキスト・フィールドも作成できます。たとえば、あるフィールドに 100、別のフィールドに 10-1 というデータが含まれる場合、この 2 つのフィールドの間にダッシュが含まれるテキスト・フィールドを挿入し、3 つのフィールドを結合して、製品次元の 100-10-1 メンバーを作成できます。

➤ フィールドを作成し、テキストを取り込む方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の、テキストを使用したフィールドの作成に関する項を参照してください。

フィールド操作を元に戻す

「元に戻す」コマンドを使用して(「編集」から「元に戻す」を選択)、最後に実行したフィールド操作(移動、分割、結合、テキストを使用した作成、結合を使用した作成など)を元に戻すことができます。フィールド操作を元に戻す処理は、他のアクションを実行した場合でも可能です。フィールド操作を元に戻す処理は、最後に実行した操作から順番に行われます。

➤ 1 つ以上のフィールド操作を元に戻す方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールド操作を元に戻す」を参照してください。

フィールドのマッピング

この項の内容は、データ・ロードのみに適用されます。次元構築を実行する場合、この項はスキップします。

ルール・ファイルを使用して、データ・ロードの際にデータ・ソースのフィールドを Essbase メンバー名にマッピングできます。データ・ソース内のどのフィールドを Essbase データベース内のどのメンバーまたはメンバーの組合せにマッピングするかを指定することによって、データ・ロード時に、データ・ソース内のフィー

ルドを直接 Essbase データベース内のフィールドにマッピングできます。データ・ソースは変更されません。

注： SQL データ・ソースを開くと、フィールドにはデフォルトで SQL データ・ソース列の名前が入ります。SQL 列の名前と Essbase の次元名が同じである場合、列名をマッピングする必要はありません。

➤ フィールドをマッピングする方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールド名のマッピング」を参照してください。

フィールド名の変更

データ・ソースをロードするには、データ・ソースのフィールドをデータベースの次元およびメンバーにマッピングする方法を指定する必要があります。ルール・ファイルでは、データ・ソースのロードのたびに、データ・ソースのフィールドがメンバー名と一致するようにフィールドを変換できます。この処理では、データ・ソースは変更されません。

ルール・ファイルの機能は次のとおりです:

- データ・ソースのメンバー・フィールドを、データベースの次元およびメンバーにマッピングします
- データ・ソースのデータ・フィールドを、データベースのメンバー名またはメンバーの組合せ(Jan、Actual など)にマッピングします

次の項を参照してください。

テキスト文字列の置換

データ・ロード時または次元構築時に、フィールドが Essbase のメンバー名にマッピングされるように、テキスト文字列を置換できます。データ・ソースは変更されません。たとえば、データ・ソースで New York が NY のように省略される場合、ルール・ファイルを使用して NY を New York で置換できます。

➤ テキスト文字列を置換する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールド名の置換」を参照してください。

空のフィールドへのテキストの配置

列内の空のフィールドをテキストで置換できます。たとえば、列内の空フィールドがデフォルト値を表す場合、デフォルト値を挿入するか、`#MI` を挿入して欠落値を表すことができます。

➤ 空のフィールドをテキストで置換する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の、空のフィールドへのテキストの配置に関する項を参照してください。

フィールドの大文字と小文字の変更

データ・ロード時または次元構築時に、フィールドが Essbase のメンバー名にマッピングされるようにフィールドの大文字と小文字を変更できます。データ・ソースは変更されません。たとえば、データベースでは jan のように小文字表記のフィールドが、データ・ソースで JAN のように大文字表記される場合、ルール・ファイルを使用してフィールドを小文字に変更できます。

➤ フィールド内の値の大文字と小文字を変更する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールドの大文字と小文字の変更」を参照してください。

先頭および末尾のスペースの削除

データ・ソースのフィールドから先頭スペースや末尾スペースを削除できます。先頭スペースや末尾スペースが含まれるフィールド値は、スペースに囲まれた名前部分が完全に一致していても、メンバー名にマッピングされません。

デフォルトでは、Essbase では、先頭および末尾のスペースが削除されます。

➤ フィールドを囲むスペースを削除する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールドの前後のスペースの削除」を参照してください。

スペースからアンダースコアへの変換

データ・ソースのフィールドにスペースがある場合は、そのスペースをアンダースコアに変換して、そのフィールド値をデータベースのメンバー名と突き合せることができます。

➤ スペースをアンダースコアに変更する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「スペースからアンダースコアへの変換」を参照してください。

フィールド値への接頭辞または接尾辞の追加

データ・ソースの各フィールド値に接頭辞や接尾辞を追加できます。たとえば、2002 を年次元のすべてのメンバー名に接頭辞として追加できます。

➤ フィールドに接頭辞または接尾辞を追加する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「接頭辞および接尾辞の追加」を参照してください。

データに対する操作の実行

この項の内容は、データ・ロードのみに適用されます。次元構築を実行する場合、この項はスキップします。

フィールド内のデータを操作できます。たとえば、レコード内の新しい位置にフィールドを移動できます。次の項を参照してください。

データ・フィールドとしての列の定義

この項の内容は、データ・ロードのみに適用されます。次元構築を実行する場合、この項はスキップします。

データ・ソースの各レコードに、各次元を表す列と 1 つのデータ列が含まれている場合は、次の例のように、データ列をデータ・フィールドとして定義する必要があります:

```
Market, Product, Year, Measures, Scenario
Texas  100-10  Jan  Sales   Actual   42
Texas  100-20  Jan  Sales   Actual   82
Texas  100-10  Jan  Sales   Actual   37
```

レコードでデータ・フィールドとして定義可能なフィールドは 1 つのみです。

➤ データ・フィールドを定義するには、『Oracle Essbase Administration Services オンライン・ヘルプ』のデータ・フィールドとしての列の定義に関する項を参照してください。

既存値に対する加算および減算

この項はデータ・ロードのみに適用されます。次元構築を実行する場合、この項はスキップします。

Essbase のデフォルトでは、データベースの既存値は、データ・ソースの値で上書きされますが、新しくロードされたデータ値が既存のデータ値にどのような影響を与えるかはユーザーが決定できます。

入力データ値を使用して、データベースの既存の値に対して加算または減算を実行できます。たとえば、週の値をロードした場合、これらの値を加算することにより、データベース内で月の値を生成できます。

このオプションを使用すると、データのロード中にデータベースがクラッシュした場合、Essbase によってアプリケーション・ログに最後にコミットされた行の番号が記録されていても、通常よりもリカバリが困難になります。804 ページの「アプリケーション・ログのコンテンツ」を参照してください。

ブロック・ストレージ・データベースに対して、「行のコミット」データベース・トランザクション・オプションを 0 に設定して、リカバリが困難になるのを防ぎます。そうすると、Essbase では、ロード全体が単一のトランザクションと見なされるようになり、ロードが完了したときにのみデータがコミットされます。861 ページの「分離レベルの理解」を参照してください。

管理サービス・コンソールを使用すると、ブロック・ストレージ・データベースおよび集約ストレージ・データベースに対して既存の値の加算および減算が可能です。MaxL を使用すると、集約ストレージ・データベースに対してのみ、既存の値の加算および減算が可能です。

➤ 既存のデータ値の加算または減算を行うには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**データ値への加算** **データ値からの減算**	Oracle Essbase Administration Services Online Help
MaxL	**import data (集約ストレージ)**	『Oracle Essbase テクニカル・リファレンス』

既存のデータ値の消去

この項はデータ・ロードのみに適用されます。次元構築を実行する場合、この項はスキップします。

新しい値をロードする前に、既存のデータ値をデータベースから消去できます。Essbase のデフォルトでは、データベースの既存の値がデータ・ソースの新しい値で上書きされます。ただし、Essbase では、データ値を加算する場合は新しいデータ値が既存の値に加算され、データ値を減算する場合は既存の値から新しい値が減算されます。

新しい値を加算または減算する前に、既存の値が正しいことを確認します。最初の値セットをデータベースにロードする前に、既存の値がないことを確認します。

たとえば、1 月の各週の Sales を加算することによって、1 月の月間 Sales を計算する場合を考えてみます:

```
January Sales = Week 1 Sales + Week 2 Sales + Week 3 Sales + Week 4 Sales
```

Week 1 Sales をロードすると、January Monthly Sales のデータベース値は消去されます。既存の値がある場合、Essbase により次の計算が行われます:

```
January Sales = Existing Value + Week 1 Sales + Week 2 Sales + Week 3 Sales + Week
4 Sales
```

データ・ロードの対象にならないフィールドからデータを消去することもできます。たとえば、データ・ソースに January、February および March のデータが含まれていて、March のデータのみをロードする場合、January と February のデータを消去できます。

➤ 既存の値を消去する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「既存のデータ値の消去」を参照してください。

注： 透過パーティションを使用している場合に値を消去する手順は、ローカル・データベースのデータを消去する場合とまったく同じです。

すべてのデータの置換

この項の内容は、集約ストレージ・データベースにデータをロードする場合のみに適用されます。ブロック・ストレージ・データベースにデータをロードする場合や次元構築を実行する場合、この項はスキップします。

集約ストレージ・データベースの場合、Essbase ではデータベース内のすべてのデータ、またはデータベース内の各増分データ・スライス内のすべてのデータを削除した後、指定したデータ・ロード・バッファのコンテンツでデータを置換できます。この機能は、完全に再ロードできるくらい小さいデータ・セットを操作している場合や、更新されない大きい静的データ・セットと変更の追跡が必要な小さい揮発性のデータ・セットに分割可能なデータを操作している場合によく使用します。

すべてのデータを置換する方法については、1066 ページの「データベースまたは増分データ・スライスのコンテンツの置換」および Oracle Essbase Administration Services Online Help の「集約ストレージ・データベースのコンテンツの置換」を参照してください。

データ値のスケーリング

この項はデータ・ロードのみに適用されます。次元構築を実行する場合、この項はスキップします。

データ・ソースのデータ値が、データベースのデータ値と同じスケールでない場合、そのデータ値をスケーリングできます。

たとえば、売上高の実際の値が$5,460 であるとします。Sales データ・ソースが値を 100 単位で追跡する場合、値は 54.6 になります。Essbase データベースが実際の値を追跡する場合、Sales データ・ソースから取得した値(54.6)に 100 を乗算して、Essbase データベースに正しい値(5460)が表示されるようにします。

➤ データ値をスケーリングする方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「データ値のスケーリング」を参照してください。

フィールド符号の反転

この項はデータ・ロードのみに適用されます。次元構築を実行する場合、この項はスキップします。

符号を反転することにより、データ・フィールドの値をリバース(反転)できます。符号の反転は、アウトラインの UDA に基づきます。たとえば、データを会計次元にロードするとき、勘定科目メンバーに Expense の UDA が含まれるレコードのプ

ラス符号がマイナス符号に変更されるように指定できます。158 ページの「UDA の作成」を参照してください。

- フィールドの符号を反転する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「符号の反転」を参照してください。

20 データ・ロードまたは次元構築の実行およびデバッグ

この章の内容

関連項目:

- 第 17 章「データ・ロードおよび次元構築の理解」
- 第 18 章「ルール・ファイルの使用」

データ・ロードおよび次元構築における前提条件

1 つ以上の外部データ・ソースから、データまたはメンバーを Essbase サーバーにロードできます。アウトラインを更新しないでデータのロードのみを実行するか、データをロードしないでアウトラインの更新のみを実行するか、データのロードと次元構築を同時実行するかを選択できます。データのロードまたは次元構築の前に、次のアイテムが準備できていることを確認します:

- Essbase データベース
- 適切な Essbase サーバーへの接続
- 1 つ以上の有効なデータ・ソース

  274 ページの「データ・ソース」を参照してください。
- ルール・ファイル(使用する場合)

  280 ページの「ルール・ファイル」を参照してください。
- ルール・ファイルを使用しない場合は、フリーフォーム・データ・ロード用に正しくフォーマットされたデータ・ソース

  281 ページの「ルール・ファイルを必要としないデータ・ソース」を参照してください。

データ・ロードまたは次元構築の実行

Administration Services を使用してデータ・ロードまたは次元構築を実行する場合、ロードまたは構築をバックグラウンドで実行することにより、ロードまたは構築の処理中も引き続き別の作業を行うことができます。その後、バックグラウンド・プロセスのステータスをチェックして、いつロードまたは構築が完了したかを確認できます。Oracle Essbase Administration Services Online Help の、データのロードおよび次元構築に関する項を参照してください。

次元構築で複数のデータ・ソースを使用している場合は、合計処理時間を削減するために、再構築が延期された次元構築を実行できます。319 ページの「再構築が延期された次元構築の実行」を参照してください。

注： データの透過パーティションへのロード手順は、ローカル・データベースにデータをロードする手順と同じです。

➤ データのロードまたは次元構築を実行するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データ・ロードおよび次元構築	Oracle Essbase Administration Services Online Help
MaxL	データのロードの場合: **import data** 次元構築の場合: **import dimensions**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	データのロードの場合: IMPORT 次元構築の場合: BUILDDIM	『Oracle Essbase テクニカル・リファレンス』

データ・ロードまたは次元構築の停止

データ・ロードや次元構築は、完了する前に停止できます。ただし、必要な場合以外は停止しないでください。データ・ロードや次元構築のプロセスが停止した場合、Essbase には部分的にロードされたファイルの名前が表示されます。

クライアントからデータ・ロードまたは次元構築を開始し、サーバーから停止する場合、クライアントが停止要求に応じるまで時間がかかることがあります。Essbase では、すべてのソース・データが読み取られるまでソース・ファイルの読取りが行われるので、この所要時間は、ファイル・サイズと Essbase で処理されたソース・データの量によって異なります。プロセスを開始元のコンピュータで停止する場合は、すぐに停止できます。

注： データをブロック・ストレージ・データベースにロードする際にデータ値に対して加算または減算を行う場合は、可能であればコミット分離レベル設定を使用します。データ・ロードが停止された場合、この設定により、データ・ロード以前の状態に戻すことができます。861 ページの「分離レベルの理解」を参照してください。データ値に対して加算または減算を行うようなデータ・ロードを停止する場合は、325 ページの「Essbase サーバーのクラッシュからのリカバリ」を参照してください。

➤ データ・ロードまたは次元構築を完了前に停止するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ユーザー・セッションおよび要求の切断	Oracle Essbase Administration Services Online Help
MaxL	**alter system kill request**	『Oracle Essbase テクニカル・リファレンス』

データのロードおよび次元の構築に関するヒント

サブトピック

- 再構築が延期された次元構築の実行
- データをロードする場所の決定
- データ・ソース内の欠落フィールドの処理
- データ・ソースからのレコードのサブセットのロード
- データのロード・オプションの保存と再利用
- 次元構築およびアウトライン編集時の名前の参照と挿入の最適化

この項には、データのロードと次元の構築に役立つトピックが含まれています。
参照:

- 319 ページの「再構築が延期された次元構築の実行」
- 320 ページの「データをロードする場所の決定」
- 321 ページの「データ・ソース内の欠落フィールドの処理」
- 322 ページの「データ・ソースからのレコードのサブセットのロード」
- 322 ページの「データのロード・オプションの保存と再利用」

再構築が延期された次元構築の実行

データのみをロードする場合や次元構築で単一データ・ソースを使用する場合は、この項をスキップします。

デフォルトでは、ユーザーがアウトラインに変更を加えるたびに、Essbase によって変更のタイプが認識され、必要に応じてデータベースが再構築されます。データベースの再構築では、データベースが再度構築されます。これは時間のかかる処理で、そのプロセスに多くのディスク・スペースが必要になるだけでなく、ファイルの最適化が行われる場合もあります。変更のタイプと必要になる再構築のタイプについては、次のトピックを参照してください:

- ブロック・ストレージ・アウトラインの場合、933 ページの「データベースの再構築の最適化」を参照してください。
- 集約ストレージ・アウトラインの場合、1131 ページの「集約ストレージ・データベースの再構築」を参照してください。

再構築が延期された次元構築(増分次元構築)では、次元構築のために複数のデータ・ソースが読み取られ、すべてのデータ・ソースが処理されるまで再構築は延期されます。再構築が延期された次元構築を行うには、次のように、様々な方法でアクションを指定します:

- Administration Services では、単一のダイアログ・ボックス内にすべてのデータ・ソースをリストできます。

  リストされたファイルがすべて次元構築の場合、再構築が延期された次元構築のオプションを使用できます。このオプションを選択すると、すべてのソースが処理されるまで再構築が延期されます。アウトラインの検証は、各次元構築処理が完了してから行われます。Oracle Essbase Administration Services Online Help の、データのロードおよび次元構築に関する項を参照してください。

- MaxL では、1 つの **import database** ステートメントにすべてのデータ・ソースを含めることができます。

  ファイルごとにアウトラインの検証を行うかどうかを制御できます。最終ファイルに対しては、アウトラインの検証を使用可能にする必要があります。『Oracle Essbase テクニカル・リファレンス』の MaxL の項にある「import database」と「import dimension」を参照してください。

- ESSCMD では、いくつかのコマンドを使用します:
  - BEGINCBUILDDIM: 再構築が延期された次元構築を行うことを指示
  - INCBUILDDIM: データ・ソースごとに、次元構築にデータ・ソースを含めることを指示
  - ENDBUILDDIM: 必要に応じて再構築をトリガー

いずれの場合も、データ・ソースは指定した順序で処理されます。

注： MaxL と ESSCMD を使用して、ファイルごとのアウトラインの確認を強制または抑制できます。アウトラインが有効であることを確認するために、最終の構築でアウトラインの検証が行われるようにします。Administration Services 内の再構築が延期された次元構築では、各データ・ソースが処理された後、アウトラインが検証されます。

データをロードする場所の決定

次元構築を行う場合または集約ストレージ・データベースを処理する場合は、この項をスキップします。

親メンバーにデータをロードする場合は、データベースの計算時に子のデータ値が集計されて、親のデータ値が上書きされることがあります。上書きを防ぐ手順は、次のとおりです:

- 可能なかぎり、データを直接親へロードしないでください。
- 親メンバーにデータをロードする必要がある場合は、その親の子の#MISSING値を集計しないように Essbase に指示します。

➤ 集計を設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	計算時の Missing 値の集約	Oracle Essbase Administration Services Online Help
計算スクリプト	SET AGGMISSG	『Oracle Essbase テクニカル・リファレンス』
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM	『Oracle Essbase テクニカル・リファレンス』

この表に示した方法は、子の値が空(#MISSING)である場合にのみ使用できます。子がデータ値を持っている場合、そのデータ値で親のデータ値が上書きされます。989 ページの「#MISSING 値の集計」を参照してください。

注： 動的計算メンバー、動的計算および保管メンバーまたは属性メンバーにはデータをロードできません。たとえば、年次元が動的計算メンバーである場合、このメンバーにはデータをロードできません。かわりに、動的計算メンバーではない Qtr1、Qtr2、Qtr3 および Qtr4 にデータをロードします。

データ・ソース内の欠落フィールドの処理

データ・ロードを実行するには、データ・ソース内の各レコードが同じ数のデータ値フィールドを持っている必要があります。データ値が欠落している場合、データ・ロードは正確に処理されません。

たとえば、次のファイルは、Apr に値がないため無効です:

```
   Actual  Ohio  Sales  Cola
Jan   Feb   Mar   Apr
10    15    20
```

ファイルを修正するには、欠落したフィールドに#MISSING または#MI を挿入します:

```
   Actual  Ohio  Sales  Cola
Jan   Feb   Mar   Apr
10    15    20    #MI
```

310 ページの「空のフィールドへのテキストの配置」を参照してください。

注： 次元フィールドまたはメンバー・フィールドが欠落している場合、Essbase では、以前にその次元フィールドまたはメンバー・フィールドで使用した値が使用されます。328 ページの「次元フィールドまたはメンバー・フィールドの欠落」を参照してください。

ルール・ファイルに余分な空白のフィールドがある場合は、空のフィールドを隣のフィールドと結合します。308 ページの「フィールドの結合」を参照してください。

集約ストレージ・データベースでは、値はレベル 0 のセルのみにロードできます。#MISSING または#MI をデータ・ソース内の値として指定すると、データベース内の対応するセル(存在する場合)が削除されます。集約ストレージ・データベースでのデータ・ロードの相違点については、1052 ページの「集約ストレージ・データベースの準備」を参照してください。

データ・ソースからのレコードのサブセットのロード

データ・ロード時または次元構築時に、データ・ソース内のレコードのサブセットをロードできます。たとえば、レコード 250 から 500 までをロードし、データ・ソースの他のレコードはロードしないように指定できます。

➤ レコードのサブセットをロードするには:

1 **テキスト編集ツールを使用して、データ・ソース内のレコードに番号を付けます。**

2 **レコード番号を含む列を無視するように、ルール・ファイルを設定します。**

307 ページの「フィールドの無視」を参照してください。

3 **ロードするレコード以外のすべてのレコードを除外する、除外条件を定義します。**

たとえば、レコード番号が 250 より小さい列と 500 より大きい列を無視するようにして、該当するすべてのレコードを除外します。303 ページの「レコードの除外」を参照してください。

注: エラー・ログに収まりきらないレコードは除外できません。デフォルトのレコード数の上限は 1000 です。この値は、`essbase.cfg` ファイルに DATAERRORLIMIT を設定することで変更できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

データのロード・オプションの保存と再利用

Administration Services でデータ・ロードと次元の構築を実行し、毎回同じデータ・ソース・ファイルとルール・ファイルを使用する場合は、「データ・ロード」ダイアログ・ボックスに指定したソース・ファイルおよびルール・ファイルのオプションを保存して、後で再利用できます。

デフォルトでは、ファイル・オプションはクライアント側の ARBORPATH ディレクトリ内の指定の XML ファイルに保存されます。保存したオプションのファイルを使用する場合は、このファイルを開きます。保存されたファイルとそのオプションは、現在開いている「データ・ロード」ダイアログ・ボックス内のリストに追加されます。

ヒント: データ・ロード・オプションのファイルはテンプレートとして使用できるので、このファイルを開いてから、任意の指定を変更できます。

Oracle Essbase Administration Services Online Help の、データのロードおよび次元構築に関する項を参照してください。

次元構築およびアウトライン編集時の名前の参照と挿入の最適化

新規ハッシュテーブルの実装により、次元構築およびアウトライン編集時に名前の参照と挿入のパフォーマンスが最適化されます。このハッシュテーブルの実装では、メモリー・フットプリントが増加されます。

ハッシュテーブルを構成するには、ESTIMATEDHASHSIZE 構成設定を使用します。この設定には、メモリーにロードされるメンバー名および別名の推定数を百万単位で指定します。

ESTIMATEDHASHSIZE 構成設定は、ブロック・アプリケーションと集約ストレージ・アプリケーションに適用されます。

データ・ロードおよび次元構築のデバッグ

データ・ソースが正常に Essbase サーバーにロードされなかった場合は、適切なアプリケーションおよびデータベースに接続していて、正しいデータ・ソースをロードしているかどうかを確認します。

それでも問題が解決されない場合は、次のトピックを参照してください。問題が修正されたら、エラー・ログを再ロードすることにより、ロードされなかったレコードを再ロードできます。833 ページの「次元構築およびデータ・ロードのエラー・ログのロード」を参照してください。

Essbase サーバーが使用可能かどうかの確認

サーバーやネットワークではなく Essbase に問題があることを識別しやすくするには、Essbase を使用しないでサーバーにアクセスしてみます。次の項目を確認します:

- サーバーが実行中であるか。

  Essbase を使用しないでサーバーへの接続を試みます。接続できない場合は、システム管理者に問い合せてください。

- Essbase サーバーが実行中であるか。

  Essbase 管理者に問い合せてください。

- クライアントがサーバーに接続しているか。

  Essbase を使用しないで、クライアントからサーバーに接続を試みます。

データ・ソースが使用可能かどうかの確認

Essbase でロードするデータ・ソースを開けない場合は、次の条件を満たしていることを確認します:

- データ・ソースが使用可能な状態である(たとえば、データ・ソースを編集中のユーザーが存在しないかどうか)。

  Essbase では、他のユーザーまたはアプリケーションによってロックされていないデータ・ソースしかロードできません。
- データ・ソースのファイル拡張子が正しい。

  テキスト・ファイルには.txt、ルール・ファイルには.rul の拡張子が必要です。
- データ・ソース名とパス名が正しい。

  スペルに間違いがないか確認します。
- データ・ソースが指定した場所にある。

  他のユーザーがデータ・ソースを移動または削除していないか確認します。
- SQL データ・ソースを使用している場合:
  - 接続情報(ユーザー名、パスワード、データベース名など)が正しい。
  - Essbase を使用しないで SQL データ・ソースに接続できる。

エラー・ログの確認

データ・ロードや次元構築が失敗した場合は、エラー・ログを貴重なデバッグ・ツールとして利用できます。832 ページの「次元構築およびデータ・ロードのエラー・ログとその表示」を参照してください。

エラー・ログが存在しない場合は、次の条件の有無を確認します:

- データ・ロードを実行しているユーザーがエラー・ログを設定した。

  デフォルトでは、ルール・ファイルを使用した場合、Essbase によってエラー・ログが作成されます。
- データ・ソースと Essbase サーバーが使用可能である。

  323 ページの「Essbase サーバーが使用可能かどうかの確認」および 323 ページの「データ・ソースが使用可能かどうかの確認」を参照してください。
- データ・ロード中に Essbase サーバーがクラッシュした。

  この場合は、通常、クライアント側でタイムアウト・エラーを受け取ります。325 ページの「Essbase サーバーのクラッシュからのリカバリ」を参照してください。
- アプリケーション・ログが存在する。

  801 ページの「Essbase サーバー・ログおよびアプリケーション・ログ」を参照してください。

エラー・ログがあっても空の場合、Essbase は、ロード中にエラーがあったと見なしません。次の条件の有無を確認します:

- ルール・ファイルに、データ・ソース内のすべてのレコードを除外するような選択条件または除外条件が定義されている。

303 ページの「レコードの選択」および 303 ページの「レコードの除外」を参照してください。

- ルール・ファイルが有効である。

  298 ページの「データ・ロード・ルール・ファイルが有効であるための要件」および 299 ページの「次元構築ルール・ファイルが有効であるための要件」を参照してください。

Essbase サーバーのクラッシュからのリカバリ

データのロード中にサーバーがクラッシュした場合、Essbase からユーザーにタイムアウト・エラーが送信されます。ユーザーが実行する必要があるリカバリ手順は、実行しているロードのタイプと、「分離レベル」データベースのトランザクション設定によって異なります(861 ページの「分離レベルの理解」を参照):

- データ・ソースの値を上書きする場合、サーバーが再度実行されているときにデータ・ソースを再ロードします。
- データ・ソース内の既存の値に対して加算または減算を行う場合、「分離レベル」は次のように設定します:
  - コミット。サーバーが再度実行されているときにデータ・ソースを再ロードします。
  - アンコミット。クラッシュ前に Essbase によってロードされたデータの量を判定します:
    1. データ・ソース内の値をデータベース内の値と比較します。
    2. 加算または減算の対象である値が変更されていない場合は、データ・ソースを再ロードします。
    3. 加算または減算の対象である値が変更されている場合は、ロード済の値を消去し、以前のデータ・ソースを再ロードします。たとえば、ロード時に毎週の売上高を加算することによって毎月の売上高を導き出している場合、データベースの売上高を消去し、今週までの各週の売上高を再ロードします。

正しくないデータがロードされた場合

データ・ソースはエラーなしでロードされたが、データベース内のデータが正しくない場合は、次の条件の有無を確認します:

- 正しいデータ・ソースをロードした。

  正しいデータ・ソースをロードした場合、そのデータ・ソースを再度確認し、正しい値が含まれているかどうかを確認します。

- データ・ソース内に空白フィールドがある。

  値が入っていないデータ・フィールドには、#MI または#MISSING を挿入する必要があります。そうしないと、データ・ソースが正しくロードされないことがあります。ルール・ファイルを使用して空白フィールドを#MI または #MISSING で置き換える方法については、310 ページの「空のフィールドへのテキストの配置」を参照してください。

- データ・ソースが正しくフォーマットされている。
  - すべての範囲が適切に設定されている。
  - データが適切である。たとえば、データ・ソースの処理時に、Essbase でメンバー名とメンバーが属する次元が認識されます。データ・ソース・レコードに、ヘッダー・レコード内にメンバー名が指定されている次元メンバーが誤って含まれている場合、その次元のヘッダー・レコードのメンバーが新規メンバー名で置き換えられます。次の例のデータ・ソースでは、Essbase で、Florida は市場次元のメンバーとして認識されます。最後の 4 つのレコードの値は、Texas の値ではなく Florida の値であると解釈されます。

```
Jan  Actual  Texas  Sales
 "100-10"  51.7
 "100-20"  102.5
 "100-20"  335.0
 Florida  96.7
 "200-20"  276.0
 "200-20"  113.1
 "200-10"  167.0
```

- 気付かないうちに暗黙的に共有された(親と子が同じデータ値を共有する)メンバーがある。

  この状況は、ある親が子を 1 つしか持っていない場合か、または親にロール・アップされる子が 1 つしかない場合に発生します。

  151 ページの「暗黙的な共有の理解」を参照してください。
- 入力データを既存のデータで置き換えるのではなく、入力データを既存のデータに追加した。

  312 ページの「既存値に対する加算および減算」を参照してください。
- 選択または除外する予定ではなかったレコードを選択または除外した。

  303 ページの「レコードの選択」および 303 ページの「レコードの除外」を参照してください。
- 符号が反転し(たとえば、プラス符号がマイナス符号に反転)、符号の反転を UDA に対して実行したかどうか。

  314 ページの「フィールド符号の反転」を参照してください。
- 消去するつもりのなかったデータの組合せを消去した。

  313 ページの「既存のデータ値の消去」を参照してください。
- 入力データ値のスケーリングが正しくない。

  314 ページの「データ値のスケーリング」を参照してください。
- すべてのメンバー名および別名が 79 文字以下である。

注： データをエクスポートしたり、そのデータに対してレポートを実行したり、スプレッドシートを使用したりすることで、データを確認できます。レポートをエクスポートまたは実行する方法については、第 34 章「レポート・スクリプトの作成」および付録 E「ESSCMD の使用」を参照してください。スプレッドシートを使用している場合は、Oracle Hyperion Smart View for Office User's Guide を参照してください。

ファイル終端マーカーの除外条件の作成

SQL データ・ソースには、データ・ロードや次元構築の失敗につながる、特殊文字で構成されたファイル終端マーカーが含まれていることがあります。この問題を解決するには、問題のあるレコードを除外する除外条件を定義します。

1. SQL データ・ソース内のファイル終端マーカーを検索します。
2. Essbase の検索コマンドを使用して、ファイル終端マーカーの検索方法を決定します。

   ファイル終端マーカーは、1 つ以上の特殊文字で構成されていることがあるため、検索が難しい場合があります。

   Oracle Essbase Administration Services Online Help の、トークンの指定によるフィールドの無視に関する項を参照してください。
3. ファイル終端マーカーを除外する除外条件を定義します。

   Oracle Essbase Administration Services Online Help の「レコードの除外」を参照してください。

Essbase でのルール・ファイルの処理方法の理解

Essbase によるルール・ファイルの初期化方法やデータ・ソースの処理方法を理解することによって、次元構築の問題を突きとめられる場合があります。

Essbase でルール・ファイルを初期化する手順は次のとおりです:

1. 関連付けられたアウトラインと比較してルール・ファイルを検証します。
2. 次元を検証します。このプロセスでは、構築方法とフィールドのタイプに互換性があり、各次元の名前が一意であることが確認されます。メンバー名は一意の名前または共有名である必要があります。
3. ルール・ファイルで定義された新しい次元をアウトラインに追加します。
4. データ・ソースに指定されたヘッダー・レコードが読み取られます。

次に、Essbase では、データ・ロード時または次元構築時に、データ・ソース内の各レコードに対して次の操作が行われます:

1. すべてのレコードにファイル区切り記号が設定されます。
2. ルール・ファイル内に定義されている順序で、データにフィールド操作が適用されます。

   フィールド操作には、テキストと結合を使用したフィールドの結合、移動、分割および作成が含まれます。ルール・ファイル内のフィールド操作の定義

順序を表示する方法については、306 ページの「フィールドに対する操作の実行」を参照してください。

3. Essbase により、各フィールドにすべてのプロパティが適用されます。その結果、field2 に進む前にすべてのプロパティが field1 に適用されます。Essbase では、次の順序でフィールドのプロパティが適用されます:
    1. データ・ロード時に無視されるように設定されたフィールドが無視される
    2. 次元構築時に無視されるように設定されたフィールドが無視される
    3. データ・フィールドにフラグが付加される
    4. フィールド名が適用される
    5. フィールドの世代が適用される
    6. すべての置換が、ルール・ファイル内で定義されている順序で実行される
    7. 先頭および末尾のスペースが削除される
    8. スペースがアンダースコアに変換される
    9. 接尾辞および接頭辞が付加される
    10. データ値がスケーリングされる
    11. テキストが小文字に変換される
    12. テキストが大文字に変換される
4. メンバーまたはメンバー情報、あるいはその両方をアウトラインに追加します。
5. 行をスキップするように設定している場合、Essbase では、ユーザーによって指定された行数がスキップされます。行をスキップするように設定していない場合、最初のレコードから処理されます。
6. Essbase では、ルール・ファイルに定義された条件の順序で選択または除外条件が実行されます。指定された条件に基づいて、データ・ソースの個々のレコードがロードまたは除外されます。

Essbase によるデータ・ロード時の欠落フィールドまたは無効なフィールドの処理方法の理解

次の項では、Essbase によるデータ・ロード時の無効なフィールドの処理方法について説明します。

次元フィールドまたはメンバー・フィールドの欠落

次元フィールドまたはメンバー・フィールドが欠落している場合、Essbase では、以前にその次元フィールドまたはメンバー・フィールドで使用した値が使用されます。以前の値が存在しない場合、データ・ロードは中断されます。

たとえば次のファイルを Sample.Basic データベースにロードする場合、Essbase では、製品次元の Root Beer および Diet Cola を持つレコードを含むすべてのレコードにおいて、Ohio メンバー・フィールドが市場次元にマッピングされます。

```
  Jan  Sales  Actual  Ohio
       Cola     25
       "Root Beer"  50
       "Diet Cola"  19
```

欠落したメンバー・フィールドの値が以前のレコードに含まれない場合、Essbase ではデータ・ロードが停止します。たとえば次のファイルを Sample.Basic データベースにロードしようとした場合、市場次元(前の例の Ohio)が指定されていないため、データ・ロードは停止します。

```
  Jan  Sales  Actual
     Cola     25
     "Root Beer"   50
     "Diet Cola"   19
```

ロードを再開する方法については、833 ページの「次元構築およびデータ・ロードのエラー・ログのロード」を参照してください。

不明なメンバー・フィールド

データ・ロードの実行中に Essbase で不明なメンバー名が検出された場合、Essbase によってレコード全体が除外されます。欠落したメンバー・フィールドのメンバー名が含まれた以前のレコードがある場合、Essbase では次のレコードの処理を続行します。以前のレコードがない場合、データ・ロードは停止します。たとえば次のファイルを Sample.Basic データベースにロードした場合、有効なメンバー名でない Ginger Ale を含むレコードは除外されます。Cola、Root Beer および Cream Soda が含まれるレコードはロードされます。ただし、Ginger Ale が最初のレコードである場合、データ・ロードは停止します。

```
   Jan, Sales, Actual
Ohio  Cola     2
   "Root Beer"  12
   "Ginger Ale"  15
   "Cream Soda"  11
```

注： 次元構築を実行している場合、データベースに新規メンバーを追加できます。318 ページの「データ・ロードまたは次元構築の実行」を参照してください。

ロードを再開する方法については、833 ページの「次元構築およびデータ・ロードのエラー・ログのロード」を参照してください。

無効なデータ・フィールド

データ・ロードの実行中に Essbase で無効なデータ・フィールドが検出されると、データ・ロードが停止します。無効なフィールドより前の読込み済のフィールド

は、すべてデータベースにロードされます。この場合、部分的なデータ・ロードになります。たとえば、次のファイルでは、Essbase で 15-データ値が検出されると、データ・ロードが停止します。Jan と Feb の Sales レコードはロードされますが、Mar と Apr の Sales レコードはロードされません。

```
East  Cola  Actual

Sales     Jan   $10

          Feb   $21

          Mar   $15-

          Apr   $16
```

ロードを続行する方法については、833 ページの「次元構築およびデータ・ロードのエラー・ログのロード」を参照してください。

21 高度な次元構築の概念の理解

この章の内容

この章のすべての例は、Sample.Basic データベースに基づいています。

構築方法の理解

データ・ソース内のデータのタイプに基づいて構築方法を選択します。選択した構築方法によって、Essbase でアウトライン内の次元、メンバーおよび別名を追加、変更または削除するときに使用するアルゴリズムが決まります。

表 43 のガイドラインを使用して、データ・ソースの適切な構築方法を選択します:

表 43　構築方法のガイドライン

各レコード内のデータのタイプ	例	目的の操作	構築方法 [1]	フィールド・タイプ情報
トップダウン・データ 各レコードでは、親の名前、子の名前、その子の子などを指定します。	年、四半期、月	既存の次元およびメンバーのプロパティを変更する	世代参照	各フィールドの世代番号。
ボトムアップ・データ 各レコードでは、メンバーの名前、その親の名前、その親の親の名前などを指定します。	月、四半期、年	• 異なる世代にロール・アップする共有メンバーを作成する • 既存の次元およびメンバーのプロパティを変更する	レベル参照	各フィールドのレベル番号。

各レコード内のデータのタイプ	例	目的の操作	構築方法 [1]	フィールド・タイプ情報
子を後ろに持つ親 各レコードでは、親の名前と新しい子メンバーの名前を、この順序で指定します。ただし、他の情報も指定できます。	Cola、Diet Cola	• 異なる世代にロール・アップする共有メンバーを作成する • 非レベル 0 のメンバーを共有する • 既存の次元およびメンバーのプロパティを変更する	親子参照	フィールドが親であるか子であるか。フィールド番号は 0 です。
新規メンバーのリスト 各データ・ソースには、新規メンバーがリストされています。このデータ・ソースでは、これらのメンバーがアウトラインのどこに属しているかは指定されません。Essbase では、これらのメンバーの追加場所を決定するためのアルゴリズムを提供します。	Jan、Feb、Mar、April	すべてのメンバーを、既存の親(ダミーの親である可能性もある)の子として追加する	指定された親の子として追加	
	800-10、800-20	すべてのメンバーを次元の末尾に追加する	最下位レベルの兄弟として追加	
	800-10、800-20	新規の各メンバーを、類似メンバーが含まれている次元に追加する	一致文字列を持つメンバーに兄弟として追加	
基本次元メンバーとその属性のリスト	Cola 16oz Can、Root Beer 14oz Bottle	属性次元にメンバーを追加し、その追加したメンバーを基本次元の適切なメンバーに関連付ける	ソース・データの構成に依存する世代、レベルまたは親子参照	各フィールドの番号。 番号は、基本次元の関連メンバーの世代番号かレベル番号、あるいはゼロのいずれかになります。

[1] レベル参照の構築を使用する場合、そのメンバーと同じ名前の別名は作成できません。この制限は、世代参照の構築方法など、他の構築方法を使用する場合は適用されません。

世代参照の使用

トップダウンのデータ・ソースは、左から右の順で、最上位レベルから最下位レベルへと編成されます。各レコードは最も全般的な情報から始まり、より詳細な情報へと進んでいきます。新規メンバーの名前は、レコードの末尾にあります。トップダウンのデータ・ソースを使用する場合、世代参照の構築方法を使用します。ルール・ファイルに、世代番号とデータ・ソースの各フィールドのフィールド・タイプを指定します。

Essbase では、次元内のメンバーに対し、その次元内でのメンバーの階層的な位置に従って番号が付けられます。この番号を世代参照と呼びます。次元は常に世代 1 です。特定の次元の同じ分岐にあるすべてのメンバーを世代と呼びます。世代には、次元(次元 1)を基準とした位置に従って、トップダウンで番号が付けられます。

たとえば、図 66 で示されているように、製品次元は世代 1 です。製品次元のメンバー 100 は世代 2 になります。100 のメンバーである 100-10 などは世代 3 になります。世代参照の構築方法を使用するには、ルール・ファイルに世代参照番号を指定します。

図 66　世代

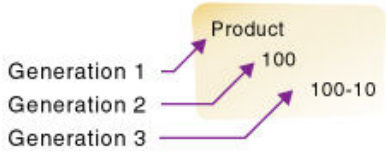


図 67 の上半分は、トップダウンのデータ・ソースです(GENREF.TXT)。このデータ・ソースは、製品次元の構築に使用されます。同じ図の下半分は、データ・ソースのルール・ファイルです(GENREF.RUL)。このルール・ファイルでは、データ・ソース内の各フィールドに世代番号を指定できます。293 ページの「フィールド・タイプ情報の設定」を参照してください。

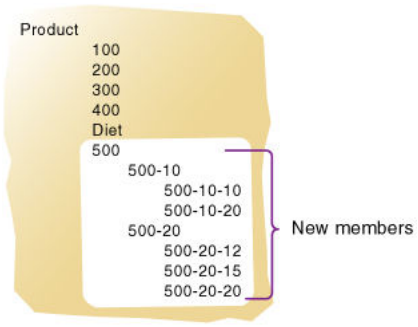
図 67　世代構築のルール・ファイル

```
1    500■500-10■500-10-10
2    500■500-10■500-10-20
3    500■500-20■500-20-12
4    500■500-20■500-20-15
5    500■500-20■500-20-20
```

	GEN2,Product	GEN3,Product	GEN4,Product
1	500	500-10	500-10-10
2	500	500-10	500-10-20
3	500	500-20	500-20-12
4	500	500-20	500-20-15
5	500	500-20	500-20-20

図 68 は、Essbase によって、GENREF.TXT データ・ソースと GENREF.RUL ルール・ファイルから構築されたツリーです:

図 68　世代参照



空フィールドの処理

世代参照の構築方法を使用する場合は、NULL 値を処理するように設定できます。NULL 値の処理により、データ・ソースから空フィールド(NULL フィールド)が検出されたときに実行される Essbase のアクションが指定されます。

NULL 値の処理が使用可能になっていない場合、Essbase では NULL 値を持つすべてのレコードが除外され、エラー・ログにエラーが書き込まれます。

NULL 値の処理が使用可能になっている場合、Essbase で NULL は次のように処理されます:

- **欠落フィールド:** Essbase により GENERATION フィールドの位置に NULL があることが検出された場合は、その次の GENERATION フィールドが移動され、欠落フィールドが置き換えられます。

  次の例では、GEN3,Products 列にはフィールドがありません:

```
     GEN2,Products  GEN3,Products  GEN4,Products
100             100-10a
```

  Essbase でこのレコードを読み取ると、GEN4 フィールド(100-10a)が GEN3 に移動します。データ・ソースの外観は、次の例のようになります:

```
     GEN2,Products  GEN3,Products  GEN4,Products
100      100-10a
```

- **セカンダリ・フィールドの前の欠落フィールド:** セカンダリ・フィールドの直前に NULL がある場合、Essbase ではセカンダリ・フィールドは無視されます(セカンダリ・フィールド・タイプには、別名、プロパティ、式、重複した世代、重複した世代の別名、通貨名、通貨のカテゴリ、属性の親、UDA および属性次元の名前があります)。

  次の例では、GEN2,Products 列や ALIAS2, Products 列にフィールドがありません:

```
     GEN2,Products  ALIAS2,Products  GEN3,Products  GEN4,Products
       Cola       100-10     100-10a
```

  Essbase でこのレコードを読み取ると、ALIAS2 フィールドは無視され、GEN3 フィールド(100-10)が GEN2 に移動、GEN4 フィールド(100-10a)が GEN3 に移動します。データ・ソースの外観は、次の例のようになります:

```
     GEN2,Products  ALIAS2,Products  GEN3,Products  GEN4,Products
100-10     Cola       100-10a
```

- **欠落したセカンダリ・フィールド:** セカンダリ・フィールドの位置に NULL がある場合、Essbase によってセカンダリ NULL フィールドは無視され、ロードが続行されます。

  次の例では、ALIAS2,Products 列にフィールドがありません:

```
     GEN2,Products  ALIAS2,Products  GEN3,Products  GEN4,Products
100              100-10     100-10a
```

  Essbase でこのレコードを読み取ると、ALIAS2 フィールドが無視され、その他のフィールドはロードされます。

レベル参照の使用

ボトムアップのデータ・ソースでは、各レコードにより、次元の単一のメンバーが定義されます。定義はメンバーに関する最も詳細な情報から始まり、より全般的な情報になっていきます。通常、レコードでは新規メンバーの名前が指定され、さらにそのメンバーの親の名前、親の親の名前などが指定されます。

レベルはボトムアップの階層構造で定義されています。たとえば、図 69 のアウトラインでは、最下位レベルのメンバーが製品次元の分岐の下部にあります。

図 69　世代番号とレベル番号

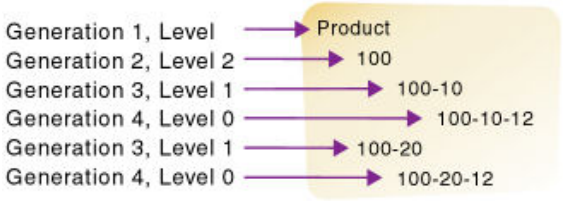


図 69 のアウトラインを構築するには、次のボトムアップのデータ・ソースを使用できます:

```
   100-10-12  100-10  100
100-20-12  100-20  100
```

レベル参照の構築では、最下位レベルのメンバーから、左から右の順で並べられます。レベル 0 メンバーは最初のフィールド、レベル 1 メンバーは 2 番目のフィールドに置かれ、以下同様に続きます。これは、世代参照(トップダウン)のデータ表示とは反対の編成です。

次の例では、ルール・ファイルは、レベル参照の構築方法を使用して製品次元にメンバーを追加しています。ルール・ファイルでは、データ・ソースの各フィールドのレベル番号およびフィールド・タイプを指定します(293 ページの「フィールド・タイプ情報の設定」を参照)。データ・ソースの最初の列は、新規メンバー(600-10-11、600-20-10 および 600-20-18)を含みます。2 番目の列は、新規メンバーの親(600-10 および 600-20)を含み、3 番目の列は親の親(600)を含みます。

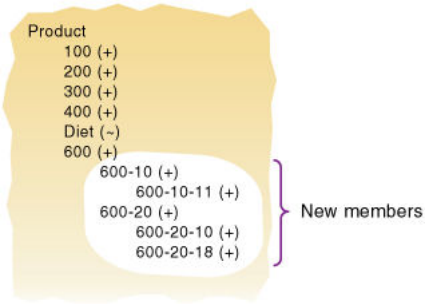
図 70　レベル構築のルール・ファイル

```
1    600-10-11■600-10■600
2    600-20-10■600-20■600
3    600-20-18■600-20■600
```

	LEVEL0,Product	LEVEL1,Product	LEVEL2,Product
1	600-10-11	600-10	600
2	600-20-10	600-20	600
3	600-20-18	600-20	600

たとえば、図 71 のようなツリーを構築するには、図 70 を使用してデータ・ソース(LEVEL.TXT)とルール・ファイル(LEVEL.RUL)を設定します。

図 71　レベル



空フィールドの処理

レベル参照の構築方法を使用する場合は、NULL 値を処理するように設定できます。NULL 値の処理により、データ・ソースから空フィールド(NULL フィールド)が検出されたときに実行される Essbase のアクションが指定されます。

NULL 値の処理が使用可能になっていない場合、Essbase では NULL 値を持つすべてのレコードが除外され、エラー・ログにエラーが書き込まれます。

NULL 値の処理が使用可能になっている場合、Essbase で NULL は次のように処理されます:

- **欠落フィールド:** Essbase により LEVEL フィールドの位置に NULL があることが検出された場合は、Essbase によりその次の LEVEL フィールドが移動され、欠落フィールドが置き換えられます。

  次の例では、LEVEL0,Products 列にフィールドがありません:

  ```
  LEVEL0,Products  LEVEL1,Products  LEVEL2,Products
     100-10       100
  ```

  Essbase でこのレコードを読み取ると、LEVEL1 フィールド(100-10)が LEVEL0、LEVEL2 フィールド(100)が LEVEL1 に移動します。データ・ソースの外観は、次の例のようになります:

  ```
  LEVEL0,Products  LEVEL1,Products  LEVEL2,Products
100-10       100
  ```

- **セカンダリ・フィールドの前の欠落フィールド:** セカンダリ・フィールドの直前に NULL がある場合、Essbase ではセカンダリ・フィールドは無視されます(セカンダリ・フィールド・オプションには、別名、プロパティ、式、重複したレベル、重複したレベルの別名、通貨名、通貨のカテゴリ、属性の親、UDA および属性次元の名前があります)。

  次の例では、LEVEL0,Products 列にフィールドがありません:

  ```
  LEVEL0,Products  ALIAS0,Products  LEVEL1,Products  LEVEL2,Products
     Cola          100-10          100
  ```

Essbase でこのレコードを読み取ると、ALIAS0 フィールドは無視され、LEVEL1 フィールド(100-10)が LEVEL0、LEVEL2 フィールド(100)が LEVEL1 に移動します。データ・ソースの外観は、次の例のようになります:

```
     LEVEL0,Products  ALIAS0,Products  LEVEL1,Products  LEVEL2,Products
100-10       Cola       100
```

- **欠落したセカンダリ・フィールド:** セカンダリ・フィールドの位置に NULL がある場合、Essbase によってセカンダリ NULL フィールドは無視され、ロードが続行されます。

  次の例では、ALIAS0,Products 列にフィールドがありません:

```
     LEVEL0,Products  ALIAS0,Products  LEVEL1,Products  LEVEL2,Products
100-10a      100-10       100
```

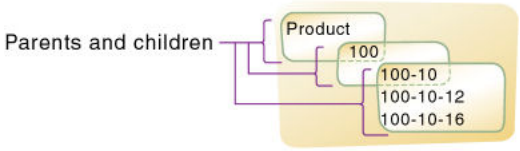
  Essbase でこのレコードを読み取ると、ALIAS0 フィールドが無視され、その他のフィールドはロードされます。

親子参照の使用

データ・ソースのすべてのレコードが、新規メンバーの名前と追加先の親の名前を指定する場合は、親子参照構築方法を使用します。

データベースのメンバーは、互いに親子関係で存在します。図 72 は、製品次元の親と子の関係が明確になっている部分を示しています。Product は 100 の親です。100 は Product の子で、100-10、100-10-12 および 100-10-16 の親でもあります。100-10、100-10-12 および 100-10-16 はすべて 100 の子です。

図 72　親と子



親子データ・ソースには 2 つ以上の列、つまり親の列と子の列がこの順番で含まれている必要があります。データ・ソースには、他の情報(たとえば新規メンバーの別名、属性、プロパティなど)の列も含めることができます。親子データ・ソース内のレコードでは、複数の親または複数の子を指定できません。また、親の列と子の列の順序を逆にすることもできません。

親子構築では、ルール・ファイルで親の列と子の列を指定します。たとえば図 73 の上半分はデータ・ソース(PARCHIL.TXT)で、このデータ・ソース内の各レコードは、親の名前とその子の名前をこの順序で指定しています。図の下半分は、親の列と子の列を指定するルール・ファイルです(PARCHIL.RUL)。さらに、この例では、別名に子フィールドが関連付けられています。

図 73　親子構築のルール・ファイル

```
1    200■200-10■Old Fashioned
2    200■200-20■Diet Root Beer
3    200■200-30■Sasparilla
4    200■200-40■Birch Beer
5    200■200-50■With Caffeine
```

	PARENT0,Product	CHILD0,Product	ALIAS0,Product
1	200	200-10	Old Fashioned
2	200	200-20	Diet Root Beer
3	200	200-30	Sasparilla
4	200	200-40	Birch Beer
5	200	200-50	With Caffeine

図 74 は、Essbase により、このデータ・ソースとルール・ファイルから構築されたツリーです。

図 74　親と子

```
Product
    200
        200-10  Alias: Old Fashioned
        200-20  Alias: Diet Root Beer
        200-30  Alias: Sasparilla
        200-40  Alias: Birch Beer
        200-50  Alias: With Caffeine
```

注：　メンバーが重複している場合、親フィールドに修飾メンバー名を含める必要があります。361 ページの「ルール・ファイルを使用した修飾メンバー名の構築」を参照してください。

新規メンバーのリストの追加

データ・ソースが新規メンバーのリストで構成されていて、その祖先が指定されていない場合、アウトライン内のどこにメンバーを追加するかを Essbase で決定する必要があります。このタイプのデータ・ソースでは、次の構築方法を使用できます:

- 各新規メンバーを、そのテキストが自分のテキストと最もよく一致する既存メンバーの兄弟として追加します。

  339 ページの「文字列の一致に基づくメンバーの追加」を参照してください。

- 各新規メンバーを、最下位レベルの既存メンバーの兄弟として追加します。

  340 ページの「最下位レベルの兄弟としてのメンバーの追加」を参照してください。

- 指定した親(通常はダミーの親)の子として、すべての新規メンバーを追加します。

  341 ページの「メンバーの指定した親への追加」を参照してください。

Essbase ですべての新規メンバーをアウトラインに追加した後、アウトライン・エディタを使用して、新規メンバーを正しい位置に移動する必要が生じることがあります。128 ページの「次元およびメンバーの位置付け」を参照してください。

注：　Essbase では、属性関連付けと「...として追加」の構築方法を同時には使用できません。

文字列の一致に基づくメンバーの追加

文字列と既存のメンバーを一致させることにより、データ・ソースから既存の次元へ新規メンバーを追加できます。Essbase でデータ・ソース内に新規メンバーが見つかった場合、アウトラインで同様のテキストのメンバー名がスキャンされ、新規メンバーは、最も一致率の高い文字列を持つメンバーの兄弟として追加されます。

たとえば、図 75(SIBSTR.TXT)は、製品次元に追加される 2 つの新規メンバー(100-11 および 200-22)を含みます。新規メンバーは、製品次元の文字列と同様に、3 桁の数字、ダッシュおよび 2 桁の数字で構成されます。

例のメンバーをデータベースに追加するには、ルール・ファイル内に表 44 の値を設定します:

表 44　文字列の一致を使用したメンバーの追加の例

フィールド	値	参照
フィールド 1(製品)	• このフィールドにはフィールド・タイプを選択しない • フィールドの次元を製品に設定(図 74 のように、フィールド 1 は製品次元として表示される)	293 ページの「フィールド・タイプ情報の設定」
フィールド 2 からフィールド 6	フィールドを無視	307 ページの「フィールドの無視」
製品次元	「一致する文字列を持つメンバーの兄弟として追加」の構築方法を選択	291 ページの「構築方法の選択」

図 75　一致文字列を持つ兄弟としてメンバーを追加するためのルール・ファイルのフィールド・セット

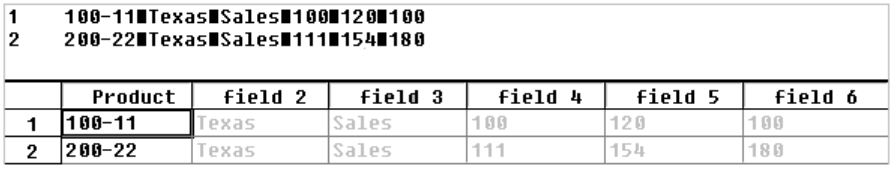
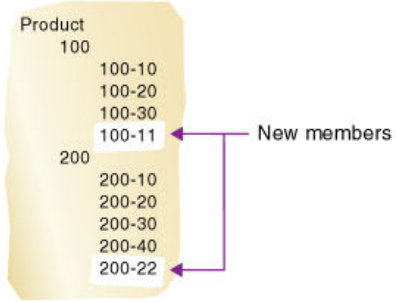


図 76 は、Essbase により、このデータ・ソースとルール・ファイルから構築されたツリーです。100-11 は 100 の兄弟として追加され、200-22 は 200 の兄弟として追加されます。

図 76　一致文字列を持つ兄弟としてメンバーが追加されたツリー



最下位レベルの兄弟としてのメンバーの追加

最下位レベルの次元(レベル 0 の分岐)のメンバーの兄弟として、データ・ソースから新規メンバーを追加できます。Essbase でデータ・ソース内に新規メンバーが見つかった場合、アウトラインでレベル 0 の分岐のメンバーがスキャンされます。新規メンバーは、このメンバーの兄弟として追加されます。

注： アウトラインに、このレベルのメンバーの複数のグループが含まれている場合は、Essbase によって、最初に検出されたグループに新規メンバーが追加されます。

たとえば、図 77 の、データ・ソース(SIBLOW.TXT)およびルール・ファイル(SIBLOW.RUL)は、メジャー次元に追加する新規メンバー(A100-10 および A100-99)を含みます。

図 77　最下位レベルの兄弟としてメンバーを追加するためのルール・ファイルのフィールド・セット

```
1    100-10■Texas■A100-10■■100■120■100
2    200-20■Texas■A100-99■■111■154■180
```

	field 1	field 2	Measures	field 4	field 5	field 6
1	100-10	Texas	A100-10	100	120	100
2	200-20	Texas	A100-99	111	154	180

例のメンバーをデータベースに動的に追加するには、ルール・ファイル内に表 45 の値を設定します:

表 45　最下位レベルの兄弟としてのメンバーの追加の例

フィールド	値	参照
フィールド 3(メジャー)	• このフィールドにはフィールド・タイプを選択しない • フィールドの次元をメジャーに設定する(図 77 のように、フィールド 3 はメジャーとして表示される)	293 ページの「フィールド・タイプ情報の設定」
フィールド 1、2、4、5 および 6	フィールドを無視	307 ページの「フィールドの無視」
メジャー次元	「最下位レベルの兄弟として追加」の構築方法を選択	291 ページの「構築方法の選択」

図 78 は、Essbase により、このデータ・ソースとルール・ファイルから構築されたツリーです。A100-20 および A100-99 は Margin の兄弟として追加されます。

図 78　最下位レベルの兄弟としてメンバーが追加されたツリー

メンバーの指定した親への追加

すべての新規メンバーを、指定された親(通常はダミーの親)の子として追加できます。Essbase ですべての新規メンバーをアウトラインに追加した後、追加されたメンバーを確認して、アウトライン・エディタで移動または削除します。

Essbase でデータ・ソース内に新規メンバーが検出されると、これらのメンバーはユーザーが定義した親の子として追加されます。親は、次元構築を開始する前にアウトラインに追加しておく必要があります。

たとえば、図 79 のデータ・ソース(SIBPAR.TXT)には、製品次元(フィールド 1)の 2 つの新規メンバー(600-54 と 780-22)が含まれています。以前に、製品次元の下に NewProducts というメンバーを追加したとします。

図 79　指定された親の子としてメンバーを追加するためのルール・ファイルのフィールド・セット

```
1    600-54■Texas■Sales■100■120■100
2    780-22■Texas■Sales■111■154■180
```

	Product	field 2	field 3	field 4	field 5	field 6
1	600-54	Texas	Sales	100	120	100
2	780-22	Texas	Sales	111	154	180

データベースの NewProducts メンバーの下に例のメンバーを追加するには、ルール・ファイル内に表 46 の値を設定します:

表 46　指定された親の子としてのメンバーを追加する例

フィールド	値	参照
フィールド 1(製品)	• このフィールドにはフィールド・タイプを選択しない • フィールドの次元を製品に設定する(図 79 のように、フィールド 1 は製品として表示される)	293 ページの「フィールド・タイプ情報の設定」
フィールド 2 からフィールド 6	フィールドを無視	307 ページの「フィールドの無視」
製品次元	「子として追加」の構築方法を選択	291 ページの「構築方法の選択」 「子として追加」テキスト・ボックスに NewProducts と入力します。

図 80 は、Essbase により、このデータ・ソースとルール・ファイルから構築されたツリーです。600-54 および 780-22 は NewProducts の兄弟として追加されます。

図 80　指定された親の子としてメンバーが追加されたツリー

属性次元の構築と属性の関連付け

データ・ソースに属性情報が含まれている場合は、1 つ以上のルール・ファイルを使用して、属性次元の構築と基本次元のメンバーへの属性の関連付けを行う必要があります。

ルール・ファイルを使用して、属性次元の動的構築、メンバーの追加や削除、および属性の関連付けの設定や変更を行うことができます。

属性を使用するには、次の操作が必要です:

- 基本次元が存在しない場合は、まず基本次元を構築する必要があります。
- 属性次元を構築する必要があります。
- 基本次元のメンバーを、属性次元のメンバーに関連付ける必要があります。

これらの操作を実行するために、次のいずれかの方法を使用できます:

- 基本次元と属性次元の構築および関連付けを同時に行います。この場合、単一のルール・ファイルを使用して基本次元と 1 つ以上の属性次元を構築し、各属性を基本次元の適切なメンバーに関連付けます。この方法は、単一のルール・ファイルを使用する最も簡単な方法です。基本次元が存在せず、ソース・データの各レコードに基本次元の各メンバーの属性情報がすべて含まれている場合は、この方法を使用します。
- 属性次元を構築した後、1 つのルール・ファイルで関連付けを行います。基本次元は別の手順で構築したか、またはすでに存在しているものとします。この場合、属性次元の構築と、その属性を基本次元のメンバーと関連付ける処理をワン・ステップで実行できます。このためには、ルール・ファイルに属性関連付けを定義するだけです。343 ページの「属性の関連付け」を参照してください。
- 属性次元を構築した後、別のルール・ファイルを使用して関連付けを行います。基本次元は別の手順で構築したか、またはすでに存在しているものとします。この場合、属性次元の構築と、その属性を基本次元のメンバーと関連付ける処理を別々のステップで実行できます。属性次元を構築した後、属性メンバーを基本次元のメンバーと関連付けます。この方法は、異なるサイズの範囲を表すメンバーを持つ、マルチレベルの数値属性次元を構築するときに使用します。

次の項では、属性次元の構築方法を説明します。

属性次元の構築

データベース内に属性次元を構築する前に、アウトラインで使用する属性メンバー名のフォーマットを定義する必要があります。173 ページの「属性次元におけるメンバー名の設定」を参照してください。

属性次元は、次のいずれかの方法で構築できます:

- 標準次元と同じ構築方法。

  273 ページの「データ・ロードおよび次元構築のプロセス」を参照してください。

- 基本次元のメンバーに属性を関連付けるのと同時に構築する方法。

  343 ページの「属性の関連付け」を参照してください。

Essbase では、属性関連付けと「...として追加」の構築方法を同時には使用できません。

属性次元の構築用のルール・ファイルを定義する際、基本次元と、属性次元ファイルの名前を指定します。

属性の関連付け

属性次元を構築すると同時にその属性メンバーを基本次元のメンバーに関連付けるか、あるいは個別の手順でこれらの作業を実行するかにかかわらず、この項での説明に従ってフィールドを定義します。

注： マルチレベルの属性次元を扱う場合、または数値、ブールあるいは日付のタイプの属性次元を扱う場合、ルール・ファイルに追加フィールドが必要になります。345 ページの「マルチレベルの属性次元の扱い」を参照してください。

ソース・データの各レコードに、2 つ以上の列が含まれている必要があります。1 つは基本次元のメンバー用の列、もう 1 つは基本次元メンバーの属性値用の列です。同じソース・データ・レコードに、基本次元のメンバーと関連付けるその他の属性の列を追加で含めることができます。属性次元のメンバーのフィールドの前に、基本次元のメンバーのフィールドを配置する必要があります。

属性次元メンバーのフィールド・タイプを属性次元の名前として定義し、関連付けられた基本次元のメンバーの世代番号またはレベル番号を使用して、基本次元名を指定します。たとえば、図 81 の ATTRPROD.RUL ファイルでは、フィールド定義 Ounces3,Product によって、このフィールドに Ounces 属性次元のメンバーが含まれることを指定しています。このフィールドの各メンバーは、基本次元である製品次元の世代 3 メンバーとして定義されたデータ・フィールドに関連付けられています。このフィールド定義に基づいて、Essbase では、属性 64 がメンバー 500-10 に関連付けられます。

図 81　属性の関連付けのルール・ファイル

```
1    500■500-10■64■True
2    500■500-20■64■False
```

	GEN2,Product	GEN3,Product	Ounces3,Product	Caffeinated3,Product
1	500	500-10	64	True
2	500	500-20	64	False

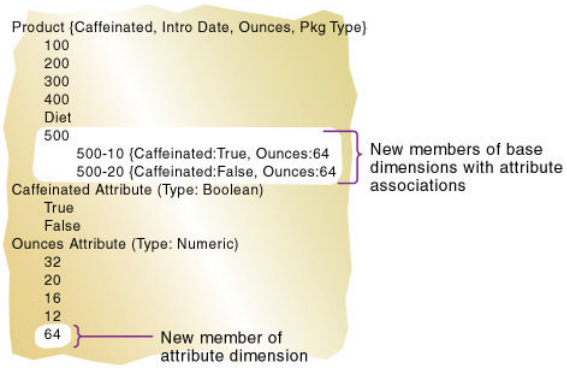
Essbase では、属性列を使用して、属性次元のメンバーを構築できます。データ準備エディタの「次元構築の設定」ダイアログ・ボックスの「次元構築の設定」タブで、基本次元に対して「メンバーを作成しない」オプションを選択解除します。Oracle Essbase Administration Services Online Help の「メンバー・プロパティの設定」を参照してください。

数値の範囲を扱う場合は、属性次元の構築と関連付けを別々の手順で行う必要が生じることがあります。347 ページの「数値範囲の扱い」を参照してください。

図 81 の Caffeinated3,Product フィールドは、追加の単一レベルの属性次元からの属性関連付けの方法を示しています。基本次元はすでに指定されているので、基本次元のメンバーと関連付ける各属性に対して追加フィールドを定義するだけです。

図 81 のファイルは、図 82 のアウトラインに示すように属性を関連付けます。メンバー 64 は Ounces 属性次元の新規メンバーです。メンバー 500、500-10 および 500-20 は、基本次元である製品の新規メンバーで、メンバー 64 に関連付けられています。

図 82　属性の関連付け



属性の関連付けの更新

図 81 のルール・ファイルを使用して、属性関連付けを変更することもできます。関連付けの変更を許可していることを確認してください。データ準備エディタの「次元構築の設定」ダイアログ・ボックスの「次元構築の設定」タブで、基本次元に対して「関連付けの変更可能」を選択します。Oracle Essbase Administration Services Online Help の「メンバー・プロパティの設定」を参照してください。

注：　メンバーが重複している場合、属性を関連付けるフィールドに修飾メンバー名を含める必要があります。361 ページの「ルール・ファイルを使用した修飾メンバー名の構築」を参照してください。

属性関連付けの削除

属性関連付けを削除するときは、属性関連付けを更新するときと同じプロセスを実行した後、さらに次の手順を実行します:

- 「フィールド・プロパティ」ダイアログ・ボックスの「次元構築プロパティ」タブで、属性フィールドに対して「フィールドが空の場合に削除」を選択します(このオプションは、「関連付けの変更可能」が選択されていない場合は無視されます)。
- データ・ソースのフィールドは空または NULL のままにしておきます。

マルチレベルの属性次元の扱い

マルチレベル、数値、ブールおよび日付の属性次元には、重複したレベル 0 メンバーを含めることができます。たとえば、製品次元と関連付けて、2 つのレベルの Size 属性次元を作成できます。レベル 1 では男性と女性でサイズを分類します。レベル 0 メンバー(属性)は実際のサイズです。Women と Men それぞれの下に、8 という名前のメンバーを作成できます。

属性がマルチレベル、数値、ブールまたは日付のいずれかの属性次元に含まれる場合、ソース・データには、その属性次元のすべての世代またはレベルの列が含まれている必要があります。ルール・ファイル内で、属性次元のレベルを構成するすべてのフィールドをコピーする必要があります。最初の属性フィールド・セットを定義して、属性次元を構築します。2 番目の属性フィールド・セットを定義して、その属性を適切な基本次元メンバーと関連付けます。正しい属性と確実に関連付けるには、親フィールドをコピーし、このコピーをフィールド・タイプ Attribute Parent として設定することにより、属性フィールドの親フィールドを指示します。

ルール・ファイル内のフィールドの位置は重要です。

- コピー対象の属性次元フィールドまたは関連付けを定義するフィールドを、基本次元メンバーのフィールドの右隣に置きます。
- マルチレベルの属性次元の場合、属性の親フィールドをその子フィールドの左隣に置きます。

次の手順では、マルチレベルの属性次元を構築したり、そのメンバーを基本次元のメンバーに関連付けたりするために、ルール・ファイル内でどのようにフィールドを定義するかを説明します。この例では、レベル参照の構築方法を使用します。

1. ルール・ファイル内のフィールド 1 とフィールド 2 に対し、標準次元を定義する場合と同じ方法で属性次元フィールドを定義し、タイプ(レベルまたは世代)、番号および次元名を指定します。

   Essbase では、フィールド 1 とフィールド 2 を使用して、属性次元が構築されます。

2. 基本次元を構築するためのフィールドを定義します。

   次の例では、製品次元のレベル 0 とレベル 1 のフィールドを定義します。図 83 は、この段階でのルール・ファイルのフィールドを示します。

図 83　関連付けフィールドを追加する前のマルチレベルの属性次元の定義

```
1    7■Women■100-A23■100
2    8■Women■100-B54■100
3    8■Men■300-R89■300
4    10■Men■300-U65■300
5    9■Men■400-J43■400
```

	LEVEL0,Size	LEVEL1,Size	LEVEL0,Product	LEVEL1,Product
1	7	Women	100-A23	100
2	8	Women	100-B54	100
3	8	Men	300-R89	300
4	10	Men	300-U65	300
5	9	Men	400-J43	400

3. 関連付けを定義するには、レベル 0 の属性を含むフィールドをコピーします。

   この例では、フィールド 1 をコピーします。

   1. フィールド・タイプに属性次元名を使用して、Essbase が属性(たとえば Size0)を関連付ける、基本次元メンバーの世代番号またはレベル番号を指定します。
   2. 基本次元(たとえば、Product)を指定します。
   3. Essbase が属性を関連付ける、基本次元のフィールドの右隣に新規のフィールドを移動します。

      この例では、フィールド Level0,Product の右に新しいフィールドを移動します。

4. 属性フィールドの親を含むフィールドをコピーします。

   この例では、フィールド 2 をコピーします。

   1. この新規フィールドのフィールド・タイプを「属性の親」として設定し、Essbase が属性(たとえば ATTRPARENT0)を関連付ける基本メンバーの世代番号またはレベル番号を指定します。
   2. 属性次元(たとえば Size)を指定します。
   3. 手順 3 で作成した属性の関連付けフィールドの左隣に ATTRPARENT フィールドを移動します。

図 84 のように、ルール・ファイルにフィールド定義が追加されました。このフィールド定義によって、属性次元 Size を構築し、Size のメンバーに基本次元である Product 次元の適切なメンバーを関連付けることができます。

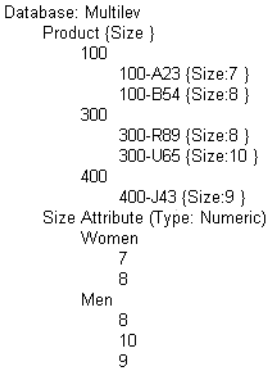
図 84　マルチレベルの属性次元を構築するソース・データとルール・ファイル

```
1    7■Women■100-A23■100
2    8■Women■100-B54■100
3    8■Men■300-R89■300
4    10■Men■300-U65■300
5    9■Men■400-J43■400
```

	LEVEL0,Size	LEVEL1,Size	LEVEL0,Product	ATTRPARENT0,Size	Size0,Product	LEVEL1,Product
1	7	Women	100-A23	Women	7	100
2	8	Women	100-B54	Women	8	100
3	8	Men	300-R89	Men	8	300
4	10	Men	300-U65	Men	10	300
5	9	Men	400-J43	Men	9	400

図 84 のデータを使用して次元構築を実行すると、Essbase により、Size 属性次元が構築され、そのメンバーが基本次元の適切なメンバーに関連付けられます。図 85 は、更新されたアウトラインです。

図 85　マルチレベルの属性次元



数値範囲の扱い

多くの場合は、単一のルール・ファイルを使用して、1 回の次元構築操作で、数値範囲の属性次元を動的に構築し、基本次元のメンバーを範囲と関連付けることができます。しかし、次のような場合は、属性次元の構築用のルール・ファイルと、属性を基本次元の適切なメンバーに関連付けるためのルール・ファイルの合計 2 つのルール・ファイルを使用する必要があります:

- 各メンバーに対する範囲のサイズが異なる場合。

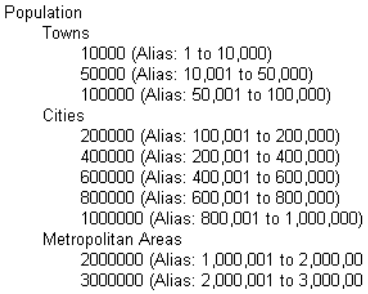
  たとえば、比較的人口の少ない町や市に対して狭い範囲を定義し、中規模の人口の市に対して比較的広い範囲を定義し、大規模な人口の市に対して 1,000,000 より大きい範囲を定義するような場合です。

- 範囲がマルチレベルの属性次元のメンバーである場合。

  たとえば、Towns、Cities および Metropolitan Areas として人口の範囲が分類されるレベル 1 のメンバーを、Population 属性次元が持つような場合です。

図 86 の Population 属性次元は、両方の場合を示しています。Population は、マルチレベルの数値属性次元で、レベル 0 メンバーを持ち、異なるサイズの範囲を表します。

図 86　異なるサイズの範囲を持つ数値属性次元



1 つのルール・ファイルで Population 次元を構築し、別のルール・ファイルで Population 次元のメンバーを、その基本次元のメンバーの属性として関連付ける必要があります。

範囲に対応する属性次元の構築

最初に、属性次元を構築するため、世代、レベルまたは親子構築方法を使用するルール・ファイルを作成します。このルール・ファイル内に、次の情報を指定します:

- 属性次元の名前とその関連基本次元の名前。
- 属性次元を構築するためのフィールド。

293 ページの「フィールド・タイプ情報の設定」を参照してください。

ソース・データは、昇順の属性順になっている必要があります。範囲のサイズが異なる場合は、ソース・データに各属性範囲のレコードを含める必要があります。

注： 後の構築操作で、属性メンバーを既存のメンバーの間に挿入することはできません。

世代を使用する方法で図 86 のアウトラインを構築するには、数値属性値に基づいてソース・データを昇順に配列する必要があります。図 87 のルール・ファイル内にフィールドを定義します。図 87 は、別名と属性を関連付ける方法も示しています。

図 87　範囲を持つ数値属性次元を構築するルール・ファイル

```
1    Towns■10000■<=10,000
2    Towns■50000■10,001 to 50,000
3    Towns■100000■50,001 to 100,000
4    Cities■200000■100,001 to 200,000
5    Cities■400000■200,001 to 400,000
6    Cities■500000■400,001 to 600,000
7    Cities■800000■600,001 to 800,000
8    Cities■1000000■800,001 to 1,000,000
9    Metropolitan Areas■2000000■1,000,001 to 2,000,000
10   Metropolitan Areas■3000000■2,000,001 to 3,000,000
```

	GEN2,Population	GEN3,Population	ALIAS3,Population
1	Towns	10000	<=10,000
2	Towns	50000	10,001 to 50,000
3	Towns	100000	50,001 to 100,000
4	Cities	200000	100,001 to 200,000
5	Cities	400000	200,001 to 400,000
6	Cities	500000	400,001 to 600,000
7	Cities	800000	600,001 to 800,000
8	Cities	1000000	800,001 to 1,000,000
9	Metropolitan Areas	2000000	1,000,001 to 2,000,000
10	Metropolitan Areas	3000000	2,000,001 to 3,000,000

基本次元メンバーの範囲属性への関連付け

数値属性次元範囲を構築した後、ルール・ファイルを使用して、基本次元のメンバーとその属性を関連付ける必要があります。ソース・データには、基本次元のメンバー用のフィールドとデータ値用のフィールドがあります。Essbase では、これらのフィールドを使用して適切な Population 属性を関連付けます。

図 88 のように、ルール・ファイルを定義します。

図 88　数値範囲属性の関連付けのルール・ファイル

```
1    South■Albany, GA■117286
2    East■Boston, MA■3227707
3    East■Hartford, CT■1144574
4    West■Oakland, CA■2209629
5    Central■Rapid City, SD■87145
6    Central■St. Joseph, MO■97336
7    West■Tacoma, WA■657272
```

	GEN1,Market	GEN2,Market	Population3,Market
1	South	Albany, GA	117286
2	East	Boston, MA	3227707
3	East	Hartford, CT	1144574
4	West	Oakland, CA	2209629
5	Central	Rapid City, SD	87145
6	Central	St. Joseph, MO	97336
7	West	Tacoma, WA	657272

関連付けフィールドを定義する際(たとえば、Population3,Market)、範囲内に属性メンバーを配置します。データ準備エディタの「フィールド・プロパティ」ダイアログ・ボックスの「次元構築プロパティ」タブで、「属性メンバーを範囲内に配置」を選択します。

注： 図 88 にある人口 3,227,707 の都市 Boston は、図 86 の属性次元の範囲を超えています。この属性次元の範囲は、3,000,000 までしか拡張できません。ソース・データ内の属性次元の範囲に含まれない値を許可するには、1000000 のような範囲サイズを入力します。Essbase では、この範囲サイズを使用して、既存の最も値が大きいメンバーを超える属性次元、または最も値が小さいメンバーに満たない属性次元にメンバーを追加できます。

注意　基本次元のメンバーと属性次元のメンバーを関連付けた後、属性次元に新規メンバーを挿入したり、属性次元のメンバー名を変更したりする処理を手動で行う場合は、既存の属性関連付けを無効にする必要が生じることがあります。数値範囲属性が「範囲の一番上」として定義されていて、100、200、500 および 1000 のメンバーを含む属性次元があるとします。この場合、値 556 の基本次元メンバーは、属性 1000 に関連付けられます。属性次元メンバーの名前を 500 から 600 に変更した場合、値 556 の基本次元メンバーの関連付けは無効になります。この基本メンバーはまだ属性 1000 に関連付けられていますが、属性 600 に関連付ける必要があります。新規メンバーの挿入や、既存のメンバーの名前の変更を手動で行う場合、関連付けが正しいことを確認するため、次元構築手順を再実行して、基本メンバーと変更された属性次元を関連付けます。たとえば、属性関連付けの手順を再実行すると、値 556 の基本次元のメンバーが新しい属性 600 に正しく関連付けられます。

関連付けの妥当性の確立

属性関連付けの妥当性を確立するには、正しい次元構築オプションを選択し、適切な順序で構築を実行する必要があります。

- **属性次元のメンバーの追加または変更:** 基本次元のメンバーに数値属性範囲を関連付けた後、属性次元への新規メンバーの挿入または属性次元の既存メンバーの名前変更を手動で実行する場合は、次のいずれかの作業を実行して、属性と基本メンバーを正しく関連付けます:

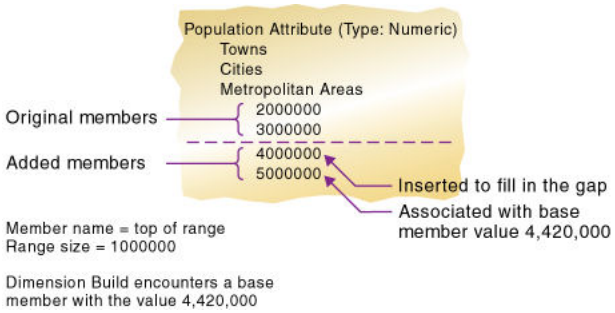
- 基本メンバーを変更後の属性次元に関連付ける、次元構築手順を再実行します。
- アウトライン・エディタを使用して、手動ですべての基本次元の関連付けを見直し、必要な場合は修正を行います。

- **属性次元のメンバーの削除:** 新しいデータで次元を再構築するために、属性次元のメンバーをすべて削除できます。データ準備エディタの「フィールド・プロパティ」ダイアログ・ボックスにある「次元構築プロパティ」タブで、「範囲」ボタンをクリックし、「この属性次元の全メンバーを削除」を選択します。Essbase では、開始値と範囲サイズ値を使用して、属性次元が再構築されます。適切な属性関連付けを確認するには、「次元構築の設定」ダイアログ・ボックスの「次元構築の設定」タブで、基本次元に対して「関連付けの変更可能」オプションを選択する必要があります。
- **基本次元へのメンバーの追加:** 同じルール・ファイルを使用して、基本次元に新規メンバーを追加すると同時に、これらの新規メンバーを数値範囲属性に関連付けることができます。範囲サイズの値を指定します。データ準備エディタの「フィールド・プロパティ」ダイアログ・ボックスの「次元構築プロパティ」タブで、「範囲」ボタンをクリックし、属性次元の範囲サイズを指定します。

  Essbase で、最上位の属性メンバーより大きく、かつその差が範囲サイズを超えるような基本次元値、または最下位の属性メンバーより小さく、かつその差が範囲サイズを超えるような新規の基本次元値が検出された場合、属性次元内で範囲外の値に対応するメンバーが作成されます。

  たとえば、図 89 では、数値範囲属性が「範囲の一番上」と定義されています。Population 属性次元で一番値が大きいメンバーは、3000000 です。ソース・データに人口 4,420,000 のレコードが含まれていて、範囲サイズが 1000000 の場合、Essbase では属性次元に 4000000 と 5000000 の 2 つのメンバーが追加され、値 4,420,000 の基本メンバーと属性 5000000 が関連付けられます。

図 89　動的な属性範囲メンバーの追加



範囲メンバーと基本次元メンバーを同時に追加した場合、Essbase では属性次元の新規メンバーの別名は作成されません。属性次元の新規メンバーの範囲値を説明する別名が必要な場合は、別名を別途追加する必要があります。

属性および基本次元の構築に関するルールの確認

この項では、次元構築によって属性の定義と関連付けを行う方法に固有の領域について説明します。

準備

- 次元構築を実行する前に、アウトラインに属性メンバー名のフォーマットを定義する必要があります。

  173 ページの「属性次元におけるメンバー名の設定」を参照してください。

- ルール・ファイルで新規の属性次元を定義することは、ルール・ファイルで新規の標準次元を定義することとは異なります。

ルール・ファイル内のフィールドの定義

単一レベルの属性次元の構築に使用されるルール・ファイルは、マルチレベルの属性次元のメンバーの構築や関連付けを行うルール・ファイルより、少ないフィールド・タイプで十分です。

- 単一レベルの属性次元の場合、属性値が含まれるフィールドを基本次元のメンバーと関連付けるフィールドとして定義します。次元構築では、この定義済フィールドを使用して、属性次元に新規メンバーを追加します。

  343 ページの「属性の関連付け」を参照してください。

- マルチレベルの属性次元の場合、Essbase には、属性次元内の各世代またはレベルを定義するフィールドと、関連付けを定義するフィールドが必要です。新しいフィールド・タイプ Attribute Parent を使用して、関連付けの対象となる、属性メンバーの親メンバーになるフィールドを識別します。

  345 ページの「マルチレベルの属性次元の扱い」を参照してください。

新規属性メンバーの追加の制御

Essbase で、属性次元のメンバーではない属性データ値が検出された場合、その値は自動的に新規メンバーとして追加されます。属性次元に新規メンバーが追加されないようにするには、「次元構築の設定」ダイアログ・ボックスの「次元構築の設定」タブで、属性次元に対して「メンバーを作成しない」オプションを選択します。

Oracle Essbase Administration Services Online Help の「メンバー・プロパティの設定」を参照してください。

関連付けの制御

次の関連付けを制御できます:

- 属性の関連付けに対する変更

  データ準備エディタの「次元構築の設定」ダイアログ・ボックスの「次元構築の設定」タブで、属性次元に対して「関連付けの変更可能」オプションを選択します。

  Oracle Essbase Administration Services Online Help の「メンバー・プロパティの設定」を参照してください。

- 値の範囲を表す属性による、基本メンバーの自動関連付けの有効化

  データ準備エディタの「フィールド・プロパティ」ダイアログ・ボックスの「次元構築プロパティ」タブで、「範囲」ボタンをクリックして、範囲のサイズを定義します。

  293 ページの「フィールド・タイプ情報の設定」を参照してください。

- 属性の同時関連付け

  「...として追加」以外の構築方法を使用します。

  331 ページの「構築方法の理解」を参照してください。

注： 属性はアウトラインでのみ定義されるため、データ・ロード・プロセスによる属性への影響はありません。

ルール・ファイルを使用した共有メンバーの作成

共有メンバーに関連付けられたデータは、共有メンバーと同じ名前を持つ実際のメンバーのデータです。共有メンバーには、実際のメンバーのデータへのポインタが保管されます。このため、データはメンバー間で共有され、1 回のみ保管されます。

たとえば、100-20(Diet Cola)メンバーは 100 ファミリと Diet ファミリにロール・アップされます。

図 90　Sample.Basic データベースの共有メンバー

メンバーを共有する親の数に制限はありません。Diet Cola には親が 2 つありますが(100 と Diet)、ロール・アップ先の親を追加で定義することもできます。

アウトライン内の複数世代でメンバーを共有できます。図 90 の Diet Cola は、アウトライン内の世代 2 の 2 つのメンバーに共有されていますが、図 95 のように世代 3 と世代 4 のメンバーで共有することもできます。

アウトライン内の異なる世代における共有メンバーは、アウトライン・エディタを使用すると簡単に作成できます。次元構築を使用して共有メンバーを作成する方法は、より複雑です。構築方法を選択し、データ・ソースを慎重にフォーマットする必要があります。

次の項では、データ・ソースとルール・ファイルを使用してアウトライン内に共有メンバーを構築する方法を説明します。

- 353 ページの「同じ世代でのメンバーの共有」
- 355 ページの「異なる世代でのメンバーの共有」
- 356 ページの「非レベル 0 のメンバーの共有」

- 358 ページの「レベル参照を使用した複数のロールアップの作成」
- 359 ページの「複数のデータ・ソースからの共有ロールアップの作成」

注： 共有メンバーが、関連付けられている実績メンバーの前に置かれるようなアウトラインは、作成しないでください。作成すると、アウトラインの検証でエラーが発生する可能性があります。ただし、次元構築中は「次元構築の設定」-「次元構築の設定」タブで、Essbase により、最上位の共有メンバーをプライマリ・メンバーとし、それまでのプライマリ・メンバーを共有メンバーに設定することでエラーを修正するオプションを選択できます。Oracle Essbase Administration Services Online Help を参照してください。

同じ世代でのメンバーの共有

同じ世代で共有されているメンバーは、同じ分岐にロール・アップされます。100-20 (Diet Cola)は 2 つの親(100 と Diet)によって共有されています。図 91 に示すように、これらの親は同じ世代 2 に属しており、同じ分岐(製品次元)にロール・アップされます:

図 91　サンプル・アウトライン: 同じ世代で共有されるメンバー

```
Product
    100
        100-20
    200
        200-20
    300
        300-20
    400
        400-20
    Diet (~)
        100-20 (+) (Shared Member)
        200-20 (+) (Shared Member)
        300-20 (+) (Shared Member)
        400-20 (+) (Shared Member)
```

同じ世代でメンバーを共有するのは、メンバーを共有する最も簡単な方法で、世代参照、レベル参照または親子参照で実行できます。参照:

- 353 ページの「世代参照による、同世代の共有メンバーの作成」
- 354 ページの「レベル参照による、同世代の共有メンバーの作成」
- 354 ページの「親子参照による、同世代の共有メンバーの作成」

サンプルのデータ・ソース・ファイルおよびルール・ファイルが各構築方法について用意されています。各構築方法の結果は、図 91 に示されています。

世代参照による、同世代の共有メンバーの作成

世代参照構築方法を使用して同じ世代の共有メンバーの親を作成するには、共有メンバーの親のフィールド・タイプを DUPGEN として定義します。重複する世代とは、子のメンバーを共有する世代です。同じ GEN 番号をプライマリ・メンバーとして使用します。

たとえば、Diet という親を作成し、100-20、200-20、300-20 および 400-20 の各メンバーを共有するには、サンプル・データ・ソース・ファイル(SHGENREF.TXT)を使用し、フィールドが図 92 のサンプル・ルール・ファイル(SHGENREF.RUL)と同じ

になるように、ルール・ファイルを設定します。100 は Cola ファミリ、200 は Root Beer ファミリ、300 は Cream Soda ファミリ、ファミリ名の後ろの-20(100-20)は、ダイエット・バージョンのソーダであることを示します。

図 92　サンプル・ルール・ファイル: 世代参照を使用して同じ世代で共有されるメンバー

```
1    100■Diet■100-20
2    200■Diet■200-20
3    300■Diet■300-20
4    400■Diet■400-20
```

	GEN2,Product	DUPGEN2,Product	GEN3,Product
1	100	Diet	100-20
2	200	Diet	200-20
3	300	Diet	300-20
4	400	Diet	400-20

レベル参照による、同世代の共有メンバーの作成

レベル参照構築方法を使用して同じ世代の共有メンバーを作成するには、まずプライマリとセカンダリのロールアップが 1 つのレコードに指定されていることを確認します。セカンダリ・ロールアップは、1 つのレコード内に必要な数だけ指定できます。

共有メンバーのフィールド・タイプを LEVEL として定義します。次に、レベル番号を入力します。同じ世代の共有メンバーを作成するには、セカンダリ・ロールアップのレベル番号にプライマリ・ロールアップと同じレベル番号が含まれるように設定します。データ・ソースの処理を実行すると、Essbase により、指定したレベルで親が作成され、その下に共有メンバーが挿入されます。

たとえば、共有メンバー 100-20 (Diet Cola)、200-20 (Diet Root Beer)、300-20 (Diet Cream Soda)および 400-20 (Diet Fruit Soda)を作成するには、サンプル・データ・ソース・ファイル(SHLEV.TXT)を使用し、フィールドが図 93(SHLEV.RUL)と同じになるようにルール・ファイルを設定します。

図 93　サンプル・ルール・ファイル: レベル参照を使用して同じ世代で共有されるメンバー

```
1    100-20■100■Diet
2    200-20■200■Diet
3    300-20■300■Diet
4    400-20■400■Diet
```

	LEVEL0,Product	LEVEL1,Product	LEVEL1,Product
1	100-20	100	Diet
2	200-20	200	Diet
3	300-20	300	Diet
4	400-20	400	Diet

親子参照による、同世代の共有メンバーの作成

親子参照構築方法を使用して同じ世代の共有メンバーを作成するには、フィールド・タイプ PARENT と CHILD を定義します。Essbase の「次元構築の設定」ダイアログ・ボックスの「次元構築の設定」タブで、「共有しない」オプションを選択解除して、共有を許可する設定にします。共有が許可されている場合、Essbase では、新しい親の下に、共有メンバーとして自動的に重複メンバーが作成されます。

図 94　サンプル・ルール・ファイル: 親子参照を使用して同じ世代で共有されるメンバー

```
1    100■100-20
2    200■200-20
3    300■300-20
4    400■400-20
5    Diet■100-20
6    Diet■200-20
7    Diet■300-20
8    Diet■400-20
```

	PARENT0,Product	CHILD0,Product
1	100	100-20
2	200	200-20
3	300	300-20
4	400	400-20
5	Diet	100-20

異なる世代でのメンバーの共有

場合によっては、アウトライン内の異なる世代の親に共有メンバーをロール・アップする必要があります。たとえば、図 95 では、共有メンバーは世代 2(Diet)と世代 3(TBC と Grandma's)の親にロール・アップされます。このアウトラインでは、TBC(The Beverage Company)が一部の飲料製品を外部ベンダーから購入することを想定しています。200-20(Diet Root Beer)は Grandma's という名前のベンダーから購入します。

図 95　サンプル・アウトライン: 異なる世代で共有されるメンバー

```
Product
    100
        100-20
    200
        200-20
    300
        300-20
    Diet
        100-20 (Shared Member)
        200-20 (Shared Member)
        300-20 (Shared Member)
    Vendors
        TBC
            100-20 (Shared Member)
            300-20 (Shared Member)
        Grandma's
            200-20 (Shared Member)
```

異なる世代でメンバーを共有するのは、レベル参照または親子参照で実行できます。参照:

- 355 ページの「レベル参照による、異なる世代での共有メンバーの作成」
- 356 ページの「親子参照による、異なる世代での共有メンバーの作成」

サンプルのデータ・ソース・ファイルおよびルール・ファイルが各構築方法について用意されています。各構築方法の結果は、図 95 に示されています。

レベル参照による、異なる世代での共有メンバーの作成

レベル参照構築方法を使用して異なる世代の共有メンバーを作成するには、プライマリとセカンダリのロールアップが 1 つのレコードに指定されていることを確認します。セカンダリ・ロールアップは、1 つのレコード内に必要な数だけ指定できます。

共有メンバーのフィールド・タイプを LEVEL として定義します。次に、レベル番号を入力します。データ・ソースの処理を実行すると、Essbase により、指定したレベルで親が作成され、その下に共有メンバーが挿入されます。

たとえば、Diet という親を持つ製品 100-20、200-20 および 300-20 を、TBC と Grandma's の 2 つの親が共有するためには、図 96 のようなサンプル・データ・ファイルとルール・ファイルを使用します:

図 96　サンプル・ルール・ファイル: レベル参照を使用して異なる世代で共有されるメンバー

```
1    100-20■100■Diet■TBC■Vendors
2    200-20■200■Diet■Grandma's■Vendors
3    300-20■300■Diet■TBC■Vendors
```

	LEVEL0,Product	LEVEL1,Product	LEVEL1,Product	LEVEL1,Product	LEVEL2,Product
1	100-20	100	Diet	TBC	Vendors
2	200-20	200	Diet	Grandma's	Vendors
3	300-20	300	Diet	TBC	Vendors

親子参照による、異なる世代での共有メンバーの作成

親子参照構築方法を使用して異なる世代の共有メンバーを作成するには、フィールド・タイプ PARENT と CHILD を定義します。Essbase の「次元構築の設定」ダイアログ・ボックスの「次元構築の設定」タブで、「共有しない」オプションを選択解除して、共有を許可する設定にします。共有が許可されている場合、Essbase では、新しい親の下に、共有メンバーとして自動的に重複メンバーが作成されます。

図 97　サンプル・ルール・ファイル: 親子参照を使用して異なる世代で共有されるメンバー

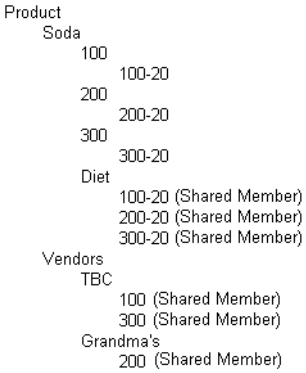
```
1    100■100-20
2    200■200-20
3    300■300-20
4    Diet■100-20
5    Diet■200-20
6    Diet■300-20
7    Vendors■TBC
8    Vendors■Grandma's
9    TBC■100-20
10   Grandma's■200-20
11   TBC■300-20
```

	PARENT0,Produc	CHILD0,Prod
1	100	100-20
2	200	200-20
3	300	300-20
4	Diet	100-20
5	Diet	200-20
6	Diet	300-20

非レベル 0 のメンバーの共有

場合によっては、非レベル 0 のメンバー(最下位でない世代のメンバー)を共有する必要があります。たとえば、図 98 の 100、200 および 300 は、TBC と Grandma's によって共有されています。このアウトラインは、TBC が一部の製品ラインを外部ベンダーから購入することを想定しています。たとえば、200 (すべてのルート・ビール)を Grandma's という名前のベンダーから購入します。

図 98　サンプル・アウトライン: 異なる世代で共有される非レベル 0 のメンバー



非レベル 0 のメンバーの共有は、レベル参照または親子参照で実行できます。参照:

- 357 ページの「レベル参照による非レベル 0 の共有メンバーの作成」
- 358 ページの「親子参照による非レベル 0 の共有メンバーの作成」

サンプルのデータ・ソース・ファイルおよびルール・ファイルが各構築方法について用意されています。各構築方法の結果は、図 98 に示されています。

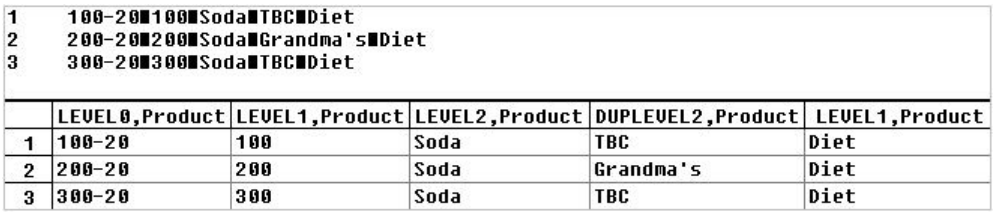
レベル参照による非レベル 0 の共有メンバーの作成

レベル参照構築方法を使用して非レベル 0 の共有メンバーを作成するには、プライマリとセカンダリのロールアップが 1 つのレコードに指定されていることを確認します。セカンダリ・ロールアップは、1 つのレコード内に必要な数だけ指定できます。

共有メンバーの親のフィールド・タイプを重複レベル(DUPLEVEL)として定義します。次に、レベル番号を入力します。同じ世代の共有メンバーを作成するには、セカンダリ・ロールアップのレベル番号にプライマリ・ロールアップと同じレベル番号が含まれるように設定します。データ・ソースの処理を実行すると、Essbase により、指定したレベルで親が作成され、その下に共有メンバーが挿入されます。

たとえば、製品ライン 100、200 および 300 を Soda という親と、TBC および Grandma's という親の間で共有するには、図 99 のサンプル・データ・ファイルおよびルール・ファイルを使用します。このデータ・ソースおよびルール・ファイルは、アウトライン内に Diet、TBC および Grandma's メンバーが存在する場合にのみ機能します。DUPLEVEL フィールドは、指定したレベル・フィールドがアウトライン内にすでに存在する場合を除き、常に世代 2 の次元の子として作成されます。

図 99　サンプル・ルール・ファイル: レベル参照を使用して異なる世代で共有される非レベル 0 のメンバー



```
1    100-20■100■Soda■TBC■Diet
2    200-20■200■Soda■Grandma's■Diet
3    300-20■300■Soda■TBC■Diet
```

	LEVEL0,Product	LEVEL1,Product	LEVEL2,Product	DUPLEVEL2,Product	LEVEL1,Product
1	100-20	100	Soda	TBC	Diet
2	200-20	200	Soda	Grandma's	Diet
3	300-20	300	Soda	TBC	Diet

親子参照による非レベル 0 の共有メンバーの作成

親子参照構築方法は、最も用途の広い共有メンバー作成方法です。この構築方法には、世代参照やレベル参照の構築方法のように、アウトライン内の共有メンバーの位置に制限がありません。

親子参照構築方法を使用して同じ世代の非レベル 0 の共有メンバーを作成するには、フィールド・タイプ PARENT と CHILD を定義します。Essbase の「次元構築の設定」ダイアログ・ボックスの「次元構築の設定」タブで、「共有しない」オプションを選択解除して、共有を許可する設定にします。共有が許可されている場合、Essbase では、新しい親の下に、共有メンバーとして自動的に重複メンバーが作成されます。

図 100　サンプル・ルール・ファイル: 親子参照を使用して同じ世代で共有される非レベル 0 のメンバー

```
1    Soda■100
2    100■100-20
3    Soda■200
4    200■200-30
5    Soda■300
6    300■300-30
7    Diet■100-20
8    Diet■200-20
9    Diet■300-20
10   Vendors■TBC
11   TBC■100
12   TBC■300
13   Vendors■Grandma's
14   Grandma's■200
```

	PARENT0,Product	CHILD0,Product
1	Soda	100
2	100	100-20
3	Soda	200
4	200	200-30
5	Soda	300
6	300	300-30
7	Diet	100-20
8	Diet	200-20
9	Diet	300-20
10	Vendors	TBC

レベル参照を使用した複数のロールアップの作成

複数のパースペクティブから合計を取得できるようにする方法として、レベル参照構築方法を使用することで、アウトライン内の異なるレベルに共有メンバーを配置する方法があります。たとえば、図 101 および図 102 のルール・ファイル (LEVELMUL.RUL)は、製品次元内の各レベルについて構築方法を指定しています:

図 101　サンプル・ルール・ファイル: レベル参照を使用した異なるレベルの複数のロールアップ

```
1    800-10-1■800-10■800■Soda■12 oz.■Cans■Steel■Berthas
2    800-10-8■800-10■800■Soda■8 oz.■Cans■Aluminum■Minis
```

	LEVEL0,Product	LEVEL1,Product	LEVEL2,Product	ALIAS2,Product	LEVEL1,Product	LEVEL2,Product
1	800-10-1	800-10	800	Soda	12 oz.	Cans
2	800-10-8	800-10	800	Soda	8 oz.	Cans

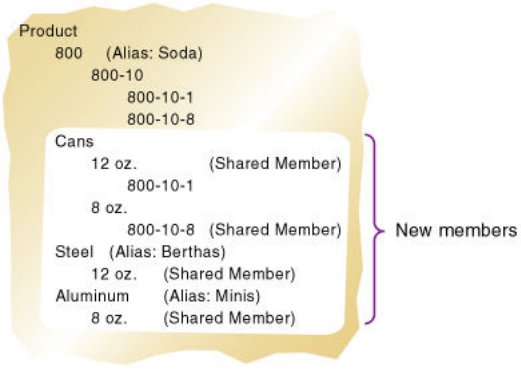
次の図では、ルール・ファイルを右にスクロールして、レコードが長いため表示されていないメンバーを表示しています:

図 102　(続き)サンプル・ルール・ファイル: レベル参照を使用した異なるレベルの複数のロールアップ

DUPLEVEL2,Product	DUPLEVALIAS2,Product
Steel	Berthas
Aluminum	Minis

図 102 のデータを使用して次元構築を実行した場合、Essbase により、図 103 に示すアウトラインが構築されます:

図 103　サンプル・アウトライン: 異なるレベルの複数のロールアップ



この例では、パッケージ・タイプ(缶)のみでなく、パッケージ材料による分析も可能です。たとえば、アルミニウム缶とスチール缶の売上を比較する分析が可能です。

製品次元は疎次元なので、代替アウトライン設計を使用して同じ情報を取得できます。たとえば、Can の下位に Steel と Aluminum をレベル 0 メンバーとして持つパッケージ・タイプに対してマルチレベルの属性次元を作成するとします。アウトラインの設計のガイドラインについては、83 ページの「データベース設計の分析」を参照してください。

複数のデータ・ソースからの共有ロールアップの作成

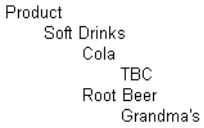
1 つの次元のデータが複数のデータ・ソースに含まれていることがよくあります。複数のデータ・ソースから次元を構築し、複数のロールアップを作成する場合は、最も適切な構築方法を使用して最初のデータ・ソースをロードした後、親子参照構築方法を使用してその他のすべてのデータ・ソースをロードします。Essbase の「次元構築の設定」ダイアログ・ボックスの「次元構築の設定」タブで「共有しない」オプションを選択解除して、共有を許可するように設定します。

たとえば、次の製品データ・ソースを使用します:

```
   "Soft Drinks"  Cola
"Soft Drinks"  "Root Beer"
Cola       TBC
"Root Beer"   Grandma's
```

Essbase は、図 104 に示されているアウトラインを構築します:

図 104　サンプル・アウトライン: Soft Drinks

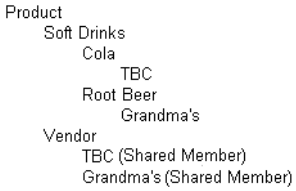


続いて、次の 2 番目のデータ・ソースをロードし、親子参照構築方法を使用して製品とベンダーを関連付けます。Essbase が共有を許可するように設定されていることを確認します。

```
   Vendor  TBC
Vendor  Grandma's
```

Essbase は、図 105 に示されているアウトラインを構築します:

図 105　サンプル・アウトライン: Vendor (共有ロールアップ)



重複するメンバー・アウトラインの構築

重複するメンバー・アウトラインには、同じ名前を持つメンバーが複数含まれていますが、その値は共有されていません。一意のメンバー・アウトラインの場合、共有メンバーのみが同じ名前を持つことができます。第 8 章「重複メンバーのアウトラインの作成および使用」を参照してください。

ルール・ファイルを使用して、重複するメンバー・アウトライン内の次元、レベルおよび世代が一意であるか、重複するメンバーを含めることができるかを設定します。

➤ 次元構築中に次元の一意性を設定する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元プロパティの設定」を参照してください。

ルール・ファイルでは、世代またはレベルの一意性が「次元プロパティ」ダイアログ・ボックスの「世代名」または「レベル名」タブで設定されます。

➤ 次元構築中に世代またはレベルの一意性を設定する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「次元プロパティの設定」を参照してください。

ルール・ファイルを使用したメンバーの一意性の識別

重複するメンバー・アウトライン階層が正しく構築されたことを確認するには、データ・ソース内の修飾メンバー名を使用するか、ルール・ファイルを使用してデータ・ソース内のフィールドから修飾メンバー名を構築します。

多くの場合、Essbase では、参照方法とフィールドの配置の情報を使用して、データ・ソース列を重複するメンバー・アウトライン内のメンバーにマッピングできます。重複するメンバーの次元構築ルール・ファイルは、一意のメンバー・アウトラインのルール・ファイルと類似しています。

次の操作を行うには、より複雑な修飾メンバー名のルール・ファイルが必要です:

- **親子次元構築:** 親子構築方法では、親と子の 2 つのフィールドが必要になります。メンバーが重複している場合、親フィールドに修飾メンバー名を含める必要があります。重複するメンバー・アウトラインでの親子次元構築では、属性関連付けはサポートされません。属性を正しいメンバーに関連付けるには、世代参照またはレベル参照ルール・ファイルを使用して関連付けを別途実行します。
- **アウトライン内の既存のメンバーへの属性の関連付け:** メンバーが重複している場合、属性を関連付けるフィールドに修飾メンバー名を含める必要があります。

ルール・ファイルを使用した修飾メンバー名の構築

データ・ソースにフィールドとして修飾メンバー名が含まれていない場合は、ルール・ファイルを使用して、複数のフィールドを編集、結合し修飾メンバー名にできます。

修飾メンバー名を作成するには、「データ・ソースのプロパティ」ダイアログ・ボックスの「フィールドの編集」タブを使用して、フィールドをコピー、移動および結合し、ブラケットやピリオドを作成します。たとえば、重複するメンバー次元内の既存の都市メンバーに人口属性を割り当てることができます。移動、結合および作成操作を使用して、修飾名を構築できます。

たとえば、次のデータ・ソース内の都市は、すでにアウトライン内に存在しています。これらの都市とこの 4 列のデータ・ソースの最後の列に入っている人口属性を関連付けるとします:

```
Central  "Kansas City"  Kansas    706010
Central  "Kansas City"  Missouri  1070052
East     "New York"     "New York"  8104079
```

修飾名の作成、フィールド列の配置を正確に行うために、ルール・ファイルを使用して次のようにこのソースを編集します:

- 「テキストを使用した作成」操作を使用して、次のテキスト要素にフィールドを 1 つずつ作成します:
  - [
  -].[
  -].[
  -]
- 「移動」操作を使用して、フィールドを次の順序で移動します:

```
     1  2     3   4     5  6      7  8
[  Central  ].[  Kansas  ].[ Kansas City ]  706010
```

- 「結合」操作を使用して、フィールド 1、2、3、4、5、6、7 を結合して単一のフィールドを作成し、このフィールドに[Central].[Kansas].[Kansas City]のように属性を関連付けます。ルール・ファイルに 2 つのフィールドが表示されます:

```
     1              2
[Central].[Kansas].[Kansas City]  706010
```

第 IV 部
データの計算

データの計算の内容：

22 Essbaseデータベースの計算

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 1075 ページの「集約ストレージ・データベースの計算」

データベースの計算について

データベースには、次の 2 タイプの値が含まれます:

- ユーザーが入力する値(入力データ)
- 入力データから計算される値

例:

- 様々な製品の地区ごとの売上高を入力します。さらに、それぞれの合計売上高を計算します。
- いくつかの地区のいくつかの製品について、売上原価の予算値および実績値を入力します。地区ごとに、各製品の予算値と実績値の差異を計算します。
- データベースには、全製品の地区ごとの売上高および価格が含まれています。ある地区である製品の価格を 5%上昇させた場合、全体の収益がどうなるかを計算します。

異なるプラットフォームで計算を実行すると、オペレーティング・システムの数理ライブラリの相違により、セルの値の精度に微妙な差異が発生する場合があります。

注： ほとんどのコンピュータでは、数値は 2 進数(バイナリ)で表現されます。このため、実数の近似表現が得られるのみです。バイナリ・コンピュータでは、小数点以下の桁数が有限なので、1/3 (0.3333...)のような分数を、端点を持つ小数として表現できません。正確に表現できるのは、分母が 2 のべき乗(0.50 など)や 10 のべき乗(0.10 など)の実数のみです。浮動小数点数表現については、IEEE 標準 754(IEEE, 1985)を参照してください。

Essbase では、次の 2 通りの方法でデータベースを計算できます:

- アウトライン計算
- 計算スクリプトの計算

どちらを選択するかは、実行する計算のタイプによって異なります。

アウトライン計算

アウトライン計算は、最も単純化された計算方法です。Essbase では、データベース・アウトラインにおけるメンバー間の関係、およびアウトラインにおけるメンバーに関連付けられている任意の式に基づいて、データベースの計算が行われます。

たとえば、図 106 は、Sample.Basic データベースの市場次元のメンバー間の関係を示しています。New York、Massachusetts、Florida、Connecticut および New Hampshire の値の合計が East の値になります。East、West、South および Central の合計が市場次元の合計値になります。

図 106　市場次元のメンバー間の関係

```
Market
    East (+) (UDAs: Major Market)
        New York (+) (UDAs: Major Market)
        Massachusetts (+) (UDAs: Major Market)
        Florida (+) (UDAs: Major Market)
        Connecticut (+) (UDAs: Small Market)
        New Hampshire (+) (UDAs: Small Market)
    West (+)
    South (+) (UDAs: Small Market)
    Central (+) (UDAs: Major Market)
```

図 107 は、Sample.Basic データベースのシナリオ次元を示しています。Variance メンバーと Variance%メンバーは、それぞれに付加された式から計算されます。

図 107　Variance と Variance%の計算

```
Scenario (Label Only)
    Actual (+)
    Budget (~)
    Variance (~) (Dynamic Calc) (Two Pass Calc) @VAR(Actual, Budget);
    Variance % (~) (Dynamic Calc) (Two Pass Calc) @VARPER(Actual, Budget);
```

メンバーの組合せは、通常のデータベース計算時に計算するのではなく、データの取得時に計算した方が効率がよいことがあります。動的計算を使用して、取得時にデータを計算できます。第 27 章「データ値の動的計算」を参照してください。

計算スクリプトの計算

計算スクリプトの計算は、2 番目の計算方法です。計算スクリプトを使用すると、データベースの計算方法を厳密に選択できます。たとえば、データベースの一部を計算したり、メンバー間でデータ値をコピーしたりできます。

計算スクリプトには、一連の計算コマンド、等式および式が含まれています。たとえば、次の計算スクリプトは、ニューヨーク地区のマーケティング・コストの実績値を 5%増やします。

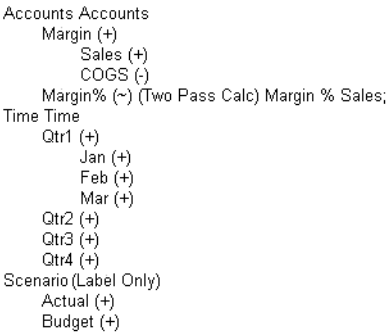
```
   FIX (Actual, “New York”)
  Marketing = Marketing *1.05;
ENDFIX;
```

第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。

多次元計算の概念について

図 108 は、単純なデータベースを使用して、多次元計算の性質を説明しています:

図 108　多次元データベースの計算



このデータベースには、勘定科目、時間およびシナリオの 3 つの次元があります。

会計次元は、次の 4 つのメンバーで構成されています:

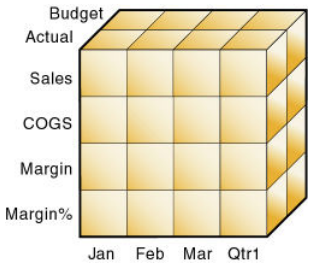
- 入力値である Sales と COGS
- Margin = Sales - COGS
- Margin% = Margin % Sales (Sales のパーセンテージとしての Margin)

時間次元には 4 つの四半期が含まれます。この例では、第 1 四半期(Qtr1)のメンバーである Jan、Feb および Mar のみが表示されています。

シナリオ次元には 2 つの子メンバーがあります。つまり、予定値を示す Budget および実績値を示す Actual です。

図 108 のアウトラインは、図 109 で 3 次元キューブとして図示されています:

図 109　3 次元データベースの図



メンバーの交差部(各次元の 1 つのメンバー)はデータ値を表しており、データ値はデータベース内の 1 つのセルに保管されます。多次元データベース内の特定のデータ値を参照するには、各次元でそれぞれのメンバーを指定する必要があります。Essbase では、メンバーの組合せは次元間演算子(->)によって示されます。ハイフン(-)と大なり記号(>)を使用して、次元間演算子を作成します。次元間演算子とメンバーの間にはスペースを入力しないでください。

Sales、Jan、Actual のデータ値が含まれている単一のセルは、図 110 に示すように、Sales -> Jan -> Actual として記述されます。

図 110　データベースの Sales、Jan、Actual スライス

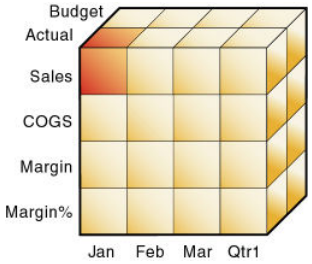


図 111 に示すように、Sales を参照するときは、次の 8 つの値を含むデータベースのスライスを参照することになります:

- Sales -> Jan -> Actual
- Sales -> Feb -> Actual
- Sales -> Mar -> Actual
- Sales -> Qtr1 -> Actual
- Sales -> Jan -> Budget
- Sales -> Feb -> Budget
- Sales -> Mar -> Budget
- Sales -> Qtr1 -> Budget

図 111　データベースの Sales、Actual、Budget スライス

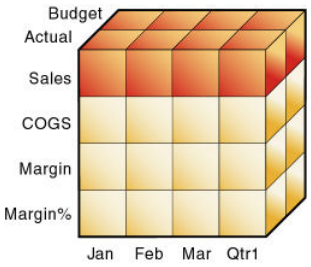
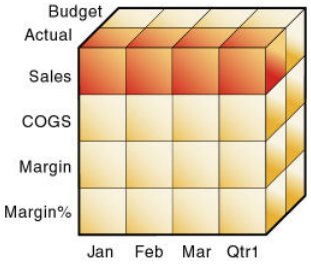


図 112 に示すように、Actual Sales を参照するときは、次の 4 つの値を参照することになります:

- Sales -> Jan -> Actual
- Sales -> Feb -> Actual
- Sales -> Mar -> Actual
- Sales -> Qtr1 -> Actual

図 112　データベースの Actual、Sales スライス



Essbase で式「Margin% = Margin % Sales」を計算するときは、各 Margin の値が取得されて、対応する Sales 値のパーセントとして計算されます。

Essbase では、データベース内が循環され、Margin%は次のように計算されます:

1. Margin -> Jan -> Actual は Sales -> Jan -> Actual のパーセンテージとして計算されます。

   結果は Margin% -> Jan -> Actual に配置されます。

2. Margin -> Feb -> Actual は Sales -> Feb -> Actual のパーセンテージとして計算されます。

   結果は Margin% -> Feb -> Actual に配置されます。

3. Margin -> Mar -> Actual は Sales -> Mar -> Actual のパーセンテージとして計算されます。

   結果は Margin% -> Mar -> Actual に配置されます。

4. Margin -> Qtr1 -> Actual は Sales -> Qtr1 -> Actual のパーセンテージとして計算されます。

   結果は Margin% -> Qtr1 -> Actual に配置されます。

5. Margin -> Jan -> Budget は Sales -> Jan -> Budget のパーセンテージとして計算されます。

結果は Margin% -> Jan -> Budget に配置されます。

6. Essbase では、データベース内のメンバーのあらゆる組合せに対する Margin% の計算が終わるまで、データベース内を循環します。

第 25 章「計算順序の定義」を参照してください。

デフォルト計算の設定

デフォルトでは、データベースの計算はデータベース・アウトラインの CALC ALL になります。CALC ALL は、すべての次元およびメンバーを集計し、アウトライン内のすべての式を計算します。

ただし、任意の計算スクリプトをデフォルトのデータベース計算として指定することもできます。たとえば、計算を実行するたびにスクリプトをロードするのではなく、よく使用するスクリプトをデータベースに割り当てることができます。データベース・レベルで定義した計算設定で計算スクリプトを使用する場合は、この計算スクリプトをデフォルトの計算として設定する必要があります。

➤ デフォルトの計算を設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	デフォルト計算の設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDEFAULTCALCFILE	『Oracle Essbase テクニカル・リファレンス』

データベースの計算

計算権限を持っていれば、データベースを計算できます。

Administration Services を使用してデータベースを計算するときには、計算をバックグラウンドで実行できるので、計算の処理中でも作業を続けることができます。バックグラウンド・プロセスのステータスを確認して、いつ計算が完了したかを確認できます。

➤ データベースを計算するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ブロック・ストレージ・データベースの計算	Oracle Essbase Administration Services Online Help
MaxL	**execute calculation**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	CALC、CALCDEFAULT、CALCLINE	『Oracle Essbase テクニカル・リファレンス』
Smart View	データベース全体の計算	Oracle Hyperion Smart View for Office User's Guide

計算の取消し

➤ Essbase の計算を途中で停止するには、計算の実行中に「取消し」ボタンをクリックします。

計算を取り消すと、Essbase により、次のいずれかの操作が実行されます:

- すべての値を以前の状態に戻す
- 取消し前に計算されたあらゆる値の保存

Essbase での取消しの処理方法は、Essbase のカーネルの分離レベル設定によって異なります。861 ページの「分離レベルの理解」を参照してください。

並列計算とシリアル計算

Essbase では、並列計算とシリアル計算がサポートされます:

- シリアル計算(デフォルト): 計算のすべての手順を単一のスレッドで実行します。各タスクが完了してから次のタスクが開始されます。
- 並列計算: Essbase の計算機では計算を分析できます。さらに、場合によっては複数の CPU(最大 4 つ)にタスクを割り当てることもできます。

961 ページの「並列計算の使用」を参照してください。

セキュリティに関する考慮事項

データベースを計算するには、データベース・アウトラインの計算権限が必要です。計算権限があれば、データベース内のあらゆる値を計算できます。セキュリティ・フィルタによって読取りおよび更新権限が禁止されている場合でも値を計算できます。ユーザーに計算権限を付与するかどうかは慎重に決定してください。

第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」を参照してください。

23 ブロック・ストレージ・データベース用の式の作成

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 24 章「ブロック・ストレージ・データベース用の式の例の確認」
- 1028 ページの「集約ストレージ・アウトラインでの式の作成」
- 第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」
- 第 58 章「計算の最適化」

この章のすべての例は、Sample.Basic データベースに基づいています。

この章で参照される関数の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

式および式計算の使用

式により、データベース・アウトライン内のメンバー間の関係が計算されます。式を使用して、次のことができます:

- データベース・アウトライン内のメンバーに式を適用。データベース計算の精度やパフォーマンスを慎重に制御する必要がない場合は、この方法を使用します。この方法では、式のサイズは 64KB 未満に制限されます。

  380 ページの「式での関数の使用」を参照してください。

- 計算スクリプトに式を挿入。データベース計算を慎重に制御する必要がある場合、この方法を使用します。

  483 ページの「計算スクリプトでの式の使用」を参照してください。

図 113 は、Sample.Basic データベースのメジャー次元を示しています。Margin %、Profit %および Profit per Ounce メンバーは、それぞれに適用された式を使用して計算されます。

図 113　Margin %、Profit %および Profit per Ounce の計算

```
Ratios (~) (Label Only)
    Margin % (+) (Dynamic Calc) (Two Pass Calc) Margin % Sales;
    Profit % (~) (Dynamic Calc) (Two Pass Calc) Profit % Sales;
    Profit per Ounce (~) Profit/@ATTRIBUTEVAL(Ounces);
```

データベース・アウトライン内のメンバーに適用された式は、ユーザーが次のアクションを実行するときに Essbase で計算されます:

- データベースのデフォルトの(CALC ALL)計算を実行するとき。
- 式(メンバーが登録されている次元の CALC DIM など)を含んでいるメンバー、またはメンバー自体を計算する計算スクリプトを実行するとき。第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。

計算スクリプト内の式は、Essbase によって計算スクリプト内で検出された時点で計算されます。

動的計算済メンバーに式が関連付けられている場合、その式はユーザーがデータ値を要求した時点で Essbase によって計算されます。計算スクリプト内では、動的計算済メンバーを計算したり、動的計算済メンバーを式の計算のターゲットに指定したりすることはできません。第 27 章「データ値の動的計算」を参照してください。

データベース・アウトラインや計算スクリプトに含まれる式内で動的計算済メンバーを使用すると、計算パフォーマンスが大幅に低下します。これは、Essbase で動的計算を行うために通常の計算を中断するためです。

データベース・アウトラインに適用する式の中では、代替変数は使用できません。398 ページの「式での代替変数の使用」を参照してください。

式の作成プロセス

アウトライン・エディタの「メンバーのプロパティ」ダイアログ・ボックス内のタブになっている式エディタを使用して、式を作成します。式テキスト領域に式を直接入力するか、式エディタの UI 機能を使用して式を作成します。

式はプレーン・テキストです。必要に応じて、好みのテキスト・エディタで式を作成し、それを式エディタに貼り付けてもかまいません。

式を作成するには:

1. アウトライン・エディタで、式を適用するメンバーを選択します。
2. 式エディタを開きます。

   Oracle Essbase Administration Services Online Help の「アウトライン内の式の作成および編集」を参照してください。
3. 式テキストを入力します。

380 ページの「式での関数の使用」と、Oracle Essbase Administration Services Online Help の「アウトライン内の式の作成および編集」を参照してください。

4. 式の構文を検査します。

   379 ページの「式構文の検査」を参照してください。

5. 式を保存します。

   Oracle Essbase Administration Services Online Help の「アウトライン内の式の作成および編集」を参照してください。

6. アウトラインを保存します。

   Oracle Essbase Administration Services Online Help の「アウトラインの保存」を参照してください。

式の構文の理解

メンバー式を作成するときは、次のルールに従います:

- 式の各ステートメントは、セミコロン(;)で終わります。例:

  ```
  Margin % Sales;
  ```

- 保存済のアウトライン・メンバー名のみを使用します。メンバー名として代替変数を使用する場合、代替変数値が保存済のアウトライン・メンバー名になっている必要があります。
- メンバー名が次のいずれかの条件に該当する場合は、そのメンバー名を二重引用符(" ")で囲みます:
  - スペースが含まれている場合。例:

    ```
    "Opening Inventory" = "Ending Inventory" - Sales + Additions;
    ```

  - 演算子、関数名またはキーワードと同じ場合。

    1206 ページの「計算スクリプト、レポート・スクリプト、式、フィルタ、代替変数値および環境変数値での命名規則」を参照してください。
  - 英数字以外の文字が含まれます。たとえば、ハイフン(-)、アスタリスク(*)、スラッシュ(/)です。
  - すべてが数字であるか、1 つ以上の数字で始まります。たとえば、"100"または"10Prod"です

  二重引用符で囲む必要があるメンバー名の完全なリストは、1206 ページの「計算スクリプト、レポート・スクリプト、式、フィルタ、代替変数値および環境変数値での命名規則」を参照してください。
- 式中の各 IF ステートメントは、ENDIF ステートメントで終わります。

  たとえば、次の式には単純な IF...ENDIF ステートメントが含まれます。この式をデータベース・アウトライン内の Commission メンバーに適用できます:

```
     IF(Sales < 100)
  Commission = 0;
ENDIF;
```

別の IF ステートメント内にネストされた IF ステートメントを使用している場合には、各 IF ステートメントを ENDIF ステートメントで終了します。例:

```
     "Opening Inventory"
(IF (@ISMBR(Budget))
  IF (@ISMBR(Jan))
  "Opening Inventory" = Jan;
  ELSE
  "Opening Inventory" = @PRIOR("Ending Inventory");
  ENDIF;
ENDIF;)
```

- ELSE ステートメントまたは ELSEIF ステートメントは、ENDIF で終わらせる必要はありません。例:

```
     IF (@ISMBR(@DESCENDANTS(West)) OR @ISMBR(@DESCENDANTS(East)
  Marketing = Marketing * 1.5;
ELSEIF(@ISMBR(@DESCENDANTS(South)))
  Marketing = Marketing * .9;
ELSE Marketing = Marketing * 1.1;
ENDIF;
```

  注： 式中に ELSEIF(1 語)ではなく ELSE IF(スペースあり)を使用した場合は、IF ステートメントに対応する ENDIF ステートメントを記述する必要があります。

- ENDIF ステートメントをセミコロン(;)で終わらせる必要はありませんが、そのようにすることをお薦めします。

式を記述するとき、式エディタの構文チェッカを使用して構文を検査できます。379 ページの「式構文の検査」を参照してください。

参照:

- 377 ページの「演算子」
- 377 ページの「次元名とメンバー名」
- 377 ページの「定数値」
- 378 ページの「非定数値」
- 378 ページの「基本的な等式」
- 379 ページの「式構文の検査」

演算子

表 47 に、式で使用できる演算子のタイプをリストします:

表 47　演算子タイプのリスト

演算子のタイプ	説明
数学	共通の算術演算を実行します。 たとえば、値の加算、減算、乗算または除算を実行できます。 385 ページの「算術演算」を参照してください。
条件	条件テストの結果に基づいて式の実行フローを制御します。 たとえば、IF ステートメントを使用して指定の条件をテストできます。 382 ページの「条件テスト」を参照してください。
次元間	特定のメンバー組合せのデータ値をポイントします。 たとえば、指定の地区の指定の製品の売上の値をポイントできます。 392 ページの「次元間にわたるメンバーの組合せの操作」を参照してください。

#MISSING、ゼロおよびその他の値での演算子の使用方法は、『Oracle Essbase テクニカル・リファレンス』の Essbase の関数に関する項を参照してください。

次元名とメンバー名

式には、次元名とメンバー名を含めることができます。例:

- `Scenario`
- `100-10`
- `Feb`

定数値

メンバーに定数値を割り当てることができます。例:

```
California = 120;
```

この式では、「California」は疎次元のメンバーで、120 は定数値です。「California」のすべての存在可能なデータ・ブロックが Essbase で自動的に作成され、すべてのデータ・セルに値 120 が割り当てられます。何千ものデータ・ブロックが作成される場合もあります。

疎次元で値が必要な交差のみに定数を割り当てるには、FIX ステートメントを使用します。973 ページの「疎次元のメンバーに割り当てられた定数値」を参照してください。

非定数値

疎次元のメンバーに定数以外の値を割り当てるときに、そのメンバーのデータ・ブロックが存在していない場合、Essbase で「等式によるブロックの作成」を使用可能にするまでは、新しいブロックが作成されません。デフォルトでは、「等式によるブロックの作成」は無効になっています。

たとえば、計算を実行する前には存在していなかった West のブロックを作成するには、次の式での「等式によるブロックの作成オプション」を有効にする必要があります:

```
West = California + 120;
```

注： データベースに対して「等式によるブロックの作成」が無効で、等式の左側または右側にデータ・ブロックが存在する場合、式によって結果が生成されます。

データベース・レベルで「等式によるブロックの作成」を使用可能にして、ブロックが常に生成されるようにできます。または、SET CREATEBLOCKONEQ ON | OFF 計算コマンドを使用することで計算スクリプト内でブロック生成を制御できます。

➤ 特定のデータベースに対するすべての計算スクリプトに対して、「等式によるブロックの作成」機能を使用可能にするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**等式によるブロックの作成の有効化**	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATE	『Oracle Essbase テクニカル・リファレンス』

アプリケーションまたはデータベース・レベルで「等式によるブロックの作成」を使用可能にすると、不要なブロックが作成され、計算パフォーマンスに影響が出る場合があります。計算スクリプトを使用してブロックの作成を制御するには、SET CREATEBLOCKONEQ ON | OFF 計算コマンドを使用します。974 ページの「疎次元のメンバーに割り当てられた非定数値」を参照してください。

基本的な等式

式で算術演算を使用して、基本的な等式を作成できます。等式は、データベース・アウトラインまたは計算スクリプト内に含めることができます。

等式の構文は次のとおりです:

```
member
 =
mathematical_operation
;
```

member は、データベース・アウトラインのメンバー名です。
mathematical_operation は、有効な算術演算です。

次の例では、Essbase はデータベースのデータを循環しながら、Sales 内の値から COGS 内の値を減算し、結果を Margin に配置します:

```
Margin = Sales - COGS;
```

次の例は、データベース・アウトラインおよび計算スクリプトにおける等式の使用方法を示しています。アウトラインでは、次の式を Markup メンバーに適用します:

```
(Retail - Cost) % Retail;
```

次に、計算スクリプトで次の式を使用します:

```
Markup = (Retail - Cost) % Retail;
```

Essbase はデータベースのデータを循環しながら、Retail 内の値から Cost 内の値を減算し、結果の値を Retail の値のパーセンテージとして計算し、その結果を Markup に配置します。

式構文の検査

Essbase には、式の構文エラーをユーザーに通知する、Essbase サーバーベースの式構文検査機能が組み込まれています。たとえば、ユーザーが間違った関数名を入力すると、Essbase によりそのことが通知されます。カスタム定義のマクロや関数名のリストに対して、不明な名前を検証できます。ユーザーがアウトラインに関連付けられたサーバーやアプリケーションに接続していない場合は、不明な名前を検証するため、Essbase により自動的に接続が確立されます。

式のセマンティク・エラーは、構文検査機能では通知できません。セマンティク・エラーは、式が期待どおりに動作しないときに発生します。セマンティク・エラーを見つけるには、計算を実行して、予想どおりの結果が得られるかどうかを確認します。

Essbase により、式エディタの下部に構文検査の結果が表示されます。Essbase で構文エラーが検出されなかった場合は、「エラーはありません」というメッセージが表示されます。

Essbase で 1 つ以上の構文エラーが検出された場合は、エラーのある行の行番号とエラーの短い説明が表示されます。たとえば、式の行末文字としてセミコロンを入力しなかった場合、Essbase により次のようなメッセージが表示されます:

```
Error: line 1: invalid statement; expected semicolon
```

式エディタまたはアウトライン・エディタでの検証で式が承認されても、アウトラインが保存されるときに Essbase サーバーでセマンティク・エラーが検出された場合は、次のような措置がとられます:

- 誤った式は、エラーがあっても、アウトラインの一部として保存されます。
- Essbase サーバーにより、エラーの内容を示すメッセージがアプリケーション・ログに書き込まれ、誤った式が表示されます。
- Essbase サーバーにより、誤った式に関連付けられたメンバーのコメント・フィールドにエラー・メッセージが書き込まれます。このメッセージでは、誤った式がロードされなかったことが示されます。このコメントを表示するには、アウトライン・エディタで、アウトラインを閉じてから再度開きます。
- メンバー式を訂正せずに、そのメンバーを含んでいる計算が実行された場合は、計算中にその式が無視されます。

式を修正し、アウトラインを保存すると、メンバー・コメント内のメッセージが削除されます。アウトラインを再度開くと、更新されたコメントが表示されます。

➤ 式構文を検査する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アウトライン内の式の作成および編集」を参照してください。

式での関数の使用

関数は、特別な計算を実行し、メンバーまたはデータ値のセットを戻す事前定義済のルーチンです。表 48 に、式で使用できる関数のタイプをリストします:

表 48　関数タイプのリスト

関数のタイプ	説明
ブール	条件テストを実行できるように、TRUE(1)または FALSE(0)の値を戻します。 たとえば、@ISMBR 関数では、現在のメンバーが指定されたメンバーと一致するかどうかを判断できます。 382 ページの「条件テスト」を参照してください。
数学	特化された数学計算を実行します。 たとえば、@AVG 関数を使用して、メンバーのリストの平均値を戻せます。 385 ページの「算術演算」を参照してください。
関係	計算中に、データベース内のデータ値を検索します。 たとえば、@ANCESTVAL 関数を使用して、指定したメンバーの組合せの祖先の値を戻せます。 386 ページの「メンバー関係関数」を参照してください。
範囲	別の関数またはコマンドに対する引数として、メンバーの範囲を宣言します。 たとえば、@SUMRANGE 関数を使用して、特定の範囲内のすべてのメンバーの合計を戻せます。 387 ページの「範囲関数」を参照してください。

関数のタイプ	説明
財務	特化された財務計算を実行します。 たとえば、@INTEREST 関数を使用して、単利を計算できます。また、@PTD 関数を使用して、期間累計値を計算できます。 387 ページの「財務関数」を参照してください。
メンバーのリストおよび範囲の指定	複数のメンバーまたはある範囲のメンバーを指定します。 たとえば、@ISMBR 関数では、現在計算中のメンバーが指定のリスト内のメンバーまたは範囲内のメンバーと一致しているかどうかをテストできます。 388 ページの「メンバーのリストおよび範囲の指定」を参照してください。
メンバー・リストの生成	指定したメンバーに基づいてメンバーのリストを生成します。 たとえば、@ICHILDREN 関数を使用して、指定したメンバーとその子を戻せます。 389 ページの「メンバー・リストの生成」を参照してください。
文字列の操作	メンバー名および次元名の文字列を操作します。 たとえば、名前に接頭辞を追加したり、名前から接尾辞を削除したり、文字列として名前を渡してメンバー名を生成できます。 392 ページの「メンバー名の操作」を参照してください。
次元間にわたるメンバーの組合せ	次元間演算子(->)を使用して、特定のメンバー組合せのデータ値をポイントします。 392 ページの「次元間にわたるメンバーの組合せの操作」を参照してください。
相互依存値	同じ次元のメンバーの値を必要とするが、必要な値がまだ計算されていない式の場合。 393 ページの「相互依存値の使用」を参照してください。
差異および差異のパーセンテージ	予定値と実績値との差異または差異のパーセンテージを計算します。 394 ページの「実績値と予定値との差異または差異のパーセンテージの計算」を参照してください。
割当て	親レベルで入力される値を、子メンバー全体に割り当てます。同じ次元内の値または複数次元の値を割り当てることができます。 たとえば、@ALLOCATE 関数を使用して、親レベルで入力される売上の値を親の子に割り当てることができます。ただし、それぞれの子の割当ては、前年の売上における占有率によって決まります。 395 ページの「値の割当て」を参照してください。
予測	データの補整や補間、あるいは今後の値の計算を目的として、データを操作します。 たとえば、@TREND 関数を使用して、過去の値に対する曲線近似に基づいて今後の値を計算できます。 396 ページの「予測関数」を参照してください。
統計	高度な統計計算を行います。たとえば、@RANK 関数を使用して、データ・セット内の指定したメンバーまたは指定した値のランクを計算できます。 396 ページの「統計関数」を参照してください。

関数のタイプ	説明
日付と時刻	計算式において日付や時刻の属性を使用します。 たとえば、@TODATE 関数を使用して、日付文字列を計算式で使用できる数値に変換できます。 397 ページの「日付と時刻関数」を参照してください。
計算モード	Essbase で式の計算に使用される計算モード(セル、ブロック、ボトムアップおよびトップダウン)を指定します。 397 ページの「計算モード関数」を参照してください。
カスタム定義	このタイプでは、計算操作用にユーザーが開発した関数を実行できます。このようなカスタム開発関数は Java プログラミング言語で作成され、Essbase 計算機フレームワークにより外部関数として呼び出されます。 397 ページの「カスタム定義関数」を参照してください。

注： 関数の省略はサポートされていません。一部のコマンドは省略形式でも機能しますが、類似した名前の関数が複数存在する場合、Essbase で誤った関数が使用される可能性があります。正しい結果が確実に得られるように、完全な関数名を使用してください。

条件テスト

単一の条件テストまたは一連の条件テストを行う式を定義すれば、計算のフローを制御できます。

IF および ENDIF コマンドでは、条件ブロックが定義されます。IF コマンドと ENDIF コマンドの間に置かれた式は、テストが TRUE(1)を戻す場合にのみ実行されます。テストが FALSE(0)を戻す場合は、ELSE および ELSEIF コマンドを使用して代替アクションを指定できます。各 ELSE コマンドに続く式は、前のテストが FALSE(0)を戻す場合にのみ実行されます。各 ELSEIF コマンドに続く条件は、前の IF コマンドが FALSE(0)を戻す場合にのみテストされます。375 ページの「式の構文の理解」を参照してください。

計算スクリプトで条件式を使用するときは、この項の例で示すように、その式を丸カッコで囲み、データベース・アウトラインのメンバーと関連付けます。

IF コマンドとともに、条件テストの結果に基づいて、TRUE または FALSE(それぞれ 1 または 0)を戻す関数を使用できます。これらの関数は、ブール関数と呼ばれています。

ブール関数を使用して、使用する式を決定します。この決定は、現在のメンバーの組合せの特性に基づいて行われます。たとえば特定の計算を、入力データを含む製品次元内のメンバーのみに制限するには、計算の前に、@ISLEV(Product,0)に基づいた IF テストを配置します。

関数パラメータのいずれかが、@ISMBR(Sales -> Budget)のような次元間メンバーである場合は、TRUE(1)の値を戻すように、次元間メンバーのすべての部分を現在のセルのプロパティと合わせる必要があります。

表 49 に、条件を指定するブール型関数をリストします:

表 49　条件をテストするブール型関数のリスト

関数	条件
@ISACCTYPE	現在のメンバーが指定された勘定科目タグを持っている(支出タグなど)
@ISANCEST	現在のメンバーが指定されたメンバーの祖先である
@ISIANCEST	現在のメンバーが指定されたメンバーの祖先であるか、指定されたメンバー自体である
@ISCHILD	現在のメンバーが指定されたメンバーの子である
@ISICHILD	現在のメンバーが指定されたメンバーの子であるか、指定されたメンバー自体である
@ISDESC	現在のメンバーが指定されたメンバーの子孫である
@ISIDESC	現在のメンバーが指定されたメンバーの子孫であるか、指定されたメンバー自体である
@ISGEN	指定された次元の現在のメンバーが指定された世代に存在している
@ISLEV	指定された次元の現在のメンバーが指定されたレベルに存在している
@ISMBR	現在のメンバーが指定されたメンバーのいずれかと一致している
@ISPARENT	現在のメンバーが指定されたメンバーで親である
@ISIPARENT	現在のメンバーが指定されたメンバーの親であるか、指定されたメンバー自体である
@ISSAMEGEN	(指定されたメンバーと同じ次元の)現在のメンバーが、指定されたメンバーと同じ世代に存在している
@ISSAMELEV	(指定されたメンバーと同じ次元の)現在のメンバーが、指定されたメンバーと同じレベルに存在している
@ISSIBLING	現在のメンバーが指定されたメンバーの兄弟である
@ISISIBLING	現在のメンバーが指定されたメンバーの兄弟であるか、指定されたメンバー自体である
@ISUDA	指定された次元の現在のメンバーに対して、指定された UDA が存在している

データベース・アウトラインに式を配置するときは、IF、ELSE、ELSEIF および ENDIF コマンドとブール関数のみを使用して、計算フローを制御できます。計算スクリプト内では、その他の制御コマンドも使用できます。

計算スクリプトの作成方法や、計算スクリプトを使用して Essbase によるデータベース計算を制御する方法については、第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。個々の Essbase 関数および計算コマンドについては、『Oracle Essbase テクニカル・リファレンス』を参照してください。

条件テストの例

次の式を、データベース・アウトラインの Commission メンバーに適用できます。

次の例の式では、売上が 500000 を超える場合、売上の 1%の歩合が計算されます:

```
IF(Sales > 500000)
```

```
 Commission = Sales * .01;
ENDIF;
```

この式を計算スクリプト内に記述する場合は、式と Commission メンバーを次のように関連付ける必要があります:

```
   Commission (IF(Sales > 500000)
 Commission = Sales * .01;
ENDIF;)
```

Essbase では、データベース内を循環して、次の計算が実行されます:

1. IF ステートメントでは、現在のメンバーの組合せに対する Sales メンバーの値が 500000 を超えているかが確認されます。
2. Sales が 500000 を超えている場合は、Essbase によって、Sales の値に 0.01 が乗算され、結果が Commission に置かれます。

次の例の式では、現在のメンバーの祖先をテストした後、適切な Payroll 計算式が適用されています:

```
   IF(@ISIDESC(East) OR @ISIDESC(West))
 Payroll = Sales * .15;
ELSEIF(@ISIDESC(Central))
 Payroll = Sales * .11;
ELSE
 Payroll = Sales * .10;
ENDIF;
```

この式を計算スクリプト内に記述する場合は、式と Payroll メンバーを次のように関連付ける必要があります:

```
   Payroll(IF(@ISIDESC(East) OR @ISIDESC(West))
 Payroll = Sales * .15;
ELSEIF(@ISIDESC(Central))
 Payroll = Sales * .11;
ELSE
 Payroll = Sales * .10;
ENDIF;)
```

Essbase では、データベース内を循環して、次の計算が実行されます:

1. IF ステートメントでは、@ISIDESC 関数を使用して、市場次元の現在のメンバーが East と West のいずれかの子孫であるかどうかを確認します。
2. 市場次元の現在のメンバーが East または West の子孫である場合は、Essbase によって、Sales の値に 0.15 が乗算されてから、次のメンバーの組合せが処理されます。
3. 現在のメンバーが East または West の子孫でない場合は、ELSEIF ステートメントで@ISIDESC 関数を使用して、現在のメンバーが Central の子孫であるかどうかを確認します。

4. 市場次元の現在のメンバーが Central の子孫である場合は、Essbase によって、Sales の値に 0.11 が乗算されてから、次のメンバーの組合せが処理されます。
5. 現在のメンバーが East、West および Central の子孫でない場合は、Essbase によって、Sales の値に 0.10 が乗算されてから、次のメンバーの組合せが処理されます。

367 ページの「多次元計算の概念について」を参照してください。@ISIDESC 関数については、『Oracle Essbase テクニカル・リファレンス』を参照してください。

算術演算

表 50 に、式内で多くの算術演算を実行できる数学関数をリストします:

表 50　数学関数のリスト

関数	演算
@ABS	式の絶対値を戻す
@AVG	指定されたメンバー・リスト内の値の平均値を戻す
@EXP	e (自然対数の底)を指定された式でべき乗した値を戻す
@FACTORIAL	式の階乗を戻す
@INT	メンバーまたは式の 2 番目に小さい整数値を戻す
@LN	指定された式の自然対数を戻す
@LOG	指定された式の指定された底の対数を戻す
@LOG10	指定された式の 10 を底とする対数を戻す
@MAX	指定されたメンバー・リスト内の式における最大値を戻す
@MAXS	指定されたメンバー・リスト内の式における最大値を戻すが、ゼロ(0)および#MISSING 値はスキップする
@MIN	指定されたメンバー・リスト内の式における最小値を戻す
@MINS	指定されたメンバー・リスト内の式における最小値を戻すが、ゼロ(0)および#MISSING 値はスキップする
@MOD	指定された 2 つのメンバーの除算から得られる余りを戻す
@POWER	指定されたメンバーの指定された階乗の値を戻す
@REMAINDER	式の剰余値を戻す
@ROUND	指定された小数点以下の桁数で丸めたメンバーまたは式を戻す
@SUM	指定されたすべてのメンバーの値の合計を戻す
@TRUNCATE	式の切捨て値を戻す

関数	演算
@VAR	指定された 2 つのメンバー間の差異(差分)を戻す。 394 ページの「実績値と予定値との差異または差異のパーセンテージの計算」を参照してください。
@VARPER	指定された 2 つのメンバー間の差異(差分)のパーセンテージを戻す。 394 ページの「実績値と予定値との差異または差異のパーセンテージの計算」を参照してください。

メンバー関係関数

表 51 に、関係関数をリストします。この関数では、特定の値を検索するため、Essbase で現在計算中のメンバーの組合せを使用できます:

表 51　メンバー関係関数のリスト

関数	検索する値
@ANCESTVAL	指定したメンバーの組合せに対する祖先の値
@ATTRIBUTEVAL	現在のメンバーに関連する、指定した数値属性次元または日付属性次元からの属性の数値
@ATTRIBUTESVAL	現在のメンバーに関連する、指定したテキスト属性次元からの属性のテキスト値
@ATTRIBUTEBVAL	現在のメンバーに関連する、指定したブール属性次元からの属性の値(TRUE または FALSE)
@CURGEN	指定した次元に対する現在のメンバーの組合せの世代番号
@CURLEV	指定した次元に対する現在のメンバーの組合せのレベル番号
@GEN	指定したメンバーの世代番号
@LEV	指定したメンバーのレベル番号
@MDANCESTVAL	複数の次元にわたる指定したメンバーの組合せに対する祖先の値
@SANCESTVAL	指定したメンバーの組合せに対する共有された祖先の値
@PARENTVAL	指定したメンバーの組合せに対する親の値
@MDPARENTVAL	複数の次元にわたる指定したメンバーの組合せに対する親の値
@SPARENTVAL	指定したメンバーの組合せに対する共有された親の値
@XREF	現在のデータベースからの値の計算に使用される、別のデータベースからのデータ値
@XWRITE	値を別の Essbase データベースまたは同じデータベースに書き込むために使用されます

個別の Essbase 関数については、『Oracle Essbase テクニカル・リファレンス』を参照してください。

範囲関数

表 52 に、特定の範囲のメンバーに対して式を実行できる範囲関数をリストします:

表 52　範囲関数のリスト

関数	計算
@AVGRANGE	メンバーの範囲におけるメンバーの平均値
@CURRMBRRANGE	Essbase で現在計算されているメンバーの組合せの相対位置を基準にしたメンバーの範囲
@MAXRANGE	メンバー範囲内のメンバーの最大値
@MAXSRANGE	メンバー範囲内のメンバーの最大値(ゼロおよび#MISSING 値はスキップ可能)
@MDSHIFT	メンバー範囲内の次のメンバーまたは n 番目のメンバー(複数の次元における現在のメンバーと同一の他のメンバーがすべて保持される)
@MINRANGE	メンバー範囲内のメンバーの最小値
@MINSRANGE	メンバー範囲内のメンバーの最小値(ゼロおよび#MISSING 値はスキップ可能)
@NEXT	メンバー範囲内の次のメンバーまたは n 番目のメンバー
@NEXT	メンバー範囲内の次のメンバーまたは n 番目のメンバー(#MISSING、ゼロまたはその両方の値はスキップ可能)
@PRIOR	メンバー範囲内の直前または n 番前にあるメンバー
@PRIORS	メンバー範囲内の直前または n 番前にあるメンバー(#MISSING、ゼロまたはその両方の値はスキップ可能)
@SHIFT @SHIFTPLUS または @SHIFTMINUS の場合もある	メンバー範囲内の次または n 番目のメンバー(現在のメンバーおよび指定された次元内で、同一の他のメンバーがすべて保持される)
@SUMRANGE	メンバーの範囲における指定したメンバーすべての値の合計

財務関数

表 53 に、式内に財務計算を含めることができる財務関数をリストします:

表 53　財務関数のリスト

関数	計算
@ACCUM	指定したメンバーまでの値の累積
@COMPOUND	複利計算の収益
@COMPOUNDGROWTH	メンバーの範囲における指定したメンバーの複合成長を表す一連の値
@DECLINE	定率法を使用して計算される特定の期間の減価償却

関数	計算
@DISCOUNT	範囲内での開始期間から割り引く金額に達する期間まで、指定した率で割り引かれる値
@GROWTH	指定された値の線形増分を表す一連の値
@INTEREST	指定したメンバーに対する指定した率での単利
@IRR	時間次元または指定されたメンバー範囲内で計算されたキャッシュ・フローの内部利益率。少なくとも 1 つの投資(負)と 1 つの収入(正)が含まれている必要があります。最初の推測として 0.07 を算入します(最初の推測は構成できません)。
@IRREX	時間次元または指定されたメンバー範囲内で計算されたキャッシュ・フローの内部利益率。少なくとも 1 つの投資(負)と 1 つの収入(正)が含まれている必要があります。最初の推測とアルゴリズムの反復回数を構成するための機能を備えています。
@NPV	一連の収支に基づいた投資の正味現在価値
@PTD	時間としてタグ付けされた次元におけるメンバーの期間累計値
@SLN	当期の資産を減価償却する場合の期間当たりの償却額(期間全体で計算)。 減価償却法は定額法を使用。
@SYD	当期の資産を減価償却する場合の期間当たりの償却額(期間全体で計算)。 使用される減価償却方法は、算術級数法です。

注： 単一のメンバー式に複数の財務関数(たとえば@NPV と@SLN、@NPV の複数のインスタンスなど)を含めることはできません。複数の財務関数を必要とするメンバー式は、複数の式に分割して、各式に財務関数が 1 つのみ含まれるようにします(たとえば、`MemberName`(@NPV(...));`Membername`(@NPV(...)))。

メンバーに関連する関数

この項では、メンバーを参照する式の作成について説明します。

- 388 ページの「メンバーのリストおよび範囲の指定」
- 389 ページの「メンバー・リストの生成」
- 392 ページの「メンバー名の操作」
- 392 ページの「次元間にわたるメンバーの組合せの操作」

メンバーのリストおよび範囲の指定

一部の関数では、複数のメンバーまたはメンバーの範囲を指定する必要があります。たとえば、@ISMBR 関数では、現在計算中のメンバーが指定のリスト内のメンバーまたは範囲内のメンバーと一致しているかどうかをテストできます。

表 54 に、メンバーを指定する構文をリストします:

表 54　メンバーのリストおよび範囲を指定する構文

メンバーのリストまたは範囲	構文
単一のメンバー	メンバー名。 例: `Mar2001`
メンバーのリスト	カンマ区切り(,)のメンバー名のリスト。 例: `Mar2001, Apr2001, May2001`
同じレベルにあるすべてのメンバーの範囲(2 つのメンバー定義の間にあり、かつそれらのメンバー定義自体も含む)	コロン(:)で区切られた 2 つのメンバー定義の名前。例: `Jan2000:Dec2000`
同じ世代にあるすべてのメンバーの範囲(2 つのメンバー定義の間にあり、かつそれらのメンバー定義自体も含む)	2 つのコロン(::)で区切られた 2 つのメンバー定義の名前。 例: `Q1_2000::Q4_2000`
関数で生成したメンバーのリストまたはメンバーの範囲	メンバー・リストのコンテンツおよび対応する関数のリストは389 ページの「メンバー・リストの生成」を参照してください。

範囲とリストの組合せ	カンマ(,)で区切られた各範囲、リストおよび関数。
	例:
	`Q1_97::Q4_98, FY99, FY2000`
	または
	`@SIBLINGS(Dept01), Dept65:Dept73, Total_Dept`

メンバー・リストまたはメンバーの範囲を必要とする関数で、メンバー・リストまたはメンバーの範囲を指定しなかった場合、Essbase では、時間のタグが付けられた次元のレベル 0 メンバーが使用されます。時間のタグが付けられた次元が存在しない場合、Essbase でエラー・メッセージが表示されます。

メンバー・リストの生成

メンバー・セット関数では、指定したメンバーまたはメンバー・リストに基づくメンバー・リストを生成できます。表 55 に、メンバー・セット関数をリストします:

表 55 メンバー・セット関数のリスト

関数	メンバー・リストのコンテンツ
@ALLANCESTORS	指定したメンバーのすべての祖先(共有メンバーとして指定したメンバーの祖先を含む)。この関数には、指定したメンバーは含まれません。
@IALLANCESTORS	指定したメンバーのすべての祖先(共有メンバーとして指定したメンバーの祖先を含む)。この関数には指定したメンバーが含まれます。
@ANCEST	指定した世代またはレベルにある指定したメンバーの祖先
@ANCESTORS	指定したメンバーのすべての祖先(オプションで、指定した世代またはレベルまで)。指定したメンバーは含まれません
@IANCESTORS	指定したメンバーのすべての祖先(オプションで、指定した世代またはレベルまで)。指定したメンバーも含まれます
@LANCESTORS	指定したメンバー・リストのすべての祖先(オプションで、指定した世代またはレベルまで)。指定したメンバーは含まれません
@ILANCESTORS	指定したメンバー・リストのすべての祖先(オプションで、指定した世代またはレベルまで)。指定したメンバーも含まれます
@ATTRIBUTE	指定した属性次元メンバーに関連付けられたすべての基本次元メンバー
@WITHATTR	指定した条件を満たす属性に関連付けられたすべての基本メンバー
@BETWEEN	名前の文字列値が指定した 2 つの文字列トークンの間(2 つのトークンを含む)に収まるすべてのメンバー
@CHILDREN	指定したメンバーのすべての子。指定したメンバーは含まれません
@ICHILDREN	指定したメンバーのすべての子。指定したメンバーも含まれます
@CURRMBR	指定した次元で計算中の現在のメンバー
@DESCENDANTS	指定したメンバーのすべての子孫(オプションで、指定した世代またはレベルまで)。共有メンバーの子孫および指定したメンバーは含まれません
@IDESCENDANTS	指定したメンバーのすべての子孫(オプションで、指定した世代またはレベルまで)。指定したメンバーは含まれますが、共有メンバーの子孫は含まれません
@LDESCENDANTS	指定したメンバー・リストのすべての子孫(オプションで、指定した世代またはレベルまで)。指定したメンバーは含まれません
@ILDESCENDANTS	指定したメンバー・リストのすべての子孫(オプションで、指定した世代またはレベルまで)。指定したメンバーも含まれます
@RDESCENDANTS	指定したメンバーのすべての子孫(オプションで、指定した世代またはレベルまで)。共有メンバーの子孫は含まれますが、指定したメンバーは含まれません
@IRDESCENDANTS	指定したメンバーのすべての子孫(オプションで、指定した世代またはレベルまで)。指定したメンバーおよび共有メンバーの子孫も含まれます
@EQUAL	指定したトークン名と一致するメンバー名
@NOTEQUAL	指定したトークン名と一致しないメンバー名
@EXPAND	メンバー・リスト内のメンバーごとにメンバー・セット関数を呼び出してメンバーの検索を拡張します

関数	メンバー・リストのコンテンツ
@GENMBRS	指定した次元の指定した世代にあるすべてのメンバー
@LEVMBRS	指定した次元の指定したレベルにあるすべてのメンバー
@LIKE	指定したパターンと一致するメンバー名。
@LIST	複数のリスト引数を必要とする関数で処理される、メンバーの個々のリスト
@MATCH	指定したワイルドカード選択に一致するすべてのメンバー
@MBRCOMPARE	比較基準と一致するメンバー名
@MBRPARENT	指定したメンバーの親
@MEMBER	文字列として指定された名前を持つメンバー
@MERGE	別の関数で処理される 2 つのメンバー・リストをマージしたリスト
@PARENT	指定した次元で計算中の現在のメンバーの親
@RANGE	ある次元の指定されたメンバーを別の次元の指定されたメンバー範囲と交差させたメンバー・リスト
@REMOVE	一部のメンバーが削除されたメンバーのリスト
@RELATIVE	指定したメンバーの上または下の指定した世代またはレベルにあるすべてのメンバー
@SHARE	指定したメンバーのすべての共有メンバーを識別するメンバー・リスト
@SIBLINGS	指定したメンバーのすべての兄弟。指定したメンバーは含まれません
@ISIBLINGS	指定したメンバーのすべての兄弟。指定したメンバーも含まれます
@LSIBLINGS	データベース・アウトライン内で指定したメンバーに先行するすべての兄弟。指定したメンバーは含まれません
@RSIBLINGS	データベース・アウトライン内で指定したメンバーに続くすべての兄弟。指定したメンバーは含まれません
@ILSIBLINGS	データベース・アウトライン内で指定したメンバーに先行するすべての兄弟。指定したメンバーも含まれます
@IRSIBLINGS	データベース・アウトライン内で指定したメンバーに続くすべての兄弟。指定したメンバーも含まれます
@SHIFTSIBLING	メンバーから指定した距離にある兄弟
@NEXTSIBLING	メンバーの次または右端の兄弟
@PREVSIBLING	メンバーの前または左端の兄弟
@UDA	Essbase サーバー上に定義された共通の UDA を持つすべてのメンバー
@XRANGE	同じレベルにある 2 つの単一メンバーまたは次元間メンバーで指定されるメンバーの範囲 (2 つのメンバーを含む)を識別するメンバー・リスト

メンバー名の操作

メンバー名を文字列として操作できます。表 56 に、文字列の操作関数をリストします:

表 56　文字列の操作関数のリスト

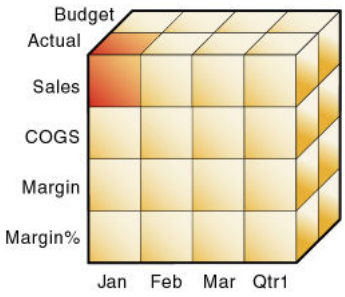
関数	文字列の操作
@CONCATENATE	メンバー名や指定した文字列を他のメンバー名または文字列に追加して、文字列を作成します
@NAME	メンバー名を文字列として戻します
@SUBSTRING	他の文字列またはメンバー名から部分文字列を戻します

次元間にわたるメンバーの組合せの操作

次元間演算子を使用して、特定メンバーの組合せのデータ値を指します。ハイフン(-)と大なり記号(>)を使用して、次元間演算子を作成します。次元間演算子とメンバーの間にはスペースを入力しないでください。

図 114 は多次元キューブを簡単に例示したもので、Jan は X 軸の 1 番目の列、Sales は Y 軸の 4 番目および最上部の行、そして Actual は Z 軸の 1 番目の行に存在します。この例では、Sales -> Jan -> Actual が単一のデータ値の交差部です。

図 114　単一のデータ値の指定



次の例では、各市場の各製品に様々な支出を割り当てることにより、次元間演算子の使用方法を示します。すべての市場のすべての製品について、Misc_Expenses の値がわかっています。この式では、Product -> Market の個々の組合せに、様々な支出(Misc_Expenses)の合計のパーセンテージが割り当てられます。割当ては、各市場の各製品の Sales の値に基づいて行われます。

```
Misc_Expenses = Misc_Expenses -> Market -> Product *

  (Sales / ( Sales -> Market -> Product));
```

Essbase では、データベース内を循環して、次の計算が実行されます:

1. 現在のメンバーの組合せの Sales 値が、すべての市場とすべての製品の Sales の合計値(Sales -> Market -> Product)で除算されます。

2. 手順 1 で計算された値と、すべての市場とすべての製品の Misc_Expenses 値 (Misc_Expenses -> Market -> Product)とが乗算されます。
3. 計算結果が現在のメンバーの組合せの Misc_Expenses に割り当てられます。

次元間演算子を使用すると、パフォーマンスに重大な影響が出る場合があります。最適化のガイドラインについては、974 ページの「次元間演算子の使用」を参照してください。

値に関連する関数

この項では、値に関連する式について説明します。

- 393 ページの「相互依存値の使用」
- 394 ページの「実績値と予定値との差異または差異のパーセンテージの計算」
- 395 ページの「値の割当て」

相互依存値の使用

Essbase では、同じ次元の特定の範囲のメンバーに対する複数の式を同時に計算することによって、計算パフォーマンスが最適化されます。ただし、一部の式では、同じ次元のメンバーから値を取得する必要があるにもかかわらず、これらの値が Essbase でまだ計算されていない場合があります。

具体的な例として、キャッシュ・フローの相互依存値があります。この場合は、期首在庫高が、前月の期末在庫高に依存します。

期首在庫高(Opening Inventory)と期末在庫高(Ending Inventory)の値を月ごとに計算する必要があります。表 57 に示すとおりの結果を得る場合を仮定します:

表 57　キャッシュ・フローのデータ値の例

	Jan	Feb	Mar
Opening Inventory	100	120	110
Sales	50	70	100
Addition	70	60	150
Ending Inventory	120	110	160

1 月の Opening Inventory の値がデータベースにロードされるとすると、表 57 の結果を得るために必要な計算は次のようになります:

```
   1. January Ending  = January Opening - Sales + Additions
2. February Opening = January Ending
3. February Ending  = February Opening - Sales + Additions
4. March Opening    = February Ending
5. March Ending     = March Opening - Sales + Additions
```

必要な計算結果を得るには、複数の相互依存する等式をデータベース・アウトラインの単一のメンバーに適用します。

データベース・アウトラインの Opening Inventory メンバーに適用される次の式で、正しい値が計算されます:

```
   IF(NOT @ISMBR (Jan))
  "Opening Inventory" = @PRIOR("Ending Inventory");
ENDIF;
"Ending Inventory" = "Opening Inventory" - Sales + Additions;
```

計算スクリプト内に式を挿入した場合は、その式を次のように Opening Inventory メンバーと関連付ける必要があります:

```
   "Opening Inventory"
(IF(NOT @ISMBR (Jan))
  "Opening Inventory" = @PRIOR("Ending Inventory");
ENDIF;)
"Ending Inventory" = "Opening Inventory" - Sales + Additions;
```

Essbase では、月の値を循環して、次の計算が実行されます:

1. IF ステートメントと@ISMBR 関数で、年次元の現在のメンバーが Jan でないことを確認します。この手順が必要なのは、1 月の期首在庫高の値が入力値になっているためです。
2. 当月が 1 月でない場合、@PRIOR 関数により、前月の期末在庫高の値が取得されます。この値は、当月の期首在庫高に割り当てられます。
3. 当月の期末在庫高が計算されます。

注： 正しい計算結果を得るには、これらの式を単一のメンバー(Opening Inventory)に対して記述する必要があります。Opening Inventory と Ending Inventory の別々のメンバーに対して式を記述した場合、Essbase により、すべての月の期首在庫高が計算された後、すべての月の期末在庫高が計算されます。これは、期首在庫高の計算時に前月の期末在庫高の値を使用できないことを意味します。

実績値と予定値との差異または差異のパーセンテージの計算

@VAR および@VARPER 関数を使用して、予定値と実績値との差異または差異のパーセンテージを計算できます。

差異は、正または負の値になることがありますが、これは、会計次元のメンバー(支出アイテムまたは支出外アイテム)に関する差異を計算しているかどうかによって異なります:

- 支出アイテム。実績値が予定値に満たない場合(たとえば実際のコストが予定コストより低い場合)、Essbase に表示される差異は正の値になります。
- 支出外アイテム。実績値が予定値に満たない場合(たとえば売上実績が売上予想より低い場合)、Essbase に表示される差異は負の値になります。

Essbase では、デフォルトで、メンバーは支出外アイテムであると想定され、これに従って差異が計算されます。

➤ Essbase にメンバーが支出アイテムであると指定するには:

1 **アウトライン・エディタで、メンバーを選択します。**

このメンバーは、勘定科目としてタグ付けされた次元上に存在している必要があります。

2 **式エディタを開きます。**

Oracle Essbase Administration Services Online Help の「アウトライン内の式の作成および編集」を参照してください。

3 **メンバーに支出アイテムとしてタグ付けします。**

143 ページの「差異レポート・プロパティの設定」を参照してください。

@VAR または@VARPER 関数を使用しているとき、実績値が予定値より低い場合、Essbase に表示される差異は正の値になります。たとえば、Sample.Basic の Total Expenses の子は支出アイテムです。シナリオ次元の Variance および Variance%メンバーを使用して、実績値と予定値の差異を計算できます。図 115 を参照してください。

図 115　差異の例

```
Database: Basic (Current Alias Table: Default)
    Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
    Measures Accounts (Label Only)
        Profit (+) (Dynamic Calc)
            Margin (+) (Dynamic Calc)
            Total Expenses (-) (Dynamic Calc) (Expense Reporting)
                Marketing (+) (Expense Reporting)
                Payroll (+) (Expense Reporting)
                Misc (+) (Expense Reporting)
        Inventory (~) (Label Only)
        Ratios (~) (Label Only)
    Product
    Market
    Scenario (Label Only)
        Actual (+)
        Budget (~)
        Variance (~) (Dynamic Calc) (Two Pass Calc) @VAR(Actual, Budget);
        Variance % (~) (Dynamic Calc) (Two Pass Calc) @VARPER(Actual, Budget);
    Caffeinated Attribute
    Ounces Attribute
    Pkg Type Attribute
    Population Attribute
    Intro Date Attribute
```

値の割当て

割当て関数では、親レベルで入力された値を同じ次元または別の次元の子メンバーに割り当てることができます。割当ては、指定された様々な基準に基づいて行われます。

表 58　割当て関数のリスト

関数	割り当てる値
@ALLOCATE	メンバー、次元間メンバーまたは同じ次元内のメンバー・リストのメンバーからの値。
@MDALLOCATE	メンバー、次元間メンバーまたは複数の次元にまたがる値から生じる値。

@ALLOCATE を使用した計算スクリプトの例については513 ページの「製品をまたがるコストの割当て」を、@MDALLOCATE を使用した計算スクリプトの例については516 ページの「複数の次元での値の割当て」を参照してください。

予測関数

予測関数では、データを補間したり将来の値を計算したりする目的でデータを操作できます。表 59 に、値を予測する関数をリストします:

表 59　予測関数のリスト

関数	データの操作
@MOVAVG	データ・セットに移動平均を適用し、リスト内の各期間を末尾平均で置換します。 この関数では、データのセットが補整目的で変更されます。
@MOVMAX	データ・セットに移動最大を適用し、リスト内の各期間を末尾最大値で置換します。 この関数では、データのセットが補整目的で変更されます。
@MOVMED	データ・セットに移動メジアンを適用し、リスト内の各期間を末尾メジアンで置換します。 この関数では、データのセットが補整目的で変更されます。
@MOVMIN	データ・セットに移動最小を適用し、リスト内の各期間を末尾最小値で置換します。 この関数では、データのセットが補整目的で変更されます。
@MOVSUM	データ・セットに移動合計を適用し、各期間を末尾合計値で置換します。 この関数では、データのセットが補整目的で変更されます。
@MOVSUMX	データ・セットに移動合計を適用し、各期間を末尾合計値で置換します。合計する数字を操作する前に、メンバーに値を割り当てる方法を指定します。 この関数では、データのセットが補整目的で変更されます。
@SPLINE	一連のデータ・ポイントに平滑化スプラインを適用します。 スプラインとは、データの平滑化または補間を行うための数学曲線のことです。
@TREND	経過値に対する曲線近似に基づいて、将来の値を計算します。

個別の Essbase 関数については、『Oracle Essbase テクニカル・リファレンス』を参照してください。

統計関数

統計関数では、Essbase 内で高度な統計を計算できます。

表 60　統計関数のリスト

関数	計算する値
@CORRELATION	2 つの並列データ・セット間の相関係数
@COUNT	指定したデータ・セット内の値の総数
@MEDIAN	指定したデータ・セットにおけるメジアン、つまり中央値

関数	計算する値
@MODE	指定したデータ・セットにおけるモード、つまり最頻値
@RANK	指定したデータ・セットにおける指定したメンバーまたは値のランク
@STDEV	サンプルを基準にした、指定したメンバーの標準偏差
@STDEVP	母集団全体を基準にした、指定したメンバーの標準偏差
@STDEVRANGE	メンバーの範囲でクロスされた、指定したメンバーの標準偏差
@VARIANCE	サンプルを基準にした、指定したデータ・セットの差異
@VARIANCEP	母集団全体を基準にした、指定したデータ・セットの差異

日付と時刻関数

日付関数により、他の関数で日付を使用できます。

@TODATE: 日付文字列が、計算式で使用できる数値に変換されます

計算モード関数

計算モード関数では、Essbase で式の計算に使用される計算モードを指定できます。

@CALCMODE: Essbase で式の計算に使用される計算モード(セル、ブロック、ボトムアップまたはトップダウン)を指定する

注： CALCMODE 構成設定を使用して、データベース、アプリケーションまたはサーバーの各レベルで、計算モードを BLOCK または BOTTOMUP に設定することもできます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

カスタム定義関数

式および計算スクリプト内で使用するカスタム定義関数を作成して、Essbase 計算スクリプト言語ではサポートされていない計算を実行できます。カスタム定義関数は Java プログラミング言語で記述し、Essbase サーバーに登録する必要があります。Essbase 計算機フレームワークでは、カスタム定義関数は外部関数として呼び出されます。

カスタム定義関数は、計算スクリプト・エディタの関数ツリー内に表示されます。計算スクリプト・エディタを使用して、カスタム定義関数を選択したり、式に挿入したりできます。

第 32 章「カスタム定義計算関数の作成」を参照してください。

式での代替変数および環境変数の使用

代替変数は頻繁に変化する情報を参照する場合に使用されます。環境変数はユーザー固有のシステム設定のプレースホルダとして使用されます。参照:

- 398 ページの「式での代替変数の使用」
- 398 ページの「式での環境変数の使用」

式での代替変数の使用

代替変数は、期間情報など、定期的に変更される情報のプレースホルダとして機能します。データベース・アウトラインに適用する式の中で、代替変数を使用できます。

アウトラインの計算が完了すると、Essbase により、代替変数はユーザーが割り当てた値に置き換えられます。Administration Services、MaxL または ESSCMD を使用して、代替変数を作成したり、代替変数に値を割り当てたりできます。

代替変数は、サーバー・レベル、アプリケーション・レベルまたはデータベース・レベルで設定できます。Essbase では、計算スクリプトを実行しているアプリケーションおよびデータベースから、代替変数にアクセスできる必要があります。118 ページの「代替変数の使用」を参照してください。

式で代替変数を使用するには、アンパサンド(&)に続けて代替変数名を入力します。

Essbase では、&に続くテキスト文字列はすべて代替変数と見なされます。

たとえば、代替変数 UpToCurr が Jan:Jun と定義されているとします。次の@ISMBR 関数を条件付きテストの一部として使用できます:

```
@ISMBR(&UpToCurr)
```

Essbase では、アウトラインの計算時に、次のように代替変数が置き換えられます:

```
@ISMBR(Jan:Jun)
```

注： 新しいアウトライン・メンバーを表すための式内の代替変数は、アウトラインを保存するまで有効になりません。

式での環境変数の使用

アウトライン・メンバー式では、ユーザー固有のシステム設定のプレースホルダとしてシステム環境変数を使用できます。環境変数は、オペレーティング・システム・レベルで定義されているので、Essbase サーバー上のすべての式で使用できます。

式での環境変数の使用方法は、計算スクリプトで環境変数を使用する場合と同じです。491 ページの「計算スクリプトおよび式での環境変数の使用」を参照してください。

注： 環境変数は、集約ストレージ・アウトライン内の MDX クエリーやメンバー式では使用できません。

パーティションでの式の使用

パーティション・アプリケーションは、複数の Essbase サーバー、プロセッサまたはコンピュータにスパンできます。

パーティション化の際、ローカル・データベースで使用するときと同様に式を使用できます。ただし、あるデータベース内で使用する式が別のデータベースの値を参照する場合、Essbase では、式の計算時にこの 2 番目のデータベースからデータを取得する必要があります。このため、参照値が最新であることを確認し、データベース計算全体へのパフォーマンスの影響を慎重に検討してください。504 ページの「パーティション用の計算スクリプトの作成」を参照してください。

透過パーティションの場合、データ・ターゲットに対して式をどのように使用するかを慎重に検討してください。242 ページの「透過パーティションとメンバー式」および 239 ページの「透過パーティションのパフォーマンス上の考慮事項」を参照してください。

第 15 章「パーティション・アプリケーションの設計」と第 16 章「パーティションの作成および管理」を参照してください。

式の表示

➤ 式を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アウトライン内の式の作成および編集	Oracle Essbase Administration Services Online Help
ESSCMD	GETMBRCALC	『Oracle Essbase テクニカル・リファレンス』
MaxL	**query database**	『Oracle Essbase テクニカル・リファレンス』

24 ブロック・ストレージ・データベース用の式の例の確認

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 1028 ページの「集約ストレージ・アウトラインでの式の作成」
- 第 30 章「ブロック・ストレージ・データベース用の計算スクリプトの例の確認」

期間累計値の計算

アウトラインに勘定科目タグが付けられた次元が含まれている場合は、@PTD 関数を使用して期間累計値を計算できます。

この例では、次のように、Sample.Basic データベースのメジャー次元の Inventory 分岐を使用します:

```
   Inventory (~) (Label Only)
 Opening Inventory (+) (TB First) (Expense Reporting)
   IF(NOT @ISMBR(Jan))
 Additions (~) (Expense Reporting)
 Ending Inventory (~) (TB Last) (Expense Reporting)
```

年および現在の四半期の期間累計値を計算するには、年次元に次の 2 つのメンバーを追加します: 四半期累計用の QTD および年次累計用の YTD。例:

```
   QTD (~) @PTD(Apr:May)
YTD (~) @PTD(Jan:May);
```

現在の月が 5 月(May)である場合、次の式を QTD メンバーに追加します:

```
@PTD(Apr:May);
```

さらに次の式を YTD メンバーに追加します:

```
@PTD(Jan:May);
```

Essbase により、月の範囲の値が適宜合計されます。ただし、期首在庫高(Opening Inventory)にはタイム・バランス・タグ First があり、期末在庫高(Ending Inventory)にはタイム・バランス・タグ Last があります。Essbase では、これらの値が別々に処理されます。463 ページの「期首、期末および平均の値の計算」を参照してください。

表 61 は、Inventory 分岐内のメンバーおよび Sales メンバーについての計算結果の例を示しています:

表 61　結果: 期間累計値を計算するための計算スクリプト例

Measures->Time	Jan	Feb	3 月	4 月	5 月	QTD	YTD
Opening Inventory	100	110	120	110	140	**110**	**100**
Additions	110	120	100	160	180	**340**	**670**
Sales	100	110	110	130	190	**320**	**640**
Ending Inventory	110	120	110	140	130	**130**	**130**

Sales と Additions の値は合計されています。

Opening Inventory には First タグがあります。QTD の場合、Essbase では、当四半期内の最初の値である Apr が使用されます。YTD の場合、年の最初の値である Jan が使用されます。

Ending Inventory には Last タグがあります。QTD の場合、Essbase では、当四半期内の最後の値である May が使用されます。YTD の場合、年の最後の値でもある May が使用されます。

注： 動的時系列メンバーを使用して、期間累計値を計算することもできます。第 28 章「時系列データの計算」を参照してください。

移動値の計算

@AVGRANGE 関数を使用すれば、移動平均を計算できます。また、@ACCUM 関数を使用すれば、移動による年次累計値を計算できます。

たとえば、データベースに、毎月の Sales のデータ値が含まれており、データベース・アウトラインに、メンバー AVG_Sales と YTD_Sales が登録されているとします。

AVG_Sales メンバーに次の式を追加します:

```
@AVGRANGE(SKIPNONE, Sales, @CURRMBRRANGE(Year, LEV, 0, , 0));
```

さらに次の式を YTD_Sales メンバーに追加します:

```
@ACCUM(Sales);
```

Essbase により、時間のタグが付けられた次元内の各月の Sales 値の平均が計算されます。SKIPNONE パラメータは、#MISSING 値を含むすべての値が含まれることを意味します。結果は、Essbase によって AVG_Sales 内に置かれます。989 ページの「#MISSING 値の集計」を参照してください。

表 62 は、Essbase で累計売上の値が計算されてから、YTD_Sales 内に結果が置かれる際の結果を示しています:

表 62　結果: 移動値を計算するための計算スクリプト例

Measures->Time	1 月	Feb	Mar	第 1 四半期
Sales	100	200	300	**600**
AVG_Sales	100	150	200	#MISSING
YTD_Sales	100	300	600	#MISSING

AVG_Sales の値は、月次累計の平均です。たとえば、AVG_Sales -> Mar は 1 月、2 月および 3 月の売上(Sales)の平均です。

YTD_Sales は当月までの値の累積値です。したがって、YTD_Sales -> Feb は Sales -> Jan および Sales -> Feb の合計になります。

毎月の資産変動の計算

@PRIOR 関数を使用すると、前月の値に基づいて値を計算できます。

たとえば、データベースに月次ベースで保管される資産のデータ値が含まれているとします。連続する月の資産値の差分(資産変動)は、当月の値から前月の値を減算することで計算できます。

データベースでは、次の 3 つのメンバーで資産値が管理されているものとします:

- 毎月の資産値を表す Assets
- 資産変動値を表す Asset_MVNT
- 年初の資産値を表す Opening_Balance

1 月の Asset_MVNT 値は、1 月の値から Opening_Balance 値を減算して算出されます。

次の式を Asset_MVNT メンバーに追加します:

```
  IF(@ISMBR(Jan)) Asset_MVNT = Assets - Opening_Balance;
ELSE Asset_MVNT = Assets - @PRIOR(Assets);
```

```
ENDIF;
```

表 63 は、連続する月の資産値の差分を、Essbase で計算した結果を示したものです:

表 63　結果: 毎月の資産変動を計算するための計算スクリプト例

Assets -> Time	Opening_Balance	1 月	Feb	3 月
Assets	1200	1400	1300	1800
Asset_MVNT		**200**	**-100**	**500**

Essbase では、月の値を循環して、次の計算が実行されます:

1. IF ステートメントと@ISMBR 関数により、年次元の現在のメンバーが 1 月であるかどうかが確認されます。この確認が必要なのは、1 月の Asset_MVNT 値は、前月の値の減算では算出できないためです。
2. 年次元の現在のメンバーが 1 月である場合、Essbase で、Jan -> Assets の値から Opening_Balance の値が減算され、結果が Jan -> Asset_MVNT に置かれます。
3. 年次元の現在のメンバーが 1 月でない場合、@PRIOR 関数は前月の資産値を取得します。Essbase により、現在の月の資産から前月の資産が減算されます。結果は、現在の月の Asset_MVNT 値に入力されます。

#MISSING 値のテスト

データベース内の#MISSING 値をテストできます。989 ページの「#MISSING 値の集計」を参照してください。

データベース・アウトラインに Commission という名前のメンバーが含まれているとします。現在のメンバーの組合せの Sales の値が#MISSING でない場合、売上高の 10%の歩合(Commission)が支払われます。データベース・アウトライン内の Commission メンバーに適用した場合、次の式で歩合が計算されます:

```
   IF(Sales <> #MISSING) Commission = Sales * .1;
 ELSE Commission = #MISSING;
ENDIF;
```

計算スクリプト内に式を記述する場合は、その式を次のように Commission メンバーと関連付ける必要があります:

```
   Commission(IF(Sales <> #MISSING) Commission = Sales * .1;
 ELSE Commission = #MISSING;
ENDIF;);
```

Essbase では、データベース内を循環して、次の計算が実行されます:

1. IF ステートメントで、現在のメンバーの組合せに対する Sales メンバーの値が確認されます。

2. Sales が#MISSING でない場合は、Essbase で、Sales メンバーの値と 0.1 とが乗算され、結果が Commission メンバーに置かれます。
3. Sales が#MISSING である場合、Essbase によって、Commission メンバーに #MISSING が置かれます。

属性の式の計算

データベース内の属性次元メンバーに対して特定の計算を実行できます。176 ページの「属性データの計算」を参照してください。

たとえば、オンス・サイズの製品のオンスごとの収益性を計算するには、計算式で@ATTRIBUTEVAL 関数を使用します。Sample.Basic データベースのメジャー次元の Ratios 分岐には、Profit per Ounce という名前のメンバーが含まれています。このメンバーに対する式は、次のとおりです:

```
Profit/@ATTRIBUTEVAL(@NAME(Ounces));
```

Essbase では、製品次元を循環して、次の計算が実行されます:

1. Ounces 属性次元のメンバーに関連付けられている各基本メンバーに対して、@ATTRIBUTEVAL 関数が数値属性値(たとえば、Ounces の下のメンバー 12 に対して 12)を戻します。

   注： 文字列 Ounces は、@NAME 関数で処理してから@ATTRIBUTEVAL 関数に渡す必要があります。
2. 次に、Essbase により、Profit が@ATTRIBUTEVAL の結果で除算され、Profit per Ounce が得られます。

注： 180 ページの「計算式での属性の使用」を参照してください。
@ATTRIBUTEVAL 関数の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

25 計算順序の定義

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 62 ページの「疎次元および密次元」
- 61 ページの「世代とレベル」
- 451 ページの「動的計算による計算順序の変更の理解」

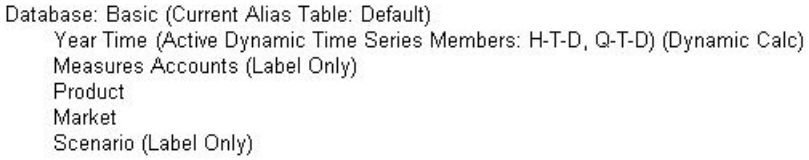
データ・ブロック内のデータ・ストレージ

Essbase では、データ値はデータ・ブロック内に保管されます。Essbase では、疎次元メンバーの一意の組合せごとに、1 データ・ブロックが作成されます(ただし、その組合せに対して、1 つ以上のデータ値が存在する場合に限ります)。

各データ・ブロックには、疎次元メンバーの一意の組合せに対するすべての密次元メンバーの値が含まれます。

Sample.Basic データベース内の年次元、メジャー次元およびシナリオ次元は密次元、製品次元と市場次元は疎次元です。図 116 は、Sample.Basic データベース内の次元のアウトラインを示しています:

図 116　Sample.Basic データベース内の次元



注： Sample.Basic には、5 つの属性次元も含まれます。これらの次元は疎な動的計算次元です。つまり、その属性データはデータベースに保管されません。第 10 章「属性の操作」を参照してください。

Essbase では、製品次元と市場次元の一意のメンバーの組合せごとにデータ・ブロックが 1 つ作成されます(ただし、その組合せに対して、1 つ以上のデータ値が存在する場合にかぎります)。たとえば、100-10, New York の組合せに対して 1 つのデータ・ブロックが作成されます。このデータ・ブロックには、100-10, New York の年次元、メジャー次元およびシナリオ次元の値がすべて含まれます。図 117 は、Sample.Basic データベースの「Product」次元と「Market」次元のアウトラインを示しています:

図 117　Sample.Basic データベースの「Product」次元と「Market」次元

Essbase では、メンバーの組合せは次元間演算子で示されます。次元間演算子の記号は->(ハイフンと大なり記号)です。このため、100-10, New York は 100-10 -> New York と記述します。

データ・ブロックを分類する手順は、次のとおりです:

- 入力

  これらのブロックは、任意のブロック内のセルにデータをロードすることによって作成されます。少なくとも 1 つの疎メンバーが親レベルのメンバーに存在する場合、(1)疎なレベル 0 メンバーの組合せ、または(2)疎な上位レベル・メンバーの組合せに対して入力ブロックを作成できます。入力ブロックはレベル 0 または上位レベル・ブロックになります。

- 非入力

  これらのブロックは、計算によって作成されます。たとえば Sample.Basic では、疎な計算プロセスで East -> Cola ブロックが作成されます(つまり、このブロックは計算前は存在していなかったことになります)。

- レベル 0

  これらのブロックは、すべての疎メンバーがレベル 0 メンバーであるとき、疎メンバーの組合せに対して作成されます。たとえば Sample.Basic で、New

York -> Cola はレベル 0 ブロックです。これは、New York と Cola がそれぞれの疎次元のレベル 0 メンバーであるためです。レベル 0 ブロックは、入力ブロックでも非入力ブロックでもかまいません。たとえば、レベル 0 の非入力ブロックは、親レベルでデータをロードした後、レベル 0 までデータを割り当てる割当てプロセスで作成されます。

- 上位レベル

  これらのブロックは、少なくとも 1 つの疎メンバーが親レベルのメンバーに存在する場合に、疎メンバーの組合せに対して作成されます。上位レベル・ブロックは、入力ブロックでも非入力ブロックでもかまいません。

61 ページの「世代とレベル」および 70 ページの「データ・ブロックおよびインデックス・システム」を参照してください。

メンバーの計算順序

Essbase では、データ・ブロック・レベルでデータベースの計算が行われます。その際、1 つ以上のブロックがメモリーに収容され、ブロック内の要求された値が計算されます。ブロックの計算は、ブロック番号に従って順番に行われます。データベース・アウトラインで、ブロックの順序を Essbase に指示します。各ブロック内では、データベース・アウトライン内の階層の順序で値が計算されます。したがって、Essbase では全体的に、データベース・アウトラインに基づいてデータベースの計算が行われます。

データベース上でデフォルト計算(CALC ALL)を実行する場合、Essbase では、次の順序で次元が計算されます:

- 勘定科目タグが付けられた次元と時間タグが付けられた次元がどちらも存在し、会計次元上のメンバーに式が適用される場合、Essbase では次の順序で計算が行われます:
  1. 勘定科目タグが付けられた次元
  2. 時間タグが付けられた次元
  3. その他の密次元(データベース・アウトラインに表示されている順)
  4. その他の疎次元(データベース・アウトラインに表示されている順)
- それ以外の場合、Essbase では、次の順序で計算が行われます:
  1. 密次元(データベース・アウトラインに表示されている順)
  2. 疎次元(データベース・アウトラインに表示されている順)

注： 属性次元は、データベース集計に含まれていないので、計算順序には影響しません。第 10 章「属性の操作」を参照してください。

Sample.Basic データベースでは、次元は、メジャー、年、シナリオ、製品、市場の順で計算されます。

計算スクリプトを使用して、デフォルトの順序を上書きできます。第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。

メンバーの関係の影響の理解

各次元内の計算の順序は、データベース・アウトライン内のメンバーの関係次第です。次元の各分岐内では、レベル 0 の値が最初に計算され、続いてその親であるレベル 1 の値が計算されます。次に、次の分岐のレベル 0 の値が計算され、その親であるレベル 1 の値が計算されます。このようにして、すべてのレベルの計算が完了するまで計算が続行されます。

図 118 は、Sample.Basic データベースの年次元を示しています。左側に計算順序が表示されています。この例では、親メンバーに動的計算のタグが付いていないと想定します。第 27 章「データ値の動的計算」を参照してください。

図 118　Sample.Basic データベースの年次元

```
17  Year Time
4       Qtr1 (+)
1           Jan (+)
2           Feb (+)
3           Mar (+)
8       Qtr2 (+)
5           Apr (+)
6           May (+)
7           Jun (+)
12      Qtr3 (+)
9           Jul (+)
10          Aug (+)
11          Sep (+)
16      Qtr4 (+)
13          Oct (+)
14          Nov (+)
15          Dec (+)
```

Jan は最初の分岐の最初のメンバーです。Jan は式を持たないので、計算されません。同じ分岐内の残りの 2 つのメンバーである Feb と Mar も同様です。

Essbase では、Qtr1 を計算するために Jan、Feb および Mar が集計されます。この例では、これらのメンバーが加算されます。

Essbase では、次に、同じ方法で Qtr2 から Qtr4 までの分岐が計算されます。

Essbase では、最後に、Qtr1 から Qtr4 までの値を集計することにより、年メンバーが計算されます。これらのメンバーが追加されます。

メンバーの集計の決定

Essbase でのメンバーの集計方法を決めるには、データベース・アウトラインのメンバーに対して算術演算子(+、-、/、*、%、~、^)を使用します。

勘定科目メンバーにタイム・バランス・タグ(First、Last または Average)が設定されている場合、Essbase ではそれに従って集計が行われます。463 ページの「期首、期末および平均の値の計算」を参照してください。

親メンバーにラベルのみ演算子が設定されている場合、Essbase では、その子から親は計算されません。

メンバーに~演算子が設定されている場合、Essbase では、そのメンバーはその親の集計には使用されません。

メンバーに^演算子が設定されている場合、Essbase では、どの次元のメンバーも集計されません。

注： 動的計算を使用する場合、Essbase では、異なる計算順序が使用されることがあります。451 ページの「動的計算の計算順序」を参照してください。

データベース・アウトライン内の次元の順序付け

必要な計算結果を確実に得るには、次のどちらのタスクを実行する場合にも、データベース・アウトライン内の次元の計算順序を考慮してください:

- 算術演算子を使用して、データベース・アウトライン内のメンバーに対する除算(/)、乗算(*)またはパーセンテージ(%)の計算を行う場合。
- データベース・アウトライン内のメンバーに式を設定する場合。

データベース・アウトライン内のメンバーの加算(+)や減算(-)の算術演算子のみを使用していて、アウトライン内の式を使用しない場合は、計算順序を考慮する必要はありません。

データベース・アウトライン内のメンバーへの式の設定

データベース・アウトライン内のメンバーに式を設定する場合、次元の計算順序を考慮してください。ある次元のメンバーに設定された式は、別の次元に対する後続の計算によって上書きされることがあります。

たとえば、Sample.Basic データベースには、勘定科目タグが付けられたメジャー次元と、時間タグが付けられた年次元があります。最初にメジャーが計算され、次に年が計算されます。メジャー次元の Margin に式を設定した場合、Essbase では、メジャー次元の計算時に式が計算されます。この式は、さらに年次元の集計時に上書きされます。416 ページの「セルの計算順序」を参照してください。

算術演算子*、/および%の使用

計算演算子を使用して、データベース・アウトライン内のメンバーの乗算(*)、除算(/)およびパーセンテージの計算(%)を行う場合、次元の計算順序を考慮してください。必要な計算済の値が、別の次元に対する後続の計算によって上書きされることがあります。

たとえば、Sample.Basic データベースには、勘定科目タグが付けられたメジャー次元と、時間タグが付けられた年次元があります。最初にメジャーが計算され、次に年が計算されます。メジャー次元のメンバーを乗算した場合、計算された結果は、Essbase で年次元の値が集計されるときに上書きされることがあります。416 ページの「セルの計算順序」を参照してください。

乗算(*)、除算(/)またはパーセンテージ(%)の演算子を使用してメンバーを集計するときは、必要な結果を得られるように、分岐内のメンバーを慎重に順序付けてください。

図 119 は、算術演算子がアウトラインに表示される順序を示しています。ここでは、ユーザーが Child 2 と Child 3 の合計を Child 1 で除算すると想定します。ただし、Child 1 が最初のメンバーである場合、Essbase では Child 1 から処理が開始され、値#MISSING が Child 1 で除算されます。結果は#MISSING です。次に、Essbase では、Child 2 と Child 3 が加算されます。明らかに、これは必要としていた結果ではありません。

図 119　データベース・アウトラインでの算術演算子

```
Parent 1
    Child 1 (/)
    Child 2 (+)
    Child 3 (+)
```

正しい結果を算出するには、Child 1 を分岐の最後のメンバーにしてください。

データベース・アウトライン上のメンバーに式を適用しても、同じ結果が得られます。ただし、図 119 に示すように、メンバーに対してこれらの計算演算子を使用するほうがはるかに効率的です。

前方計算参照の回避

予想どおりの計算結果を得るには、アウトラインに前方計算参照が含まれていないことを確認します。前方計算参照は、計算メンバーの値が Essbase でまだ計算されていないメンバーに依存している場合に発生します。このような場合、Essbase では、必要な計算結果が生成されない可能性があります。

たとえば、図 120 に示される Product 次元について検討してください。この次元には、3 つの前方計算参照(2 つの共有メンバー(P100-20 および P300-20)と 1 つの非共有メンバー(P500-20))があります:

図 120　前方計算参照のある Product 次元

```
Product
    Diet (~)
        P100-20 (+) (Shared Member)
        P200-20 (+) (Shared Member)
        P300-20 (+) (Shared Member)
        P400-20 (+) "P200-10"*2;
        P500-20 (+) ("P200-20"+"P300-20");
    Regular (+)
        P100 (+)
            P100-10 (+)
            P100-20 (+)
                P100-20-01 (+)
                P100-20-02 (+)
        P200 (+)
            P200-10 (+)
            P200-20 (+)
        P300 (+)
            P300-10 (+)
            P300-20 (+) "P100-20"+"P300-20";
```

アウトライン・エディタでアウトラインを確認するとき、Essbase により、前方計算参照を持つ共有メンバーが識別されます。アウトラインの確認では、前方計算参照を持つ非共有メンバーは識別されません。前方計算参照を含むアウトラインは保存して使用できます。

➤ アウトラインを確認する方法については、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アウトラインの確認」を参照してください。

Diet の下位の 5 つのメンバーについて考えます。メンバー P100-20、P300-20 および P500-20 には、前方計算参照があります:

- P100-20 (+) (共有メンバー): Essbase では、実績メンバー P100-20 が計算される前に共有メンバー P100-20 が計算されます。実績メンバー P100-20 には子があるので、Essbase で共有メンバー P100-20 を正確に計算するには、実績メンバー P100-20 の子を加算して、実績メンバーを先に計算する必要があります。
- P300-20 (+) (共有メンバー): Essbase では、実績メンバー P300-20 が計算される前に共有メンバー P300-20 が計算されます。実績メンバー P300-20 には式が設定されているので、Essbase で共有メンバー P300-20 を正確に計算するには、実績メンバーを先に計算する必要があります。
- P500-20 (+) ("P200-20" + "P300-20"): P500-20 に適用された式は、Essbase でまだ計算されていないメンバーを参照します。参照メンバー P300-20 には固有の式があり、Essbase で P500-20 を正確に計算するには、P300-20 を先に計算する必要があります。メンバー P200-20 および P400-20 は、前方計算参照を持たないため正しく計算されます。
- P200-20 (+) (共有メンバー): Essbase では、実績メンバー P200-20 の計算前に共有メンバー P200-20 が計算されますが、P200-20 は前方計算参照ではありません。実績メンバー P200-20 には計算の依存関係はありません(子や式がない)。このため、Essbase で、共有メンバーの前に実績メンバーを計算する必要はありません。Essbase では、単純に実績メンバーの値が使用されます。
- P400-20 (+) ("P200-10" * 2): P400-20 に適用された式は Essbase でまだ計算されていないメンバーを参照しますが、P400-20 は前方計算参照ではありません。式で参照されるメンバー自体には計算の依存関係はありません。P200-10 が式の唯一のメンバーであり、P200-10 自体には子や式はありません。Essbase では、P400-20 が正確に計算されます。

P100-20、P300-20 および P500-20 の正しい計算結果を得るには、アウトライン内のメンバーの順序を変更します。図 121 に示すように、Regular メンバーの後ろに Diet 共有メンバーを置くと、Essbase では、必要な順序でメンバーが計算されます。

図 121　前方計算参照のない変更された製品次元

```
Product
    Regular (+)
        P100 (+)
            P100-10 (+)
            P100-20 (+)
                P100-20-01 (+)
                P100-20-02 (+)
        P200 (+)
            P200-10 (+)
            P200-20 (+)
        P300 (+)
            P300-10 (+)
            P300-20 (+) "P100-20"+"P300-20"
    Diet (~)
        P100-20 (+) (Shared Member)
        P200-20 (+) (Shared Member)
        P300-20 (+) (Shared Member)
        P400-20 (+) "P200-10"*2;
        P500-20 (+) ("P200-20"+"P300-20");
```

Essbase では、次が計算されます:

- 共有メンバー P100-20 の計算前に、実績メンバー P100-20 が計算されます。したがって、P100-20 には前方計算参照がありません。
- 共有メンバー P300-20 の前に、実績メンバー P300-20 が計算されます。したがって、P300-20 には前方計算参照がありません。
- メンバー P500-20 の前に、式を持つ参照メンバー P300-20。したがって、P500-20 には前方計算参照がありません。

ブロックの計算順序

Essbase では、ブロックの番号順にブロックを計算します。Essbase では、データベース・アウトラインの最初の疎次元を開始点とします。この最初の次元から疎メンバーの組合せが定義されます。

Sample.Basic データベースでは、製品がデータベース・アウトラインの最初の疎次元です。

図 122　Sample.Basic データベースの次元

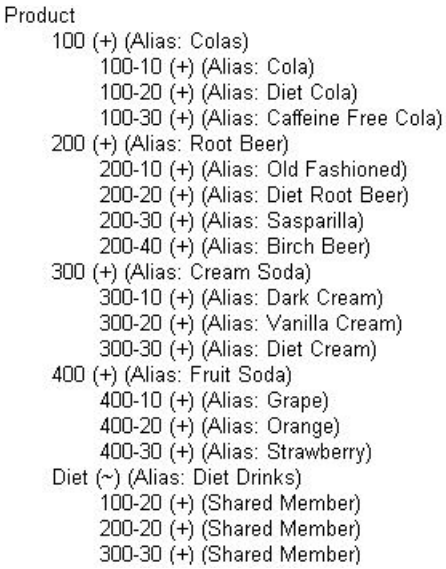
```
Database: Basic (Current Alias Table: Default)
    Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
    Measures Accounts (Label Only)
    Product
    Market
    Scenario (Label Only)
```

注：　Sample.Basic アウトラインの属性次元(上の図にはありません)は、データベース集計に組み込まれていないので、ブロックの計算順序に影響しません。第 10 章「属性の操作」を参照してください。

図 123 に示すように、製品には 19 個のメンバーがあります(共有メンバーを除く。Essbase では共有メンバーに対してデータ・ブロックを作成しません)。したがっ

て、データベースの最初の 19 個のデータ・ブロックは、製品次元のメンバーの計算順序に従って番号付けされます。

図 123　Sample.Basic データベースの製品次元



もう 1 つの疎次元は市場です。最初の 4 個のデータ・ブロックには、計算対象である市場次元の最初のメンバー(New York)が含まれています。表 64 は、これら 19 のデータ・ブロックの最初に 5 つについて、各製品次元と New York の疎メンバーの組合せを示しています:

表 64　疎メンバーの組合せ: データ・ブロック 0 から 4

ブロック番号	製品のメンバー	市場のメンバー
0	Cola (100-10)	New York
1	Diet Cola (100-20)	New York
2	Caffeine Free Cola (100-30)	New York
3	Colas (100)	New York
4	Old Fashioned (200-10)	New York

市場次元の次のメンバーは Massachusetts です。Essbase では、製品の各メンバーと Massachusetts の疎な組合せに対して、その次の 19 個のデータ・ブロックを作成します。表 65 は、ブロック番号 19 から 23 について、疎メンバーの組合せを示しています:

表 65　疎メンバーの組合せ: データ・ブロック 19 から 23

ブロック番号	製品のメンバー	市場のメンバー
19	Cola (100-10)	Massachusetts
20	Diet Cola (100-20)	Massachusetts

ブロック番号	製品のメンバー	市場のメンバー
21	Caffeine Free Cola (100-30)	Massachusetts
22	Colas (100)	Massachusetts
23	Old Fashioned (200-10)	Massachusetts

Essbase では、少なくとも 1 つのデータ値が存在する疎次元メンバーのすべての組合せに対してブロックが作成されるまで、処理を続行します。

Essbase では、ブロックに 1 つ以上の値が存在する場合にのみ、データ・ブロックを作成します。たとえば、Massachusetts の Old Fashioned Root Beer (200-10)にデータ値がない場合、Essbase では、200-10 -> Massachusetts のデータ・ブロックを作成しません。ただし、将来そのメンバーの組合せに対してデータがロードされる場合に備えて、Essbase では 200-10 -> Massachusetts の該当するブロック番号が予約されます。

データベースでデフォルトの計算(CALC ALL)を実行すると、各ブロックは、そのブロック番号に従って順番に処理されます。高機能計算をオンにしていて、ブロックを計算する必要がない場合、Essbase では、そのブロックをスキップして、次のブロックに移動します。高機能計算を使用してパフォーマンスを最適にする方法は、第 26 章「高機能計算についての理解」を参照してください。

データ・ブロック番号の再設定

Essbase では、次の変更が行われた場合にデータ・ブロック番号が再設定されます:

- 疎次元の移動
- 疎次元の追加
- 疎次元への密次元の変更
- 疎次元における任意のメンバーの移動
- 疎次元における任意のメンバーの削除
- 疎次元へのメンバーの追加

セルの計算順序

各データ・ブロックには、疎次元メンバーの一意の組合せに対するすべての密次元メンバーの値が含まれます。各データ値は、データ・ブロックのセルにあります。

Essbase によって各ブロック内のセルが計算される順序は、データベースの構成方法によって異なります。データベースの構成方法によって、各ブロック内の密次元メンバーのメンバー計算順序が定義されます。また、疎次元メンバーを表すブロックの計算順序も定義されます。

次の例を参照してください。

セルの計算順序: 例 1

最も単純な事例のこの例では、次の条件が満たされます:

- 時間または勘定科目タグが付けられた次元はありません。
- #MISSING 値を集計する設定がオンになっています。
- 市場と年は密次元です。

  Essbase では、データベース・アウトラインに定義された順序で密次元が計算されます。データベース・アウトラインで、年次元が市場次元より前に配置されているとすると、年次元は市場次元より先に計算されます。

表 66 に、データ・ブロック内のセルのサブセットを示します:

表 66　計算順序の例 1: 入力セルと計算済セル

Year-Market	New York	Massachusetts	East
Jan	112345	68754	**3**
Feb	135788	75643	**4**
Mar	112234	93456	**5**
Qtr1	**1**	**2**	**6**

データ値は次の入力セルにロードされています:

- Jan -> New York
- Feb -> New York
- Mar -> New York
- Jan -> Massachusetts
- Feb -> Massachusetts
- Mar -> Massachusetts

Essbase は次のセルを計算します。表 66 で、これらのセルの計算順序はセルに表示される 1 から 6 までの数字で示されます:

1. Qtr1 -> New York
2. Qtr1 -> Massachusetts
3. Jan -> East
4. Feb -> East
5. Mar -> East
6. Qtr1 -> East

Qtr1 -> East には複数の集計パスがあり、市場または年で集計できます。市場で集計した場合、Qtr1 -> New York と Qtr1 -> Massachusetts の集計になります。年で集計した場合、Jan -> East、Feb -> East および Mar -> East の集計になります。

Essbase では、Qtr1 -> East に複数の集計パスがあることが認識されます。このため、表 67 に示すように、Qtr1 -> East は Qtr1 の値を集計することで 1 回のみ計算され、最後に計算された次元(この例では、市場次元)の集計パスが使用されます。

表 67　計算順序の例 1: 結果

Year-Market	New York	Massachusetts	East
Jan	112345	68754	**181099**
Feb	135788	75643	**211431**
Mar	112234	93456	**205690**
Qtr1	**360367**	**237853**	**598220**

計算順序に基づいて、データベース・アウトラインの Qtr1 にメンバー式を設定した場合、その式は、Qtr1 -> East の計算時に Essbase によって無視されます。データベース・アウトラインの East にメンバー式を設定した場合、その式は、Essbase によって市場の集計パスに基づいて Qtr1 -> East が集計されるときに計算されます。

必要に応じて、計算スクリプトを使用して、ユーザーの選択順で次元を計算できます。第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。

セルの計算順序: 例 2

この例では、次の条件が満たされます:

- 時間または勘定科目タグが付けられた次元はありません。
- #MISSING 値を集計する設定がオフ(デフォルト)になっています。
- 市場と年は密次元です。

  Essbase では、データベース・アウトラインに定義された順序で密次元が計算されます。データベース・アウトラインで、年次元が市場次元より前に配置されているとすると、年次元は市場次元より先に計算されます。

表 68 に、データ・ブロック内のセルのサブセットを示します:

表 68　計算順序の例 2: 入力セルと計算済セル

Year-Market	New York	Massachusetts	East
Jan	112345	68754	**4**
Feb	135788	75643	**5**
Mar	112234	93456	**6**
Qtr1	**1**	**2**	**3/7**

データ値は次の入力セルにロードされています:

- Jan -> New York
- Feb -> New York
- Mar -> New York
- Jan -> Massachusetts
- Feb -> Massachusetts
- Mar -> Massachusetts

Essbase は New York、Massachusetts および East の Qtr1 セルと、Jan、Feb および March の East セルを計算します。表 68 で、これらのセルの計算順序はセルに表示される 1 から 7 までの数字で示されます:

1. Qtr1 -> New York
2. Qtr1 -> Massachusetts
3. Qtr1 -> East
4. Jan -> East
5. Feb -> East
6. Mar -> East
7. Qtr1 -> East

Qtr1 -> East は、年と市場の両方の集計パスに基づいて計算されます。最初に、Qtr1 -> New York および Qtr1 -> Massachusetts の集計として Qtr1 -> East が計算されます。2 番目に、Jan -> East、Feb -> East および Mar -> East の集計として Qtr1 -> East が計算されます。

結果は、表 69 に示すように、例 1 の結果と同じです(表 67 を参照)。ただし、Qtr1 -> East は 2 回計算されています。このことは、データを親レベルでロードする必要があるときに重要です(420 ページの「セルの計算順序: 例 3」を参照)。

表 69　計算順序の例 2: 結果

Year-Market	New York	Massachusetts	East
Jan	112345	68754	**181099**
Feb	135788	75643	**211431**
Mar	112234	93456	**205690**
Qtr1	**360367**	**237853**	**598220**

計算順序に基づいて、データベース・アウトラインで Qtr1 にメンバー式を設定した場合、その結果は、Essbase によって市場の集計パスに基づいて Qtr1 -> East が集計されるときに上書きされます。データベース・アウトラインで East にメンバー式を設定した場合、市場の集計パスは最後に計算されるので、その結果は維持されます。

セルの計算順序: 例 3

この例では、次の条件が満たされます:

- 時間または勘定科目タグが付けられた次元はありません。
- #MISSING 値を集計する設定がオフ(デフォルト)になっています。
- データ値は親レベルでロードされています。
- 市場と年は密次元です。

  Essbase では、データベース・アウトラインに定義された順序で密次元が計算されます。データベース・アウトラインで、年次元が市場次元より前に配置されているとすると、年次元は市場次元より先に計算されます。

表 70 に、データ・ブロック内のセルのサブセットを示します:

表 70　計算順序の例 3: 入力セルと#MISSING 値

Year-Market	New York	Massachusetts	East
Jan	#MISSING	#MISSING	**181099**
Feb	#MISSING	#MISSING	**211431**
Mar	#MISSING	#MISSING	**205690**
Qtr1	#MISSING	#MISSING	

セルは、418 ページの「セルの計算順序: 例 2」と同じ順序で計算されます。Qtr1 -> East は、年と市場の両方の集計パスに基づいて計算されます。

#MISSING 値を集計する設定がオフなので、Essbase では、#MISSING 値は集計されません。このため、親レベルでロードされているデータは、それより下の #MISSING 値によって上書きされません。

ただし、子のデータ値のいずれかが#MISSING でない場合、その値は集計されるので、その値によって親の値は上書きされます。たとえば、Jan -> New York が 50000.00 の場合、親レベルでロードされている値はこの値によって上書きされます。

結果は、表 71 に示すように、Essbase では最初に Jan -> East、Feb -> East および Mar -> East を集計することによって、Qtr1 -> East セルを正しく計算してから、市場の集計パスに基づいて計算します。ただし、Qtr1 -> New York および Qtr1 -> Massachusetts の#MISSING 値は集計されないので、Qtr1 -> East の値は上書きされません。

表 71　計算順序の例 3: 結果

Year-Market	New York	Massachusetts	East
Jan	#MISSING	#MISSING	**181099**
Feb	#MISSING	#MISSING	**211431**
Mar	#MISSING	#MISSING	**205690**

Year-Market	New York	Massachusetts	East
Qtr1	#MISSING	#MISSING	**598220**

Essbase では、Qtr1 -> East セルの値を確実に計算するために、セルを 2 回計算する必要があります。Qtr1 -> East が最後の集計パスのみに基づいて計算された場合、その結果は#MISSING となり必要な結果が得られません。

セルの計算順序: 例 4

この例では、次の条件が満たされます:

- 年次元には時間タグが付いています。
- メジャー次元には勘定科目タグが付いています。

  Essbase では、最初に勘定科目タグが付けられた次元を計算し、次に時間タグが付けられた次元を計算します。したがって、この例では、メジャーが年より先に計算されます。
- #MISSING 値を集計する設定がオフ(デフォルト)になっています。
- Marketing、Payroll および Misc Expenses の値は、Qtr1 (親レベル)でロードされています。

図 124 は、Sample.Basic データベースのメジャー次元の Profit 分岐を示しています。この例では、Total Expenses が動的計算メンバーではないと仮定しています。

図 124　メジャー次元の Profit 分岐

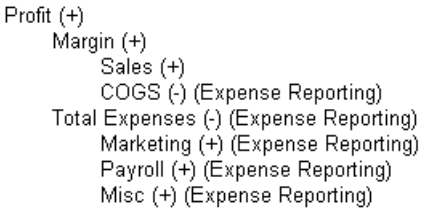


表 72 に、データ・ブロック内のセルのサブセットを示します:

表 72　計算順序の例 4: 入力セル、#MISSING 値と計算済セル

メジャー->年	1 月	Feb	Mar	第 1 四半期
Sales	31538	32069	32213	**13**
COGS	14160	14307	14410	**14**
Margin	**1**	**4**	**7**	**10/15**
Marketing	#MISSING	#MISSING	#MISSING	15839
Payroll	#MISSING	#MISSING	#MISSING	12168
Misc	#MISSING	#MISSING	#MISSING	233

メジャー->年	1 月	Feb	Mar	第 1 四半期
Total Expenses	**2**	**5**	**8**	**11/16**
Profit	**3**	**6**	**9**	**12/17**

次のセルには、複数の集計パスがあります:

- Margin -> Qtr1
- Total Expenses -> Qtr1
- Profit -> Qtr1

#MISSING 値を集計する設定がオフなので、Essbase では、#MISSING 値は集計されません。このため、親レベルでロードされているデータは、#MISSING 値によって上書きされず、複数の集計パスがあるセルが Essbase によって 2 回計算されます。

結果を表 73 に示します:

表 73　計算順序の例 4: 結果

メジャー->年	1 月	Feb	Mar	第 1 四半期
Sales	31538	32069	32213	**95820**
COGS	14160	14307	14410	**42877**
Margin	**17378**	**17762**	**17803**	**52943**
Marketing	#MISSING	#MISSING	#MISSING	15839
Payroll	#MISSING	#MISSING	#MISSING	12168
Misc	#MISSING	#MISSING	#MISSING	233
Total Expenses				**28240**
Profit	**17378**	**17762**	**17803**	**12/17**

計算順序に基づいて、データベース・アウトラインの Margin などにメンバー式を設定した場合、その結果は、Qtr1 に関する集計で上書きされます。

密次元に対する式のセル計算順序

データ・ブロック内のセルの計算順序は、メンバーの式の影響を受けません。Essbase では、データ・ブロック内で式が検出されると、他の必要なデータ・ブロックがロックされ、その式が計算され、データ・ブロックの計算が続行されます。

密次元のメンバーに式を設定するときには、セルの計算順序に十分注意してください。前の例で説明したように、複数の集計パスがあるセルでは、最後に計算された次元によって以前に行われたセルの計算が上書きされます。必要に応じて、計算スクリプトを使用して、次元の計算順序を変更できます。第 29 章「ブロッ

ク・ストレージ・データベース用の計算スクリプトの作成」および第 23 章「ブロック・ストレージ・データベース用の式の作成」を参照してください。

計算パス

Essbase では、可能なかぎり 1 つの計算パスでデータベースが計算されます。つまり、必要な各データ・ブロックがメモリーに 1 回のみ読み取られ、データ・ブロックに対して関連するすべての計算が実行され、保存されます。ただし、場合によっては、Essbase で複数の計算パスを使用してデータベースを計算する必要があります。その場合、後続の計算パスでは、Essbase によってデータ・ブロックがメモリーに戻され、そのデータ・ブロックに対して次の計算が実行され、再度保存されます。

デフォルトのデータベースのフル計算(CALC ALL)を実行すると、Essbase では、1 つの計算パスでデータベースを計算しようとします。勘定科目または時間タグが付けられた次元がある場合は、Essbase で、複数の計算パスを実行してデータベースを計算する必要がある場合があります。

表 74 は、時間または会計タグが付けられた次元があり、会計次元に少なくとも 1 つの式が設定されている場合に Essbase で実行される計算パスの数を示しています:

表 74　会計次元および時間次元の計算パス

勘定科目タグが付けられた次元	時間タグが付けられた次元	計算パス	各計算パスで、Essbase が対象とする次元:
密または疎	なし	1	すべての次元
密	密	1	すべての次元
密	疎	2	パス 1: 会計次元と時間次元 パス 2: その他の次元
疎	疎	2	パス 1: 会計次元と時間次元 パス 2: その他の次元
疎	密	2	パス 1: 会計次元 パス 2: その他の次元

2 パス・タグが付けられた式を使用している場合は、Essbase で追加の計算パスを実行して式を計算する必要がある場合があります。980 ページの「2 パス計算の使用」を参照してください。

計算スクリプトを使用してデータベースを計算するときに、Essbase で実行する必要がある計算パスの数は、計算スクリプトによって決まります。423 ページの「計算パス」および436 ページの「複数パス計算の理解」を参照してください。486 ページの「式と計算のグループ化」も参照してください。

コミット・アクセスに対して分離レベルが設定されていて、複数のパスが必要な場合は、Essbase によって、各パスの最後にデータ値が書き込まれます。パス間で行われるデータ取得では、中間値を取得できます。

データベースを計算すると、Essbase によって、データベース内のパスごとに、次元の計算順序が自動的に表示され、計算中に Essbase でデータベースのデータを循環使用した回数が表示されます。Essbase によって、この情報が ESSCMD のウィンドウおよびアプリケーション・ログに表示されます。

➤ アプリケーション・ログを表示する方法は、817 ページの「Essbase サーバー・ログおよびアプリケーション・ログの表示」を参照してください。

Essbase では、データ・ブロックごとに密計算と疎計算のどちらを実行するのかが決定されます。計算のタイプは、データ・ブロック内の値のタイプによって決まります。データベースでデフォルトの計算(CALC ALL)を実行すると、各ブロックは、そのブロック番号に従って順番に処理されます。

Essbase では、次の手順によりブロックが計算されます:

- 高機能計算をオンにしていて、ブロックを計算する必要がない場合(ブロックにクリーンが設定されている場合)、Essbase ではそのブロックはスキップされ、次のブロックに移動します。第 26 章「高機能計算についての理解」を参照してください。
- ブロックを再計算する必要がある場合、Essbase ではそのブロックがレベル 0、入力、上位レベル・ブロックのどれであるかが確認されます。407 ページの「データ・ブロック内のデータ・ストレージ」を参照してください。
- ブロックがレベル 0 のブロックまたは入力ブロックの場合、Essbase でブロックの密計算が実行されます。ブロックの各セルが計算されます。416 ページの「セルの計算順序」を参照してください。
- ブロックが上位レベル・ブロックの場合、Essbase では、データ・ブロックに対して値の集計または疎計算が実行されます。

  各上位レベル・ブロックの疎メンバーの組合せには、少なくとも 1 つの親メンバーがあります。Essbase では、親メンバーの次元に基づいてブロックが集計または計算されます。たとえば、上位レベル・ブロックが Sample.Basic データベースの Product -> Florida に関するものである場合、Essbase では、製品次元が選択されます。

  ブロックの疎メンバーの組合せに複数の親メンバーがある場合、Essbase では、親メンバーを含み、計算順序の最後にある次元が選択されます。たとえば、ブロックが Product -> East に関するもので、Sample.Basic データベースでデフォルトの計算を実行した場合、Essbase では、East を含んでいる市場次元が選択されます。市場次元は、データベース・アウトラインで製品次元の後に配置されているので、デフォルトの計算順序では最後になります。409 ページの「メンバーの計算順序」を参照してください。

  選択された疎次元に基づいて、Essbase では、データ・ブロックに対して次のように値の集計または疎計算が実行されます:

  - 選択された疎次元のデータ・ブロック・メンバーに式が適用される場合は、Essbase で、疎次元に対して式の計算が実行されます。Essbase により、データ・ブロック内の各セルが評価されます。式は、疎次元のメンバーにのみ影響するので、計算パフォーマンス全体に大きな影響はありません。

- 選択された疎次元がデフォルトの集計の場合、Essbase では以前に計算された子のデータ・ブロックの値を取得して、値が集計されます。

共有メンバーの計算

共有メンバーとは、他のメンバーとデータ値を共有するメンバーのことです。たとえば、Sample.Basic データベースでは、Diet Cola、Diet Root Beer および Diet Cream が、2 つの親の下に集計されています。つまり、Diet とそれぞれの製品タイプ(Colas、Root Beer、Cream Soda)の下です。図 125 に示すように、親の Diet の下のメンバーは共有メンバーです。148 ページの「共有メンバーの理解」を参照してください。

図 125　共有メンバーの計算

共有メンバーの計算は、実際のメンバーに対する計算です。FIX コマンドを使用してデータベースのサブセットを計算するときにそのサブセットに共有メンバーが含まれる場合は、Essbase によって実際のメンバーが計算されます。

26 高機能計算についての理解

この章の内容

高機能計算の概要

デフォルトでは、Essbase では、データベースのフル計算を実行するときに、計算するデータ・ブロックを追跡します。後続の計算でデータのサブセットをロードした場合、Essbase では、計算されていないデータ・ブロック、および新しいデータが原因で再計算が必要な計算済ブロックのみを計算します。このプロセスを高機能計算と呼びます。

デフォルトでは、高機能計算はオンです。このデフォルト設定は essbase.cfg で変更できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

計算スクリプトで高機能計算をオンまたはオフにすることもできます。430 ページの「高機能計算のオンとオフ」を参照してください。

その他の計算の最適化方法は、次を参照してください:

- 第 15 章「パーティション・アプリケーションの設計」
- 第 27 章「データ値の動的計算」
- 第 58 章「計算の最適化」

高機能計算の利点

高機能計算は、次のタイプの計算において計算パフォーマンスを大きく向上させるよう設計されています:

- データベースのフル計算(CALC ALL)、ただし、例外があります。

  429 ページの「高機能計算の制限」を参照してください。
- 1 つの CALC DIM ステートメントですべてのメンバーを計算する計算スクリプト。
- フル計算に高機能計算を使用できないデータベースの計算においても、計算の一部に高機能計算を使用できることがあります。

たとえば、デフォルトの集計を実行してからデータを割り当てることによってデータベースを計算する場合の計算パフォーマンスを大きく向上させるには、デフォルトの集計では高機能計算を使用可能にし、割当てでは高機能計算を使用不可にします。

高機能計算がオンになっている(デフォルト)場合、計算スクリプトを作成し、次の手順で部分的な高機能計算を行います:

- 高機能計算を使用可能にします(使用不可になっている場合)
- CALC ALL を使用してデータベースを計算します
- SET UPDATECALC コマンドを使用して高機能計算を使用不可にします
- データを割り当てます
- 必要に応じて、高機能計算を再度使用可能にします

高機能計算とデータ・ブロックのステータス

高機能計算を実行するために、Essbase ではデータベースのデータ・ブロックのステータスを確認します。データ・ブロックには、クリーンとダーティという算出ステータスがあります。Essbase では、計算後はデータ・ブロックがクリーンとしてマークされます。

高機能計算が使用可能な場合、Essbase ではダーティ・ブロックとそれに依存する親のみが計算されます。高機能計算が使用不可の場合、データ・ブロックがクリーンとダーティのどちらとしてマークされているかに関係なく、Essbase ではすべてのデータ・ブロックが計算されます。

ブロックをクリーンとしてマーク

Essbase では、次のタイプの計算時にデータ・ブロックがクリーンとしてマークされます:

- データベースのフル計算(CALC ALL、デフォルトの計算)。
- 1 つの CALC DIM ステートメントですべての次元を計算する計算スクリプト。

  たとえば、次の計算スクリプトでは、Sample.Basic データベースのすべてのメンバーが計算されます:

```
CALC DIM(Measures, Product, Market, Year, Scenario);
```

  これを、2 つの CALC DIM ステートメントですべてのメンバーを計算する計算スクリプトと比較してみます:

```
CALC DIM(Measures, Product);
CALC DIM(Market, Year, Scenario);
```

2 つの CALC DIM ステートメントを使用すると、Essbase では、データベースで少なくとも 2 つの計算パスが実行されます。この計算で、デフォルトではデータ・ブロックは Essbase によってクリーンとしてマークされません。高機能計算は正確

なクリーン・ステータスとダーティ・ステータスに依存するので、これらのマーカーを慎重に管理する必要があります。429 ページの「クリーン・ステータスとダーティ・ステータスの管理」を参照してください。

Essbase では、前述の状況でのみ、計算済のデータ・ブロックがクリーンとしてマークされます。ただし、計算スクリプトで SET CLEARUPDATESTATUS コマンドを使用している場合を除きます。432 ページの「SET CLEARUPDATESTATUS コマンドの使用」を参照してください。

ブロックをダーティとしてマーク

Essbase では、次の場合にデータ・ブロックがダーティとしてマークされます:

- データベースの一部のデータ・ブロックを計算した場合。ただし、SET CLEARUPDATESTATUS AFTER が、その計算スクリプト内の部分的計算ステートメントに含まれていない場合のみ
- データをデータ・ブロックにロードした場合
- 密次元へのメンバーの追加などにより、データベースを再構築した場合
- DATACOPY などを使用して、データをデータ・ブロックにコピーした場合

クリーン・ステータスとダーティ・ステータスの管理

データベースのサブセットを計算するとき、またはデータベースで複数の計算パスを実行するときに高機能計算を使用する場合は、Essbase でどのようにデータ・ブロックがクリーンとしてマークされるかを十分に考慮する必要があります。高機能計算を使用するときには、データ・ブロックのクリーン・ステータスとダーティ・ステータスを正確に維持して、Essbase によってデータベースをできるだけ効率的に再計算できるようにする必要があります。

たとえば、データベースのサブセットを計算した場合、新しく計算されたデータ・ブロックは、デフォルトではクリーンとしてマークされません。計算スクリプトで SET CLEARUPDATESTATUS AFTER コマンドを使用すると、この新しく計算されたブロックが確実にクリーンとしてマークされるようにできます。計算スクリプトを作成する前に、432 ページの「SET CLEARUPDATESTATUS コマンドの使用」および『Oracle Essbase テクニカル・リファレンス』を参照してください。

高機能計算の制限

高機能計算を使用するときは、次の制限と状況を考慮してください:

- 高機能計算は、データ・ブロック・レベルで機能し、セル・レベルでは機能しません。たとえば、データ値をデータ・ブロックの 1 つのセルにロードした場合、データ・ブロック全体がダーティとしてマークされます。
- データベースで 2 パスが必要な CALC ALL は、正しく計算されない可能性があります。最初のパスでクリーンとしてマークされたブロックが次のパスでスキップされるためです。この問題を回避するには、高機能計算をオフにするか、データベースに対して CALC ALL を実行するのではなく、各次元に対して CALC DIM を実行します。CALC ALL では、次の状況で、データベースでの 2 パスが必要になります:

    - 会計次元が疎の場合
    - 会計次元が密、時間次元が疎、およびアウトラインに少なくともももう 1 つの密次元がある場合
- データベース・アウトラインの式またはデータベース・アウトラインの勘定科目プロパティを変更しても、Essbase ではデータベースが再構築されません。このため、Essbase では、影響を受けたブロックがダーティとしてマークされません。該当するデータ・ブロックを再計算する必要があります。440 ページの「式または勘定科目プロパティの変更」を参照してください。
- Essbase では可能なかぎり、勘定科目タグが付けられた次元にある 2 パス・タグが付けられた式が、データベースの主計算の一部として計算されます。ただし、場合によっては、一部の式は計算スクリプトを使用して 2 回計算する必要があります。計算スクリプトを使用するときには、高機能計算を使用不可にしてから式を再計算します。
- 計算スクリプトで SET CREATENONMISSINGBLK がオンに設定されている場合、高機能計算はオフになり、影響を受けるブロックは、クリーンまたはダーティのいずれにマークされているかにかかわらず計算されます。

高機能計算の使用

この項では、高機能計算のオンとオフについて、および様々なタイプの計算での高機能計算の使用について説明します。

高機能計算のオンとオフ

デフォルトでは、高機能計算はオンです。デフォルトを変更するには、`essbase.cfg` ファイルの UPDATECALC 設定を使用します。

計算スクリプトの実行中に高機能計算をオンまたはオフにするには、計算スクリプトで SET UPDATECALC コマンドを使用します。

『Oracle Essbase テクニカル・リファレンス』を参照してください。

デフォルトのフル計算に対する高機能計算の使用

データベースのフル計算(CALC ALL)を実行するときに高機能計算を使用すると、パフォーマンスで大きなメリットが得られます。フル計算を実行する場合は、高機能計算をオン(デフォルト)のままにして、パフォーマンス上のメリットを生かしてください。

➤ 現在の計算の設定を確認するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「デフォルト計算の設定」を参照してください。

注意 高機能計算を使用するときには、429 ページの「高機能計算の制限」に記載されている情報を確認してください。

初回の計算

データベースのフル計算を初めて実行するときには、Essbase ですべてのブロックが計算されます。パフォーマンスは、高機能計算をオンにした場合もオフにした場合も変わりません。

再計算

高機能計算をオンにして、データベースの完全再計算を実行すると、Essbase では、各ブロックがクリーンとダーティのどちらにマークされているかが確認されます。428 ページの「高機能計算とデータ・ブロックのステータス」を参照してください。

データ・ブロックの確認には、5%から 10%のパフォーマンス・オーバーヘッドがかかります。これは、高機能計算を使用可能にしたときに得られるパフォーマンスに比べるとわずかな影響です。

ただし、全体の約 80%を超える値が変更されたデータベースを再計算する場合は、高機能計算のオーバーヘッドが、改善されるパフォーマンスを上回るおそれがあります。このような場合は、高機能計算を使用不可にします。

部分的な計算を行う計算スクリプトでの高機能計算の使用

Essbase では、フル計算(CALC ALL)でデータ・ブロックが計算されたとき、または 1 つの CALC DIM コマンドですべての次元が計算されたときに、データ・ブロックがクリーンとしてマークされます。428 ページの「ブロックをクリーンとしてマーク」を参照してください。

その他の計算では、Essbase では、計算済のデータ・ブロックはクリーンとしてマークされません。ただし、計算スクリプトで SET CLEARUPDATESTATUS コマンドを使用する場合を除きます。たとえば、データベースのサブセットを計算する場合、または 2 つの計算パスでデータベースを計算する場合、Essbase では、SET CLEARUPDATESTATUS コマンドを使用しないかぎり、計算済のブロックはクリーンとしてマークされません。

次の計算スクリプトを使用しても、Essbase では計算済のデータ・ブロックはクリーンとしてマークされません:

```
    FIX(“New York”)
  CALC DIM(Product, Measures);
ENDFIX
```

```
    CALC DIM(Measures, Product);
CALC DIM(Market, Year, Scenario);
```

不要な再計算を避けるために、SET CLEARUPDATESTATUS を使用してください。

SET CLEARUPDATESTATUS コマンドの使用

データベースのサブセットを計算する場合や、2 つの計算パスでデータベースを計算する場合などに、Essbase で、計算済のブロックがクリーンとしてマークされないことがあります。高機能計算で使用するためにデータ・ブロックを手動でクリーンとしてマークするには、計算スクリプトで SET CLEARUPDATESTATUS コマンドを使用します。この項および 428 ページの「高機能計算とデータ・ブロックのステータス」を参照してください。

SET CLEARUPDATESTATUS の理解

SET CLEARUPDATESTATUS コマンドには、AFTER、ONLY および OFF の 3 つのパラメータがあります。

- SET CLEARUPDATESTATUS AFTER;

  Essbase によって、データベースのサブセットを計算した場合でも、計算済のデータ・ブロックはクリーンとしてマークされます。

- SET CLEARUPDATESTATUS ONLY;

  Essbase によって、指定したデータ・ブロックはクリーンとしてマークされますが、そのデータ・ブロックは計算されません。このパラメータでは、AFTER と同じ結果が得られますが、計算は実行されません。

- SET CLEARUPDATESTATUS OFF;

  Essbase によって、データ・ブロックは計算されますが、計算済のデータ・ブロックはクリーンとしてマークされません。データベースのフル計算(CALC ALL)を実行した場合でも、クリーンとしてマークされません。計算済データ・ブロックの既存のクリーン・ステータスまたはダーティ・ステータスは変化しません。

SET CLEARUPDATESTATUS 設定の選択

SET CLEARUPDATESTATUS コマンドを使用して計算済のデータ・ブロックをクリーンとマークする際には、パラメータ(AFTER、ONLY、OFF)を選択する前に次の推奨事項を考慮してください:

- クリーンとしてマークされるのは、計算済のデータ・ブロックのみです。
- 同時計算で SET CLEARUPDATESTATUS AFTER コマンドを使用しないでください。ただし、同時計算で同じデータ・ブロックを計算する必要がないことが確実な場合を除きます。高機能計算が使用可能な場合に、同時計算で同じデータ・ブロックを計算しようとすると、Essbase では、データ・ブロックが他の同時計算によってすでにクリーンとしてマークされている場合に、そのデータ・ブロックを再計算しません。436 ページの「同時計算の処理」を参照してください。
- Essbase では、データベースの最初の計算パスでデータ・ブロックを計算したとき、そのデータ・ブロックがクリーンとしてマークされます。高機能計算を使用可能にして、次のパスで同じデータ・ブロックを計算しようとすると、

そのデータ・ブロックはすでにクリーンとしてマークされているので、Essbase では再計算されません。

SET CLEARUPDATESTATUS の使用例の確認

Sample.Basic データベースを使用した、次のようなシナリオを考えてみます:

- 疎次元は市場と製品です。
- New York は、市場疎次元のメンバーです。
- 高機能計算はオンになっています(デフォルト)。

次の例は、SET CLEARUPDATESTATUS の異なる使用方法を示しています:

例 1: CLEARUPDATESTATUS AFTER

```
   SET CLEARUPDATESTATUS AFTER;
FIX(“New York”)
  CALC DIM(Product);
ENDFIX
```

この例では、Essbase で、New York のダーティな親データ・ブロック(たとえば、New York -> Colas。Colas は製品次元の親メンバー)が検索されます。このダーティ・ブロックが計算され、クリーンとしてマークされます。Essbase では、レベル 0 のデータ・ブロックは計算されないので、このデータ・ブロックはクリーンとしてマークされません。レベル 0 のブロックの詳細は、第 25 章「計算順序の定義」を参照してください。

例 2: CLEARUPDATESTATUS ONLY

```
   SET CLEARUPDATESTATUS ONLY;
FIX(“New York”)
  CALC DIM(Product);
ENDFIX
```

Essbase で、New York のダーティな親データ・ブロック(たとえば、New York -> Colas。Colas は製品次元の親メンバー)が検索されます。ダーティな親データ・ブロックは Essbase でクリーンとしてマークされますが、計算されません。Essbase では、レベル 0 のデータ・ブロックは計算されないので、そのデータ・ブロックはクリーンとしてマークされません。たとえば、New York -> 100-10 (レベル 0 のブロック)がダーティの場合、このブロックはダーティのままです。

例 3: CLEARUPDATESTATUS OFF

```
   SET CLEARUPDATESTATUS OFF;
CALC ALL;
```

```
CALC TWOPASS;
SET CLEARUPDATESTATUS ONLY;
CALC ALL;
```

この例では、Essbase で最初にデータベースのすべてのダーティ・データ・ブロックが計算されます。計算済のデータ・ブロックはダーティのままです。Essbase でこのデータ・ブロックがクリーンとしてマークされることはありません。

Essbase では次に、勘定科目タグが付けられた次元内で、2 パス・タグが付けられたメンバーが計算されます。データ・ブロックはダーティとしてマークされたままなので、Essbase でそのデータ・ブロックが再計算されます。ここでも計算済データ・ブロックはクリーンとしてマークされません。

次に、Essbase で、データベースのすべてのダーティ・ブロックが検索され、それらのデータ・ブロックがクリーンとしてマークされます。CALC ALL コマンドが使用されていますが、ブロックは計算されません。

データ・ブロックの計算

Essbase では、疎次元メンバーの一意の組合せごとに、データ・ブロックが作成されます。ただし、その組合せに少なくとも 1 つのデータ値が存在する場合に限ります。各データ・ブロックは、疎次元メンバーの一意の組合せに対する、すべての密次元のメンバー値を表します。

たとえば、Sample.Basic データベースでは、市場次元と製品次元が疎です。したがって、データ・ブロック New York -> Colas は、疎の組合せ New York -> Colas に対する、年、メジャーおよびシナリオ次元のすべてのメンバー値を表します。

以降の項では、上位レベル、レベル 0 および入力のデータ・ブロックという概念を読者が理解していることを前提としています。407 ページの「データ・ブロック内のデータ・ストレージ」を参照してください。

密次元の計算

密次元の計算で FIX コマンドを使用していない場合、Essbase では、データベース内のすべてのデータ・ブロックで少なくとも一部のデータ値が計算されます。

たとえば、次の計算スクリプトは Sample.Basic データベースに基づいています:

```
   SET CLEARUPDATESTATUS AFTER;
CALC DIM(Year);
```

このスクリプトでは、密次元である年次元を計算します。年は密なので、データベースのすべてのデータ・ブロックに年次元のメンバーが含まれています。このため、Essbase では、すべてのデータ・ブロックのデータ値が計算されます。スクリプトで SET CLEARUPDATESTATUS AFTER コマンドが使用されているので、Essbase ではすべてのデータ・ブロックがクリーンとしてマークされます。

疎次元の計算

疎次元を計算するときに、Essbase で、データベース内のすべてのデータ・ブロックを計算する必要があるとは限らない場合があります。

たとえば、次の計算スクリプトは Sample.Basic データベースに基づいています:

```
   SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product);
```

このスクリプトでは、疎次元である製品次元を計算します。製品は疎なので、製品次元のメンバーごとにデータ・ブロックが存在します。たとえば、New York -> Colas のデータ・ブロックや、New York -> 100-10 のデータ・ブロックが存在します。

レベル 0 の影響

データ・ブロック New York -> 100-10 はレベル 0 のブロックです。このデータ・ブロックは、どちらの疎次元(市場または製品)の親メンバーも表していません。New York -> 100-10 のデータ値は入力値です。このデータ値はデータベースにロードされます。したがって、Essbase ではこのデータ・ブロックを計算する必要がありません。また、スクリプトで SET CLEARUPDATESTATUS AFTER コマンドが使用されている場合でも、New York -> 100-10 のデータ・ブロックは Essbase によってクリーンとしてマークされません。

注： Essbase では、レベル 0 のデータ・ブロックは、そのデータ・ブロックに適用される式が対応する疎のレベル 0 メンバーにある場合に計算されます。

データをデータベースにロードする場合、データをロードするレベル 0 のデータ・ブロックはダーティとしてマークされます。その後、疎次元のみを計算すると、Essbase ではレベル 0 のブロックが計算されないので、そのブロックはダーティのままです。したがって、疎次元のみを再計算すると、Essbase ではすべての上位レベルのデータ・ブロックが計算されます。これは、上位レベルのブロックがもともとクリーンの場合でも、その子ブロックがダーティになると上位レベルのブロックはダーティとしてマークされるためです。

上位レベルの影響

Colas は製品次元の親レベルのメンバーです。Essbase では、Colas の値を計算する必要があるので、このデータ・ブロックが Essbase によって計算されます。スクリプトで SET CLEARUPDATESTATUS AFTER コマンドが使用されているので、データ・ブロックは Essbase でクリーンとしてマークされます。

Essbase で疎次元を計算するときに、上位レベルのデータ・ブロックが 1 つ以上のダーティな子ブロックに依存している場合はその上位レベルのデータ・ブロックが再計算されます。

不要な計算

少なくとも 1 つの密次元を計算することによって、不要な計算を避けることができます。FIX コマンドを使用せずに、密次元を計算すると、レベル 0 のブロックを含むすべてのデータ・ブロックでデータ値が計算されます。これにより、レベル 0 のブロックはクリーンとしてマークされます。

同時計算の処理

高機能計算がオンの場合に同時計算で同じデータ・ブロックを計算しようとすると、データ・ブロックはすでにクリーンとしてマークされているので、Essbase でそのデータ・ブロックが再計算されないことがあります。

Sample.Basic データベースを使用した次の例では、Actual と Budget がシナリオ密次元のメンバーです。シナリオは密なので、データベースの各データ・ブロックに Actual 値と Budget 値が含まれます。ユーザー 1 が次の計算スクリプトを実行した場合、Essbase で、New York を表すすべてのデータ・ブロックの Actual 値が計算されます。計算済のデータ・ブロックは Essbase でクリーンとしてマークされます。ただし、各計算済ブロックのすべてのデータ値が計算されたわけではありません。たとえば、Budget 値は計算されていません。

```
   SET CLEARUPDATESTATUS AFTER;
FIX(“New York”, Actual)
 CALC DIM(Product, Year);
ENDFIX
```

ユーザー 2 が New York の Budget 値を計算するために次の計算スクリプトを実行した場合、指定したデータ・ブロックはすでにクリーンとしてマークされているので、Essbase ではそのデータ・ブロックは再計算されません。このため、正しい Budget の計算結果が得られません。

```
   SET CLEARUPDATESTATUS AFTER;
FIX(“New York”, Budget)
 CALC DIM(Product, Year);
ENDFIX
```

この問題を解決する方法の 1 つは、シナリオ次元を疎にすることです。そうすることで、Actual 値と Budget 値は、New York -> Colas -> Actual と New York -> Colas -> Budget のように異なるデータ・ブロックに配置されます。この場合、2 番目の計算スクリプトによって、Budget データ・ブロックは正しく計算されます。

同時計算の実行には、データ・キャッシュの増加が必要になる場合があります。919 ページの「データ・キャッシュ・サイズの設定」を参照してください。

複数パス計算の理解

Essbase では、可能なかぎり 1 つの計算パスでデータベースが計算されます。423 ページの「計算パス」を参照してください。

計算スクリプトを使用してデータベースを計算するときに、Essbase で実行する計算パスの数は、計算スクリプトによって決まります。428 ページの「高機能計算とデータ・ブロックのステータス」および486 ページの「式と計算のグループ化」を参照してください。

たとえば、Essbase で、データベースの最初の計算パスでデータ・ブロックが計算され、クリーンとしてマークされるとします。高機能計算を使用可能にして、次のパスで同じデータ・ブロックを計算しようとすると、そのデータ・ブロックはすでにクリーンとしてマークされているので、Essbase では再計算されません。

複数パス計算の例と解決策の確認

以降の例では、正しくない計算結果が生成される状況、および正しい結果を得るための解決策について説明します。これらの例では、Sample.Basic データベースを使用しており、高機能計算がオンであると想定しています。

例 1: 高機能計算と 2 パス

この計算スクリプトでは、デフォルト計算が実行された後、2 パス計算が実行されます:

```
   CALC ALL;
CALC TWOPASS;
```

エラー

Essbase で、データベースのダーティ・データ・ブロックが計算され、すべてのデータ・ブロックがクリーンとしてマークされます。Essbase では次に、勘定科目タグが付けられた次元に属する、2 パス・タグが付けられたメンバーを再計算する必要があります。ただし、Essbase では、指定したデータ・ブロックはすでにクリーンとしてマークされているので、そのデータ・ブロックが再計算されません。このため、正しい計算結果が得られません。

解決策

正しい結果を得るには、2 パス計算に対する高機能計算を使用不可にします。

例 2: SET CLEARUPDATESTATUS と FIX

この計算スクリプトでは、New York のデータ値を計算します。計算は、製品次元に基づいて行われます:

```
   SET CLEARUPDATESTATUS AFTER;
FIX( “New York” )
  CALC DIM(Product);
ENDFIX
CALC TWOPASS;
```

エラー

➤ Essbase では、次のプロセスを実行します:

1 Essbase で、データベースを循環して、New York を表すダーティ・データ・ブロックを計算します。計算は、製品次元に基づいて行われます。したがって、Essbase では製品次元で親メンバーを表すブロックのみ(New York -> Colas、New York -> Root Beer、New York -> Fruit Soda など)が計算され、次に製品次元の集約と式のみが計算されます。

2 SET CLEARUPDATESTATUS AFTER コマンドが使用されているので、Essbase では、計算済のデータ・ブロックがクリーンとしてマークされます。ただし、各計算済ブロックのすべてのデータ値が計算されているわけではありません。

3 Essbase で、勘定科目タグが付けられた次元に属する、2 パス・タグが付けられたメンバーを再計算する必要があります。ただし、一部のデータ・ブロックは手順 2 の計算ですでにクリーンとしてマークされています。Essbase では、クリーンとしてマークされているデータ・ブロックは再計算されません。このため、正しい計算結果が得られません。

解決策

正しい結果を得るには、2 パス計算に対する高機能計算を使用不可にします。

例 3: SET CLEARUPDATESTATUS と 2 つの CALC DIM コマンド

この計算スクリプトでは、製品次元と年次元に基づいてデータベースを計算します。2 つの CALC DIM コマンドが使用されているので、Essbase では、データベースで 2 つの計算パスを実行します:

```
   SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product);
CALC DIM(Year);
```

エラー

➤ Essbase では、次のプロセスを実行します:

1 Essbase で、データベースを循環して、ダーティ・データ・ブロックを計算します。この計算は、437 ページの「例 2: SET CLEARUPDATESTATUS と FIX」と同様、製品次元に基づいて行われます。

2 SET CLEARUPDATESTATUS AFTER コマンドが使用されているので、Essbase では、計算済のデータ・ブロックがクリーンとしてマークされます。ただし、各計算済ブロックのすべてのデータ値が計算されているわけではありません。

3 Essbase では、データ・ブロックを再計算する必要があります。再計算は、年次元に基づいて行われます。ただし、手順 2 の計算の結果、一部のデータ・ブロックはすでにクリーンとしてマークされているので、Essbase でそのデータ・ブロックは再計算されません。このため、正しい計算結果が得られません。

解決策

正しい計算結果を得るには、1 つの CALC DIM コマンドを使用して、製品次元と年次元を計算します。Essbase では、データベースで 1 つの計算パスを実行して両方の次元を計算します。次の計算スクリプトでは、正しい結果が得られます:

```
   SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product, Year);
```

注： 複数の次元を 1 つの CALC DIM コマンドで計算する場合、Essbase では、コマンドに指定した順序ではなく、デフォルトの計算順序で次元が計算されます。409 ページの「メンバーの計算順序」を参照してください。

例 4: 2 つの計算スクリプト

この例では、New York のデータ値を計算しますが、2 つの計算スクリプトを使用し、2 つの次元に基づいて計算します。最初の計算スクリプトでは、次のように製品次元を計算します:

```
   SET CLEARUPDATESTATUS AFTER;
FIX(“New York”)
  CALC DIM(Product);
ENDFIX
```

Essbase で、New York を含んでいるデータ・ブロックを計算します。計算は製品次元に基づくので、Essbase では、製品次元に親メンバーを含んでいるダーティ・ブロックのみ(New York -> Colas、New York -> Root Beer、New York -> Fruit Soda など)が計算され、製品次元の集約と式のみが計算されます。

CLEARUPDATESTATUS AFTER コマンドが使用されているので、Essbase では、計算済のデータ・ブロックがクリーンとしてマークされます。ただし、各計算済ブロックのすべてのデータ値が計算されているわけではありません。

2 番目の計算スクリプトでは、次のように年次元を計算します:

```
   SET CLEARUPDATESTATUS AFTER;
FIX(“New York”)
  CALC DIM(Year);
ENDFIX
```

Essbase で、New York を表すデータ・ブロックを計算します。計算は密次元である年次元に基づくので、Essbase では New York を含んでいるすべてのデータ・ブロックを計算する必要があります。ただし、各ブロック内では、年次元の集約と式のみが Essbase で計算されます。

エラー

最初の計算の結果、New York の一部のデータ・ブロックはクリーンとしてマークされます。Essbase では、クリーンとしてマークされたデータ・ブロックは 2 番目

の計算スクリプトで再計算されません。このため、正しい計算結果が得られません。

解決策

正しい結果を得るには、Essbase で計算済のデータ・ブロックがクリーンとしてマークされないように指定します。次の計算スクリプトでは、正しい結果が得られます:

```
   SET CLEARUPDATESTATUS OFF;
FIX(“New York”)
 CALC DIM(Product);
ENDFIX
SET CLEARUPDATESTATUS AFTER;
FIX(“New York”)
 CALC DIM(Year);
ENDFIX
```

SET CLEARUPDATESTATUS OFF コマンドを使用すると、SET CLEARUPDATESTATUS AFTER コマンドの場合と違い、Essbase でダーティ・データ・ブロックが計算されますが、そのデータ・ブロックはクリーンとしてマークされません。

この解決策では、前回のデータベースの部分的な計算で、データ・ブロックがクリーンとしてマークされていないことが前提になります。

高機能計算を使用不可にすると、データ・ブロックのステータスに関係なく、すべてのデータ・ブロックを確実に計算できます。次の計算スクリプトでは、データ・ブロックのステータス(クリーンまたはダーティ)に関係なく、指定したすべてのデータ・ブロックが計算されます:

```
   SET UPDATECALC OFF;
FIX(“New York”)
 CALC DIM(Year, Product);
ENDFIX
```

SET CLEARUPDATESTATUS AFTER コマンドを使用していないので、計算済のデータ・ブロックは Essbase でクリーンとしてマークされません。

高機能計算による影響の理解

高機能計算を使用すると、データベースの管理方法に影響を与えることがあります。この項では、アクションごとの影響について説明します。

式または勘定科目プロパティの変更

データベース・アウトラインの式を変更しても、データベース・アウトラインの勘定科目プロパティを変更しても、Essbase によってデータベースは再構築されないので、この変更の影響を受けるデータ・ブロックはダーティとしてマークされません。たとえば、勘定科目タグが付けられた次元でタイム・バランス・タグを

変更した場合、Essbase によってデータベースは再構築されず、影響を受けるブロックはダーティとしてマークされません。

その後、高機能計算をオンにした状態でデフォルトの計算を実行しても、変更は計算されません。適切なデータ・ブロックを再計算するには、計算スクリプトを使用して次のいずれかのタスクを実行します:

- 高機能計算を使用不可にして、変更されたメンバー式を計算します。
- 高機能計算を使用不可にし、FIX コマンドを使用してデータベースの該当するサブセットを計算します。
- 高機能計算を使用不可にして、データベースに対してデフォルトの CALC ALL を実行します。

関係関数と財務関数の使用

疎次元または密次元の式で関係関数(@PRIOR、@NEXT など)または財務関数(@ACCUM、@NPV、@INTEREST など)を使用する場合、Essbase では、その式を含んでいるデータ・ブロックが常に再計算されます。

『Oracle Essbase テクニカル・リファレンス』を参照してください。

データベースの再構築

メンバーを密次元に追加する場合など、データベースを再構築するときは、すべてのデータ・ブロックの再計算が必要になる可能性があります。このため、Essbase では、すべてのデータ・ブロックがダーティとしてマークされます。再構築されたデータベースを計算すると、すべてのブロックが計算されます。

注： データベース・アウトラインの式またはデータベース・アウトラインの勘定科目プロパティを変更しても、Essbase によってデータベースは再構築されません。該当するデータ・ブロックを再計算する必要があります。440 ページの「式または勘定科目プロパティの変更」を参照してください。

データのコピーと消去

DATACOPY コマンドを使用して値をデータ・ブロックにコピーすると、コピー先のデータ・ブロックはダーティとしてマークされます。Essbase では、データベースの再計算時にこのブロックが計算されます。

CLEARDATA コマンドと CLEARBLOCK コマンドを使用してデータ値を消去すると、Essbase では、ブロックがどのようにマークされていたかに関係なく、すべてのブロックが消去されます。

通貨換算

CCONV コマンドを使用して通貨を換算すると、換算後のデータ・ブロックはダーティとしてマークされます。Essbase では、データベースの再計算時に、すべての換算済のブロックが計算されます。

27 データ値の動的計算

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

動的計算の理解

データベースの全体的な計算を設計するとき、データベースのバッチ計算時にメンバーの組合せを事前に計算するのではなく、データの取得時に一部のメンバーの組合せを計算する方が効率的な場合があります。データベースで一部の値を動的に計算することで、データベースの全体的な計算のパフォーマンスが大幅に向上します。

Essbase では、メンバーに動的計算を定義できます。この定義は、ユーザーの要求時に Essbase でメンバーのデータ値を計算するように指定します。動的計算によって、データベースのバッチ計算時間は短縮されますが、動的に計算されるデータ値の取得にかかる時間が増加する可能性があります。455 ページの「取得時間に対する影響の軽減」を参照してください。

Essbase では、メンバーごとに動的計算を指定します。データベース・アウトラインのメンバーは、次の 2 つのタイプのうち、いずれかの動的計算済メンバーとして定義できます:

- 動的計算
- 動的計算および保管

動的計算メンバーの理解

Essbase では、データベースのバッチ計算時(CALC ALL の実行時など)に、動的計算タグが付けられたメンバーのデータ値は計算されません。かわりに Essbase では、データの取得時(Smart View にデータを取得するときなど)にデータ値が計算されます。

具体的には、次の 2 つのうち、いずれかの方法でデータ値を要求したときに、Essbase でデータ値が動的に計算されます:

- データ値を Smart View に取得する方法
- データ値を表示するレポート・スクリプトを実行する方法

Essbase では、計算済の値は保管されず、取得ごとに値が再計算されます。

動的計算および保管メンバーの理解

Essbase では、動的計算メンバーの場合と同じ方法で、データの取得時に動的計算および保管タグが付けられたメンバーのデータ値が計算されます。ただし、動的計算および保管メンバーの場合、Essbase では動的に計算されたデータ値が保管されます。その後同じデータ値を再び取得するときに、再計算は必要ありません。ただし、Essbase で値の再計算が必要であることが検出された場合を除きます。

データの再計算

Essbase で、動的計算および保管メンバーのデータ値の再計算が必要であることが検出されると、その値を含んでいるデータ・ブロックにインディケータが設定されます。こうすることで、Essbase では、データ値の次の取得時にそのブロックを再計算する必要があることを認識できます。

Essbase では、データ値自体ではなく、その値を含んでいるデータ・ブロックにインディケータが設定されます。つまり、Essbase では、動的計算および保管メンバーをデータ・ブロック・レベルで追跡します。70 ページの「データ・ブロックおよびインデックス・システム」を参照してください。

データ・ブロックを再計算する必要がある場合は、Essbase でその必要性が検出され、次のいずれかの状況が発生したときにデータ・ブロックにインディケータが設定されます:

- バッチ計算を行うとき。
- データベースを再構築するとき。
- CLEARBLOCK DYNAMIC 計算コマンドを使用するとき。

  『Oracle Essbase テクニカル・リファレンス』を参照してください。

次にデータ値を取得するときに、Essbase で、インディケータが設定されたデータ・ブロックが再計算されます。

再計算で更新された値による影響

データの更新時には、Essbase でデータ・ブロックの再計算の必要性が検出されず、データ・ブロックにインディケータは設定されません。そのため、更新されたブロックは次のバッチ計算時にのみ再計算されます。次のシナリオについて考えてみます:

- データ・ロードの実行。
- スプレッドシートからデータ送信を実行します。

動的計算および保管メンバーの子にデータをロードし、そのメンバーが子メンバーの集計である場合、Essbase では、次の取得時にその動的計算および保管メンバーに再計算が必要であることが認識されません。親メンバーは、次のバッチ計算時にのみ再計算されます。

データのロード後に、データベースのバッチ計算を実行するか、CLEARBLOCK DYNAMIC 計算コマンドを使用して、動的計算および保管メンバーを再計算する必要があります。『Oracle Essbase テクニカル・リファレンス』を参照してください。

動的に計算される子の値に対する親の値の取得

動的計算の子メンバーまたは動的計算および保管の子メンバーから計算される親の値を取得する場合、Essbase では、親の値を計算する前に子メンバーの組合せを動的に計算する必要があります。Essbase では、子が動的計算および保管メンバーである場合でも、子の値は保管されません。

たとえば、Market が親メンバーで、East と West が動的計算および保管の子メンバーで、Market に集計されるとします。Market のデータ値を取得するときは、East と West の値を個別に取得していない場合でも、Essbase ではその値が計算されます。ただし、Essbase では East と West の値は保管されません。

動的計算による利点

データベースの一部の値を動的に計算できれば、データベースの全体的な計算のパフォーマンスが大幅に向上します。

一部のデータ値を動的に計算することで、次の項目を減らすことができます:

- データベースのバッチ計算時間。Essbase で計算するメンバーの組合せが少なくなるためです。
- ディスク使用量。Essbase に保管する計算済データ値が少なくなるためです。データベース・サイズとインデックス・サイズも小さくなります。
- データベースの再構築時間。たとえば、密次元の動的計算メンバーを追加または削除しても、データ・ブロック・サイズは変化しません。このため、Essbase ではデータベースを再構築する必要がありません。459 ページの「データベースの再構築」を参照してください。
- データベースのバックアップに必要な時間。データベース・サイズが縮小されるので、Essbase でバックアップを実行する時間が短くなります。

Essbase で動的に計算されるデータ値は、取得に時間がかかる可能性があります。動的に計算されるメンバーの取得時間を推測できます。455 ページの「取得時間に対する影響の軽減」を参照してください。

動的計算の使用

どのメンバーも動的計算または動的計算および保管としてタグ付けできます。ただし、次のメンバーは除きます:

- 式が設定されていないレベル 0 メンバー
- ラベルのみメンバー
- 共有メンバー

動的計算の対象とするメンバーは、データベース構造、および(1)計算時間の短縮やディスク使用量の削減のニーズと、(2)ユーザー向けの高速なデータ取得のニーズとのバランスによって変わります。446 ページの「動的に計算する値の選択」を参照してください。

アウトライン・エディタで、動的計算メンバーと動的計算および保管メンバーを確認できます。図 126 は動的計算メンバーを示しています。

図 126　動的計算メンバーを示している Sample.Basic アウトライン

Smart View で、ユーザーはビジュアル・キューを表示して動的計算値を区別できます。Oracle Hyperion Smart View for Office User's Guide を参照してください。

動的計算値を含むスプレッドシートを作成するときに、スプレッドシート・デザイナは、「データなし操作」オプションを使用すれば、テスト用のスプレッドシートの作成中に、Essbase によって値の動的計算と保管が行われないようにできます。

動的に計算する値の選択

一部のデータ値を動的に計算すれば、計算時間の短縮、ディスク使用量の低減およびデータベース再構築時間の短縮が実現されますが、動的に計算されるデータ値の取得時間は長くなります。

動的に計算するメンバーを決定する場合は、次の項で説明するガイドラインを使用してください。

密のメンバーと動的計算

密次元のメンバーに、次の変更を加えることを考えます:

- 密次元の上位レベル・メンバーに動的計算タグを付けます。

- 単純な式を設定したレベル 0 の密次元メンバーに動的計算タグを付けて、取得時間の増加を評価します。

  Essbase では、単純な式にコストのかかる計算は必要ありません。財務関数または次元間演算子(->)を含んでいる式は複雑な式です。
- 密次元のメンバーには、動的計算および保管タグを付けないでください。

疎のメンバーと動的計算

疎次元メンバーに、次の変更を加えることを考えます:

- 6 個以下の子を持つ疎次元の上位レベル・メンバーの一部に、動的計算タグまたは動的計算および保管タグを付けます。
- 複雑な式を設定した疎次元のメンバーに動的計算タグまたは動的計算および保管タグを付けます。

  Essbase では、複雑な式にはコストのかかる計算が必要です。たとえば、財務関数を含んでいる式は複雑な式です。971 ページの「複雑な式の使用」を参照してください。
- 頻繁に再構築する次元の上位レベル・メンバーに動的計算タグまたは動的計算および保管タグを付けます。
- 20 個以上の子孫を持つ疎次元の上位レベル・メンバーには動的計算タグと動的計算および保管タグを付けないでください。

449 ページの「「動的計算」または「動的計算および保管」の選択」を参照してください。

2 パス・メンバーと動的計算

バッチ計算の実行に必要な時間を短縮するには、2 パス・メンバーに動的計算タグを付けます。会計次元に属していないメンバーを含む、すべての動的計算メンバーまたは動的計算および保管メンバーに 2 パス・タグを付けることができます。980 ページの「2 パス計算の使用」を参照してください。

2 パス・タグが付けられたメンバーと属性メンバーの相互作用については、167 ページの「属性次元と標準次元の比較」を参照してください。

密次元の 2 パス・メンバーでのクエリーによる、動的計算キャッシュへの影響の詳細は、447 ページの「2 パス・メンバーと動的計算」を参照してください。

親子関係と動的計算

親メンバーに 1 つの子メンバーがあり、その子に動的計算タグを付ける場合は、親にも動的計算タグを付ける必要があります。同様に、子に動的計算および保管タグを付ける場合は、親にも動的計算および保管タグを付ける必要があります。ただし、親メンバーに 1 つの子メンバーがあり、親が動的計算メンバーまたは動的計算および保管メンバーの場合は、その子に動的計算タグまたは動的計算および保管タグを付ける必要はありません。

計算スクリプトと動的計算

Essbase では、計算スクリプトの CALC ALL または CALC DIM ステートメントを計算するときに、動的計算メンバーと動的計算および保管メンバーの計算をバイパスします。

同様に、メンバー・セット関数(@CHILDREN や@SIBLINGS など)を使用して、計算するメンバーのリストを指定した場合、Essbase では、作成されたリストのすべての動的計算メンバーまたは動的計算および保管メンバーの計算がバイパスされます。

計算スクリプトで明示的に動的計算メンバーまたは動的計算および保管メンバーを指定した場合、その計算スクリプトは失敗します。計算スクリプトで動的計算メンバーまたは動的計算および保管メンバーの計算は実行できません。計算スクリプトを使用してメンバーを明示的に計算する場合は、メンバーに動的計算タグを付けないでください。

たとえば、次の計算スクリプトは、Qtr1 が動的計算メンバーでない場合にのみ有効です:

```
   FIX (East, Colas)
  Qtr1;
ENDFIX
```

式と動的に計算されるメンバー

式をデータベース・アウトラインに適用するときに、動的に計算されるメンバーをその式に追加できます。たとえば、Qtr1 が動的計算済メンバーの場合、データベース・アウトラインで Qtr1 に次の式を設定できます:

```
   Qtr1 = Jan + Feb;
```

動的に計算されるメンバーを、計算スクリプトで式計算の対象にすることはできません。Essbase では、動的に計算される値に対してメモリーが確保されないため、値をそのメンバーに割り当てることができません。たとえば、Qtr1 が動的計算済メンバーまたは動的計算および保管メンバーの場合に計算スクリプトに次の式を追加すると、Essbase によって構文エラーが表示されます:

```
   Qtr1 = Jan + Feb;
```

Qtr1 が動的計算メンバーまたは動的計算および保管メンバーで、年が動的計算でも動的計算および保管でもない場合は、計算スクリプトで次の式を使用できます:

```
   Year = Qtr1 + Qtr2;
```

Essbase では、動的に計算されるメンバーに値を割り当てないので、この式は有効です。

注： データベース・アウトラインまたは計算スクリプトの式で、動的に計算されるメンバーを参照すると、Essbase では、動的計算を実行するために通常の計算が中断されます。この中断によって、計算パフォーマンスが大幅に低下する可能性があります。

スクリプトと動的に計算されるメンバー

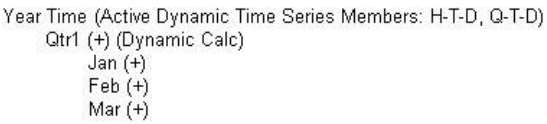
計算スクリプトの前処理段階では、アウトラインに密の動的計算メンバーが含まれているかどうかを判断できません。スクリプトにランタイムに依存する式がある場合、Essbase では、スクリプトの実行時に密の動的計算メンバーをすべて計算する必要があります。SET FRMLRTDYNAMIC OFF 計算コマンドを使用すると、この動的計算メンバーの計算が停止されるのでパフォーマンスを向上させることができます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

動的に計算される子

メンバーの計算が動的計算の子メンバーまたは動的計算および保管の子メンバーの計算に依存する場合、Essbase では、親を計算するためにデータベースのバッチ計算時に子メンバーを最初に計算する必要があります。このため、通常の計算時間は短縮されません。この要件は、疎次元のメンバーおよび密次元のメンバーに適用されます。

たとえば、図 127 では、Qtr1 は動的計算メンバーです。その子 Jan、Feb および Mar は動的メンバーではありません。その親である年も動的メンバーではありません。Essbase では、データベースのバッチ計算時に年を計算する際、Qtr1 を含む子の値を集計する必要があります。このため、Qtr1 が動的計算メンバーでも、Qtr1 を計算するための追加の時間が必要になります。

図 127　動的計算メンバーとして Qtr1 を示している Sample.Basic アウトライン



「動的計算」または「動的計算および保管」の選択

ほとんどの場合、計算パフォーマンスを最適化し、ディスク使用量を減らすには、動的計算および保管メンバーではなく、動的計算メンバーを使用します。ただし、特定の状況では、動的計算および保管メンバーの方が適しています。

疎次元メンバーの推奨事項

ほとんどの場合、疎次元メンバーを動的に計算するときには、動的計算および保管タグではなく、動的計算タグを付けます。動的計算メンバーを含んでいるメンバーの組合せのデータ値を Essbase で計算するときに、関連するデータ・ブロック

の要求された値のみが Essbase によって計算されます。この値には、データ・ブロックのサブセットである値を使用できます。

ただし、動的計算および保管メンバーを含んでいるメンバーの組合せのデータ値を Essbase で計算するときに、要求されたデータ値がデータ・ブロックのサブセットの場合でも、Essbase ではデータ・ブロック全体を計算し、保管する必要があります。このため、計算時間が長くなり、最初の取得時間も長くなります。

Essbase では、要求されたデータ値を含んでいるデータ・ブロックのみを保管します。要求されたデータ・ブロックを計算するために中間のデータ・ブロックを計算する必要がある場合、Essbase では、その中間ブロックを保管しません。

中間のデータ・ブロックを計算すると、最初の取得時間が大幅に長くなります。たとえば、Sample.Basic データベースでは、市場と製品は疎次元です。市場と市場の子が動的計算および保管メンバーであるとします。ユーザーがメンバーの組合せ Market -> Cola -> Jan -> Actual -> Sales のデータ値を取得する場合、Essbase で、Market -> Cola データ・ブロックが計算され保管されます。Market -> Cola を計算して格納するために、Essbase では、中間のデータ・ブロック(East -> Cola、West -> Cola、South -> Cola および Central -> Cola)が計算されます。Essbase では、これらの中間のデータ・ブロックは格納されません。

特定の特徴を持つメンバーの推奨事項

動的計算および保管を使用すると、最初の取得が遅くなることがありますが、後続の取得は動的計算メンバーよりも速くなります。次のメンバーには、動的計算ではなく、動的計算および保管を使用します:

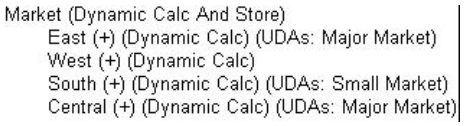
- リモート・データベースに子を持つ上位レベルの疎次元メンバー。

  Essbase では、リモート・データベースから値を取得する必要があるので、取得時間は長くなります。460 ページの「パーティション内のデータの動的計算」を参照してください。
- 複雑な式を持つ疎次元メンバー。

  Essbase では、複雑な式にはコストのかかる計算が必要です。財務関数または次元間メンバーを含んでいる式は複雑な式です。
- ユーザーが上位レベルの疎次元メンバーを頻繁に取得する場合は、取得時間が短いことが重要です。

たとえば、Sample.Basic データベースでほとんどのユーザーが市場レベルでデータを取得する場合は、市場に動的計算および保管タグを付けて、その子に動的計算タグを付けます。

図 128　Sample.Basic アウトライン、市場は動的計算および保管メンバー



密次元メンバーの推奨事項

密次元メンバーには動的計算メンバーを使用します。密次元でメンバーを動的計算および保管として定義しても、取得時間とバッチ計算時間はわずかしか短縮されません。また、Essbase では、メンバーのデータ値に対してデータ・ブロックにスペースを確保するので、データベース・サイズ(ディスク使用量)は大幅には縮小されません。

多数の同時ユーザーが使用するデータの推奨事項

同時ユーザーが使用するデータには、動的計算メンバーを使用します。多数のユーザーが同時に Essbase のデータを取得する場合、動的計算および保管メンバーの最初の取得時間は、動的計算メンバーの場合よりも大幅に長くなる可能性があります。

動的計算および保管メンバーの取得時間は、同時に取得するユーザーの数が増えるにつれて長くなります。ただし、動的計算メンバーの取得時間は、同時に取得するユーザーの数が増えても長くなりません。

多数のユーザーが同時にデータにアクセスする場合は、動的計算および保管メンバーではなく、動的計算メンバーを使用した方が取得時間が大幅に短くなる可能性があります。

動的計算による計算順序の変更の理解

動的に計算されるデータ値を使用すると、Essbase で値を計算する順序が変更され、データベースの管理方法に影響が及ぶ場合があります。

動的計算の計算順序

Essbase では、データ値を動的に計算するときに、データベースのバッチ計算順序とは異なる順序でデータを計算します。

バッチ計算の場合、Essbase では次の順序でデータベースが計算されます:

1. 勘定科目タグが付けられた次元
2. 時間タグが付けられた次元
3. その他の密次元(データベース・アウトラインに表示されている順)
4. その他の疎次元(データベース・アウトラインに表示されている順)
5. 2 パス計算

第 25 章「計算順序の定義」を参照してください。

動的計算値の場合、Essbase では取得時に、値を計算するために、次の順序でデータベースが計算されます:

1. 疎次元

- 時間タグが付けられた次元が疎で、データベース・アウトラインに時系列データが使用されている場合、Essbase では時間次元に基づいて疎の計算が行われます。
- その他の場合は、Essbase では、通常バッチ計算に使用する次元に基づいて計算が行われます。

2. 密次元
    1. 密の場合は、勘定科目タグが付けられた次元
    2. 密の場合は、時間タグが付けられた次元
    3. 時系列計算
    4. 残りの密次元
    5. 2 パス計算
    6. 属性

データの取得で属性メンバーを使用する場合、計算順序の最後の手順は属性の総計です。属性計算は、クエリーで指定された属性メンバーに一致する、即時的な集約をデータ・ブロックで実行します。クエリーに 2 パス計算メンバーが含まれる場合、属性計算は、すべての集約値が収集された後に 2 パス計算メンバー式を適用します。この 2 パス計算は、実際のデータ・ブロックの値ではなく、属性計算からデータ値を使用します。

クエリーで属性メンバーを使用すると、Essbase では動的計算でタイム・バランス・メンバーの値が無視されます。属性を使用しない取得では、タイム・バランス・メンバーの値が計算に適用されます。属性メンバーを使用する場合と使用しない場合の計算手順の違いによって、動的に計算される上位レベルの時間メンバーの結果が異なるものになります。

属性を使用しない取得では、動的に計算されるメンバーは最後の手順で計算されるので、タイム・バランスの機能が適切に適用されます。一方、属性を使用する取得では、属性の総計が最後に適用される手順です。計算順序の違いによって、動的に計算される上位レベルの時間メンバーに対して 2 つの異なる予測可能な結果が生成されます。

動的に計算する 2 パス・メンバーの計算順序

Essbase で 2 パス・タグが付けられたメンバーのデータ値を動的に計算するときに必要な計算結果が得られるように、次の情報を考慮してください(980 ページの「2 パス計算の使用」を参照)。

複数の動的計算または動的計算および保管の密次元メンバーに 2 パス・タグが付けられている場合、Essbase では、最初のパスで動的計算が実行され、その後 2 パス・メンバーが次の順序で計算されます:

1. 会計次元の 2 パス・メンバー(存在する場合)
2. 時間次元の 2 パス・メンバー(存在する場合)
3. 残りの密次元の 2 パス・メンバー(順序は、アウトラインにおける次元の順序どおり)

たとえば、Sample.Basic データベースで、次のように想定します:

- メジャー密次元(勘定科目タグが付けられた次元)の Margin%には、動的計算および 2 パス・タグが付けられています。
- シナリオ密次元の Variance には、動的計算および 2 パス・タグが付けられています。

Essbase では、勘定科目次元メンバーが先に計算されます。そのため、Essbase は、Margin% (メジャー次元から)を計算し、その次に Variance (シナリオ次元から)を計算します。

シナリオが疎次元の場合、Essbase では、動的計算の通常の計算順序に従って、最初に Variance が計算されます。次に Essbase によって Margin%が計算されます。451 ページの「動的計算の計算順序」を参照してください。

この計算順序では、Variance の式ではなく、Margin%の式を使用して Essbase で Margin% -> Variance を計算する必要があるので、必要な結果が得られません。シナリオを密次元にすると、この問題を防ぐことができます。メジャー次元(会計次元)が疎の場合、Essbase では最初に Margin%を計算するので、この問題は発生しません。

非対称型データの計算順序

動的計算に使用される計算順序は、データベースのバッチ計算に使用される計算順序と異なるので、一部のデータベース・アウトラインでは、メンバーに動的計算タグまたは動的計算および保管タグを付けた場合に異なる計算結果を得る可能性があります。この違いは、Essbase で非対称型データを動的に計算したときに発生します。

対称型データの計算では、計算される次元に関係なく、同じ結果が生成されます。

表 75 のデータ・セットを使用して、Qtr1 -> Profit を計算すると、時間タグが付けられた次元に従って計算しても、勘定科目タグが付けられた次元に従って計算しても、同じ結果になります。時間次元に従って計算すると、次のように Jan、Feb および Mar の値が加算されます:

```
50+100+150=300
```

会計次元に従って計算すると、次のように Qtr1 -> Sales から Qtr1 -> COGS が減算されます:

```
600-300=300
```

表 75　対称型計算の例

時間->勘定科目	Jan	Feb	Mar	Qtr1
Sales	100	200	300	600
COGS	50	100	150	300

時間->勘定科目	Jan	Feb	Mar	Qtr1
Profit (Sales - COGS)	50	100	150	300

非対称型データの計算では、次元ごとに異なる計算が行われます。

表 76 のデータ・セットを使用して、East -> Sales を計算すると、市場次元に従って計算した場合は正しい結果になりますが、会計次元に従って計算した場合は誤った結果になります。市場次元に従って計算すると、次のように New York、Florida および Connecticut の値が加算され、正しい結果になります:

```
50 + 100 + 100 = 250
```

会計次元に従って計算すると、次のように East -> Price の値と East -> UnitsSold の値が乗算され、誤った結果になります:

```
15 * 50 = 750
```

表 76　非対称型計算の例

市場->勘定科目	New York	Florida	Connecticut	East
UnitsSold	10	20	20	50
Price	5	5	5	15
Sales (Price * UnitsSold)	50	100	100	250

次のアウトラインでは、East が疎次元で、Accounts が密次元です:

```
East
     New York (+)
     Florida (+)
     Connecticut (+)
Accounts
     UnitsSold (~)
     Price (~)
     Sales (~) UnitsSold*Price;
```

East と Sales に動的計算タグが付けられている場合、Essbase では、East と Sales に動的計算タグが付けられていない場合とは異なる計算結果が生成されます。

East と Sales が動的計算メンバーでない場合、Essbase では次の次元が計算され、正しい結果が生成されます:

1. Accounts 密次元(New York、Florida および Connecticut の UnitsSold、Price および Sales の値を計算します)
2. East 疎次元(East の Sales 値を得るために、New York、Florida および Connecticut の UnitsSold、Price および Sales の計算値を集約します)

East と Sales が動的計算メンバーの場合、Essbase では次の次元が計算され、誤った結果が生成されます:

1. East 疎次元(East の値を得るために、New York、Florida および Connecticut の UnitsSold、Price および Sales の値を集約します)

2. East -> Sales の値(East データ・ブロック内の集約値を取得し、Sales の値を得るためにこの集約値を使用して式計算を実行します)

この問題を回避して必要な結果を確実に得るには、Sales メンバーに動的計算タグまたは動的計算および保管タグを付けないでください。

取得時間に対する影響の軽減

密次元のメンバーを動的に計算するときの取得時間の増加は、メンバーに複雑な式が含まれていないかぎり、大きくありません。取得時間は、疎次元のメンバーに動的計算タグまたは動的計算および保管タグを付けた場合に大幅に増加する可能性があります。

次の各項では、データベースに対する動的計算メンバーの影響を分析および管理する方法について説明します。

注： クエリー取得に最も重大な影響を与える関数のリストは、988 ページの「メンバー・セット関数またはパフォーマンスの選択」を参照してください。

取得係数の表示

Essbase では、データベース・アウトラインの保存時にそのアウトラインの取得係数が計算されるので、これによって取得時間の増加を推測できます。この取得係数は、Essbase で計算するのに最もコストのかかる、動的に計算されるデータ・ブロックに基づいて Essbase によって計算されます。取得係数では、集約のみを考慮しています。式の取得の影響は考慮していません。

取得係数は、最もコストのかかるブロックを計算するために、ディスクまたはデータベースから Essbase によって取得する必要があるデータ・ブロックの数です。データベースで、密次元にのみ動的計算メンバーまたは動的計算および保管メンバーが存在する(疎次元に動的計算メンバーと動的計算および保管メンバーが存在しない)場合の取得係数は、1 です。

取得係数の高い(たとえば、2000 を超える)アウトラインでは、ユーザーがデータを取得するときに大きな遅延が発生する可能性があります。ただし、取得時間への実際の影響は、ユーザーが取得する動的に計算されるデータ値の数によっても異なります。取得係数は 1 つの指標にすぎません。アプリケーションによっては、動的計算メンバーを使用することで、データベース・サイズとインデックス・サイズが小さくなり、取得時間が短縮される場合もあります。

Essbase では、アプリケーション・ログに取得係数の値が表示されます。

➤ 推定の取得係数を表示する方法は、817 ページの「Essbase サーバー・ログおよびアプリケーション・ログの表示」を参照してください。

このサンプルと同様のメッセージは取得係数を表します:

```
   [Wed Sep 20 20:04:13 2000] Local/Sample///Info (1012710)
Essbase needs to retrieve [1] Essbase kernel blocks in order
```

```
to calculate the top dynamically-calculated block.
```

このメッセージでは、Essbase で、最もコストのかかる、動的に計算されるデータ・ブロックを計算するために 1 つのブロックを取得する必要があることがわかります。

動的に計算されるメンバーの要約の表示

動的計算メンバーまたは動的計算および保管メンバーをデータベース・アウトラインに追加して、そのアウトラインを保存すると、Essbase により、動的計算タグと動的計算および保管タグが付けられたメンバーの数に関する要約が作成されます。この要約は Essbase のアプリケーション・ログに表示されます。

➤ 動的に計算されるメンバーの要約を表示する方法は、817 ページの「Essbase サーバー・ログおよびアプリケーション・ログの表示」を参照してください。

次のサンプルと同様のメッセージが表示されます:

```
   [Wed Sep 20 20:04:13 2000]Local/Sample///Info(1007125)
The number of Dynamic Calc Non-Store Members = [ 8 6 0 0 2]

[Wed Sep 20 20:04:13 2000]Local/Sample///Info(1007126)
The number of Dynamic Calc Store Members = [ 0 0 0 0 0]
```

このメッセージでは、データベース・アウトラインの最初の次元に 8 個、2 番目の次元に 6 個、5 番目の次元に 2 個の動的計算メンバーがあることがわかります。この数には動的時系列メンバーが含まれています。

この例には、動的計算および保管メンバーは含まれていません。

取得バッファ・サイズの拡張

Smart View にデータを取得するとき、またはレポート・ライターを使用してデータを取得するときに、Essbase では、取得バッファを使用して取得を最適化します。Essbase では、データがセクション単位で処理されます。取得バッファ・サイズを増やすと、Essbase で一度に処理できるデータのセクションが増えるので、取得時間を大幅に短縮できます。

デフォルトの取得バッファ・サイズは 10KB です。ただし、取得バッファ・サイズに 10KB を超えるサイズを設定すると、取得時間が短縮できます。993 ページの「取得バッファのサイズの設定」を参照してください。

➤ 取得バッファ・サイズを設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	取得バッファ・サイズの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』

ツール	トピック	場所
ESSCMD	SETDBSTATEITEM	『Oracle Essbase テクニカル・リファレンス』

動的計算キャッシュの使用

デフォルトでは、Essbase では、密次元の動的計算メンバーを、たとえばクエリーのために計算するときに、計算に必要なすべてのブロックを動的計算キャッシュと呼ばれるメモリーの領域に書き込みます。Essbase では、そのブロックを動的計算キャッシュに書き込むときに、そのブロックを拡張して、密次元のすべての動的計算メンバーを含めます。

クエリーに密次元の 2 パス計算メンバーが含まれる場合、クエリーでは取得対象のブロックに対してそれぞれ 1 つの動的計算キャッシュが必要です。

Essbase の動的計算キャッシュを使用すると、動的計算のメモリー使用率を集中制御できます。動的計算キャッシュでデータ・ブロックを管理すると、全体のメモリー・スペース要件が小さくなり、メモリー割当てを行うためにオペレーティング・システムを呼び出す回数を減らせるので、パフォーマンスを向上させることができます。

注： 動的計算キャッシュと計算キャッシュでは、計算パフォーマンスを最適化するためのアプローチが異なります。920 ページの「計算機キャッシュのサイズ設定」を参照してください。

動的計算キャッシュの使用状況の確認

Essbase では、データの取得ごとに 2 つのメッセージがアプリケーション・ログに書き込まれます。次の例で、最初のメッセージは取得にかかった合計時間を示しています:

```
   [Thu Aug 03 14:33:00 2005]Local/Sample/Basic/aspen/Info(1001065)
Regular Extractor Elapsed Time : [0.531] seconds

[Thu Aug 03 14:33:00 2005]Local/Sample/Basic/aspen/Info(1001401)
Regular Extractor Big Blocks Allocs -- Dyn.Calc.Cache : [30] non-Dyn.Calc.Cache : [0]
```

動的計算キャッシュを使用している場合は、2 つ目のメッセージに、データ計算機キャッシュ内で計算されたブロックの数(Dyn.Calc.Cache: [n])、および動的計算キャッシュの外部のメモリーで計算されたブロックの数(non-Dyn.Calc.Cache: [n])が表示されます。

動的計算キャッシュが効果的に使用されているかどうかを判断するために、両方のメッセージを確認して、essbase.cfg の設定を検討します。たとえば、ブロックが動的計算キャッシュの外部および内部で計算されたことがメッセージに示されている場合は、DYNCALCCACHEMAXSIZE 設定を増やすことができます。指定した最大サイズがサーバー上のすべての動的計算キャッシュに用意できる上限の場合、および計算機キャッシュの外部のメモリーを使用して動的に計算される取得を実行すると、スワッピングやページング・アクティビティなどが原因で許容

できない遅延が発生する場合は、DYNCALCCACHEWAITFORBLK を TRUE に設定します。

performance statistics 文法と一緒に **query database** MaxL ステートメントを使用して、動的計算キャッシュのアクティビティの要約を表示できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

標準的な手順での動的計算の使用

標準の Essbase 手順で動的計算を使用すると、次のプロセスに影響します:

- データとデータ・ブロックの消去

  CLEARBLOCK DYNAMIC コマンドを使用して、動的計算および保管メンバーの組合せのデータ・ブロックを削除できます。

  CLEARDATA コマンドを使用して、動的計算および保管データ・ブロックにマークできるので、Essbase でブロックの再計算を認識できます。CLEARDATA コマンドは、動的計算メンバーのデータ値に影響しません。

- データのコピー

  動的に計算されるデータ値にはデータをコピーできません。動的計算メンバーと動的計算および保管メンバーは、DATACOPY 計算コマンドの対象として指定できません。

- 通貨換算

  動的計算メンバーと動的計算および保管メンバーは、CCONV コマンドの対象として指定できません。

- データのロード

  データのロード時に、Essbase では、動的計算メンバーまたは動的計算および保管メンバーを含んでいるメンバーの組合せにデータをロードしません。Essbase では、データのロード時にこれらのメンバーはスキップされ、エラー・メッセージは表示されません。

  データのロード後に動的計算メンバーと動的計算および保管メンバーにデータを配置する場合は、Essbase で動的計算および保管メンバーが確実に再計算されるようにしてください。445 ページの「再計算で更新された値による影響」を参照してください。

- データのエクスポート

  Essbase では、データをエクスポートする前に動的計算値が計算されることはありません。Essbase では、動的計算メンバーの値はエクスポートされません。以前にユーザーがデータを取得した際の計算値がデータベースに存在する場合にのみ、Essbase で動的計算および保管メンバーの値がエクスポートされます。

- データのレポート

  Essbase では、動的に計算されるメンバーに SPARSE データ抽出メソッドを使用できません。SPARSE データ抽出メソッドは、報告されるデータ行の大部分が#MISSING の場合にパフォーマンスを最適化します。『Oracle Essbase テクニカル・リファレンス』の<SPARSE コマンドに関する説明を参照してください。

- 計算スクリプトに含まれる動的メンバー

  データベースの計算時に、Essbase では、すべての動的計算メンバーと動的計算および保管メンバーの計算がスキップされます。計算スクリプト内で動的計算メンバーまたは動的計算および保管メンバーのメンバー計算を実行しようとすると、Essbase でエラー・メッセージが表示されます。448 ページの「計算スクリプトと動的計算」を参照してください。

動的計算メンバーと動的計算および保管メンバーの作成

➤ アウトライン・エディタを使用して動的計算メンバーと動的計算および保管メンバーを作成する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「メンバー・ストレージ・プロパティの設定」を参照してください。

➤ 次元の構築時に、次元構築データ・ファイルで動的計算メンバーと動的計算および保管メンバーを作成するには、動的計算にプロパティ X を使用し、動的計算および保管にプロパティ V を使用します。292 ページの「データ・ソースを使用したメンバー・プロパティの操作」を参照してください。

データベースの再構築

動的計算メンバーを密次元に追加すると、Essbase では、そのメンバーの値に対してデータ・ブロックのスペースが確保されません。このため、Essbase ではデータベースを再構築する必要がありません。ただし、動的計算および保管メンバーを密次元に追加すると、Essbase で、そのメンバーの値に対して関連データ・ブロックのスペースが確保されるので、データベースを再構築する必要があります。

動的計算メンバーまたは動的計算および保管メンバーを疎次元に追加すると、Essbase によりインデックスが更新されますが、関連データ・ブロックは変更されません。837 ページの「インデックス・マネージャ」を参照してください。

Essbase では、データベースを再構築する必要がなければ、データベース・アウトラインの変更がきわめて高速で保存されます。

次の場合は、Essbase でデータベースの再構築およびインデックスの変更が行われません(Essbase ではデータベース・アウトラインのみを保存し、この処理は非常に高速です):

- 密次元の動的計算メンバーを追加、削除または移動する。

  メンバーが動的計算および保管の場合は、Essbase でデータベースが再構築されます。
- 密次元メンバーのストレージ・プロパティを動的計算および保管メンバーから非動的ストレージ・プロパティに変更する。
- 疎次元の動的計算メンバーまたは動的計算および保管メンバーのストレージ・プロパティを非動的ストレージ・プロパティに変更する。

- 動的計算メンバーまたは動的計算および保管メンバーの名前を変更する。

次の場合、Essbase では、データベースは再構築されませんが、データベースのインデックスは再構築されます。この処理はきわめて高速です:

- 疎次元の動的計算メンバーまたは動的計算および保管メンバーを追加、削除または移動する。
- 密次元メンバーのストレージ・プロパティを非動的値から動的計算および保管に変更する。

次の場合、Essbase ではデータベースが再構築されます:

- 密次元の動的計算および保管メンバーを追加、削除または移動する。

  メンバーが動的計算の場合、Essbase ではデータベースは再構築されません。
- 密次元の動的計算および保管メンバーを動的計算メンバーに変更する。
- 密次元の動的計算メンバーを動的計算および保管メンバーに変更する。
- 密次元の非動的メンバーのストレージ・プロパティを動的計算に変更する。
- 密次元のストレージ・プロパティを動的計算メンバーから非動的値に変更する。
- 疎次元の非動的メンバーのストレージ・プロパティを動的計算または動的計算および保管に変更する。

934 ページの「データベースの再構築のタイプ」を参照してください。

パーティション内のデータの動的計算

パーティションの透過領域、複製領域またはリンク領域で、動的計算メンバーと動的計算および保管メンバーを定義できます。第 15 章「パーティション・アプリケーションの設計」を参照してください。

たとえば、リモート・データベース(透過データベース・パーティション)上に子を持つ、上位レベルの疎次元メンバーに動的計算および保管タグを付けた場合、Essbase では、他のデータベースから子の値を取得するので、取得時間が長くなります。動的計算および保管のかわりに動的計算を使用できますが、後続の取得時間への影響が大きくなりすぎる可能性があります。

たとえば、ローカル・データベースが Corporate データベースで、そのデータベースに East、West、South および Central の地域データへの透過パーティションがあるとします。親メンバーの Market に動的計算および保管タグを付けることができます。

透過パーティションでは、リモート・データベース上の定義がローカル・データベース上の定義よりも優先されます。たとえば、ローカル・データベースで動的計算タグが付けられているメンバーに、リモート・データベースではそのタグが付けられていない場合、Essbase では、リモート・データベースから値が取得され、ローカル計算は実行されません。

複製パーティションを使用する場合は、動的計算および保管メンバーではなく、動的計算メンバーを使用することを検討してください。複製データの計算時に、

Essbase ではリモート・データベースから子ブロックを取得しないので、取得時間への影響は大きくありません。

注： Essbase では、データを複製するときに、各ソース・データ・ブロックと対応するターゲット・データ・ブロックで、タイム・スタンプを確認します。ソース・データ・ブロックの方が新しい場合は、Essbase でそのデータ・ブロックのデータが複製されます。ただし、動的に計算されるデータには、データ・ブロックとタイム・スタンプがありません。このため、Essbase では、動的に計算されるデータは必ず複製されます。

28 時系列データの計算

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

はじめに

時系列計算では、アウトラインに動的時系列メンバーが定義されていることを前提とします。時系列データの計算は、期間の最初と最後の値を計算して在庫を追跡したり、期間累計の値を計算したりするときに役に立ちます。

期首、期末および平均の値の計算

勘定科目タグが付けられた次元で、タイム・バランス・タグと差異レポート・タグを使用することで、勘定科目データでのタイム・バランス計算の実行方法を Essbase に指定できます。

Essbase では、通常、親の子に設定されている式を集計または計算することによって、時間タグが付けられた次元を計算します。ただし、タイム・バランス・タグや差異レポート・タグなどの勘定科目タグを使用して、別の種類の値を集計できます。たとえば、会計次元の親メンバーに期首のタイム・バランス・プロパティのタグを設定した場合、Essbase では、メンバーの最初の子の値を集計することによってメンバーを計算します。たとえば、Sample.Basic データベースのメジャー次元(会計次元)の Opening Inventory メンバーには、期首のタイム・バランス・プロパティが設定されています。このメンバーは、期間の初めの在庫を表します。期間が Qtr1 の場合、Opening Inventory は、Jan (Qtr1 分岐の最初のメンバー)の初めに使用可能な在庫を表します。

勘定科目タグを使用するには、勘定科目タグが付けられた次元と時間タグが付けられた次元が必要です。勘定科目タグが付けられた次元のメンバーには、期首、期末および平均タグ(タイム・バランス・プロパティ)、および支出タグ(差異レポート・プロパティ)のみを使用します。時間と勘定科目タグが付けられた次元は、密次元でも疎次元でもかまいません。

タイム・バランス勘定科目メンバーのセルの場合、^集計演算子が設定された時間次元以外の次元のメンバーは、平均の計算から除外されます; ただし、そのメンバーは、期首と期末の計算では対象となります。

注： 高機能計算を使用している場合、データベース・アウトラインの勘定科目タグを変更しても、Essbase によってデータベースは再構築されません。必要なデータ値を再計算するように Essbase に明示的に指定する必要があります。440 ページの「式または勘定科目プロパティの変更」を参照してください。

会計次元と時間次元の指定

次元に勘定科目タグを付けると、その次元に勘定科目タグが付けられたメンバーが存在することが Essbase で認識されます。次元に時間タグを付けると、その次元が勘定科目タグの期間の基準となる次元であることが Essbase で認識されます。

図 129 に示すように、メジャー次元に勘定科目タグが付けられ、年次元に時間タグが付けられています。

図 129　勘定科目タグと時間タグを表示した Sample.Basic アウトライン

Database: Basic (Current Alias Table: Default)
- Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
- Measures Accounts (Label Only)
- Product
- Market
- Scenario (Label Only)

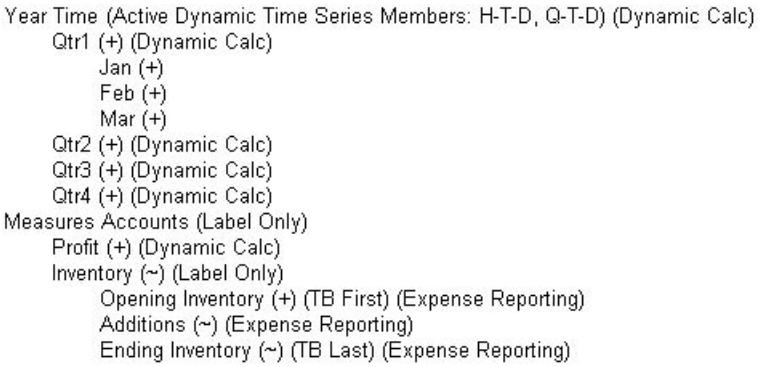
140 ページの「時間次元の作成」および 140 ページの「会計次元の作成」を参照してください。

各期間の期末の値のレポート

会計次元メンバーの場合、各期間の期末の値を次のレベルに上げるように Essbase を設定できます。各期間の期末の値をレポートするには、メンバーのタイム・バランス・プロパティを期末に設定します(タグはデータベース・アウトラインでは「TB Last」と表示されます)。

図 130 に示すように、勘定科目メンバー Ending Inventory は「TB Last」でタグ付けされています。Ending Inventory は、各四半期の最後の月の値を集計し、その月の親の値に使用します。たとえば、第 1 四半期の値は 3 月の値と同じです。

図 130 「期末」タグを示す Sample.Basic アウトライン



勘定科目メンバーに「期末」のタグを付ける方法は、140 ページの「タイム・バランス・プロパティの設定」を参照してください。

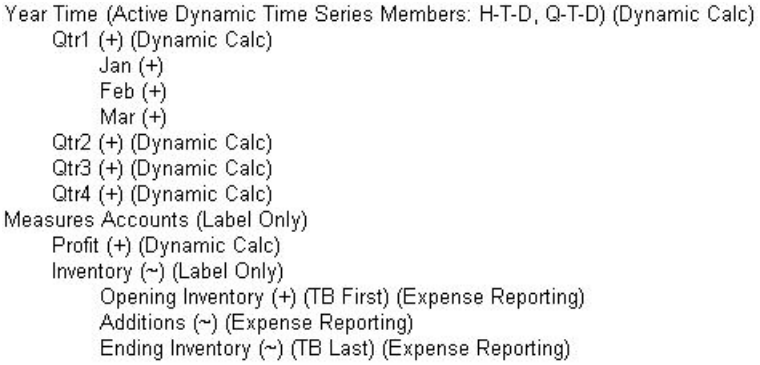
デフォルトでは、Essbase では親の値を計算するときに#MISSING 値またはゼロ(0)値はスキップされません。この値をスキップするように選択できます。#MISSING 値をスキップする方法と理由は、466 ページの「#MISSING 値とゼロ値のスキップ」を参照してください。

各期間の期首の値のレポート

会計次元メンバーの場合、各期間の期首の値を次のレベルに上げるように Essbase を設定できます。各期間の期首の値をレポートするには、メンバーのタイム・バランス・プロパティを期首に設定します(タグはデータベース・アウトラインでは「TB First」と表示されます)。

図 131 に示すように、勘定科目メンバー Opening Inventory は「TB First」でタグ付けされています。Opening Inventory は、各四半期の最初の月の値を集計し、その月の親の値に使用します。たとえば、第 1 四半期の値は 1 月の値と同じです。

図 131 「期首」タグを示す Sample.Basic アウトライン



勘定科目メンバーに「期首」のタグを付ける方法は、140 ページの「タイム・バランス・プロパティの設定」を参照してください。

デフォルトでは、Essbase では親の値を計算するときに#MISSING 値またはゼロ(0)値はスキップされません。この値をスキップするように選択できます。466 ページの「#MISSING 値とゼロ値のスキップ」を参照してください。

各期間の平均値のレポート

勘定科目メンバーの場合、期間の値を平均して、その平均値を 1 つ上のレベルに集計するように Essbase を設定できます。たとえば、1 月、2 月および 3 月の値を平均し、第 1 四半期の値としてその値を使用するように Essbase を設定できます。各期間の平均値をレポートするには、メンバーのタイム・バランス・プロパティを平均に設定します。

勘定科目メンバーに「平均」のタグを付ける方法は、140 ページの「タイム・バランス・プロパティの設定」を参照してください。

デフォルトでは、Essbase では親の値を計算するときに#MISSING 値またはゼロ(0)値はスキップされません。このため、平均を計算するときに Essbase では、子の値を集約して、子の数で割ります。このとき、子に#MISSING 値またはゼロ値があるかどうかは考慮されません。#MISSING 値とゼロ値をスキップするように Essbase を設定できます。466 ページの「#MISSING 値とゼロ値のスキップ」を参照してください。

#MISSING 値とゼロ値のスキップ

タイム・バランス計算を行うときに#MISSING 値とゼロ(0)値を処理する方法を Essbase で指定できます。#MISSING 値は、Essbase のマーカーの 1 つで、この場所にデータが存在しない、データに有効な値が含まれていない、またはデータが入力されていないことを示します。

デフォルトでは、Essbase では親の値を計算するときに、#MISSING 値または 0(ゼロ)値はスキップされません。

スキップ・プロパティを設定して、このデフォルトを上書きできます。142 ページの「スキップ・プロパティの設定」を参照してください。

たとえば、会計次元メンバーに「期末」および「#Missing のスキップ」のタグを付けた場合、Essbase では最後の欠落していない子が親に集計されます。表 77 の例で説明します:

表 77　#Missing のスキップの影響の例

勘定科目->時間	1 月	Feb	Mar	第 1 四半期
勘定科目メンバー(期末、#Missing のスキップ)	60	70	#MI	70

勘定科目に「平均」および「#Missing のスキップ」のタグを付けると、勘定科目に「平均」および「スキップしない」のタグを付けた場合とは異なる結果が得られます。「平均」および「スキップしない」で実行される計算では、データがスキップされないので正しい結果を得ることができます。ただし、子を持つ親の親は、平均を合計することによって集計されるので、「平均」および「#Missing のスキップ」のタグが付いた勘定科目の計算では、正しい結果を得ることができません。ただし、「動的計算」タグまたは「2 パス」タグを使用している場合を除きます。

「期首」、「期末」および「平均」のタグの影響の検討

表 78 は、会計次元メンバーに関するタイム・バランス(TB)の「期首」、「期末」および「平均」タグに基づいて Essbase で時間次元を集計する方法を示しています。

表 78　(TB)期首、期末および平均の影響の例

勘定科目->時間	1 月	Feb	Mar	第 1 四半期	
勘定科目メンバー 1	11	12	13	36	1 月 + 2 月 + 3 月の値
勘定科目メンバー 2(**TB 期首**)	20	25	21	20	1 月の値
勘定科目メンバー 3(**TB 期末**)	25	21	30	30	3 月の値
勘定科目メンバー 4(**TB 平均**)	20	30	28	26	1 月、2 月、3 月の平均

時間次元と会計次元に基づく式の設定

時間次元または会計次元に基づいてメンバー式を設定した場合、その式は、タイム・バランス計算によって上書きされる可能性があります。

表 79 は、期首在庫に「期首」のタグが付けられた例を示しています:

表 79　(TB)期首の影響の例

メジャー->年	1 月	Feb	Mar	第 1 四半期
期首在庫: **期首**	30000	28000	27000	30000

期首在庫に「期首」のタグが付けられているので、Essbase では、1 月の期首在庫の値を取得して、第 1 四半期の期首在庫を計算します。データベース・アウトラインで、第 1 四半期に設定されているメンバー式は、このタイム・バランス計算によって上書きされます。

期間累計値の計算

データの期間累計値を計算できます。たとえば、当月までの当四半期の売上高を計算できます。当月が 5 月の場合に、標準カレンダの四半期を使用すると、四半期合計は 4 月と 5 月の値の合計になります。

Essbase では、次の方法で期間累計値を計算できます:

- バッチ計算時に、@PTD 関数を使用して算出する方法
- ユーザーが値を要求したときに、動的時系列メンバーを使用して動的に算出する方法
- MDX クエリーの一部として、DTS 関数を使用して算出する方法

ここでは、動的時系列メンバーを使用して、期間累計値を動的に計算する方法を説明します。動的時系列メンバーの使用は、ほとんどの場合において最も効率的な方法です。たとえば、401 ページの「期間累計値の計算」を参照してください。

動的時系列メンバーの使用

期間累計値を動的に計算するには、時間のタグが付けられた次元で期間に動的時系列メンバーを使用する必要があります。464 ページの「会計次元と時間次元の指定」を参照してください。

動的時系列メンバーは、データベース・アウトラインに直接作成しません。かわりに、事前定義済の動的時系列メンバーを使用可能にし、適切な世代番号と関連付けます。

たとえば、四半期間累計値を計算するには、Q-T-D メンバーを使用可能にして、そのメンバーを動的時系列メンバーを適用する世代に関連付けます。Sample.Basic では、四半期を含んでいる世代は世代番号 2 です。この世代番号は第 1 四半期、第 2 四半期、第 3 四半期および第 4 四半期を含んでいます。Essbase では、Q-T-D という動的時系列メンバーを作成し、そのメンバーを世代 2 に関連付けます。Q-T-D メンバーは、四半期の当月までの月次の値を計算します。469 ページの「動的時系列メンバーを使用可能にする」を参照してください。

動的時系列メンバーはデータベース・アウトラインにメンバーとして表示されません。ただし、Essbase によって、現在アクティブな動的時系列メンバーが時間次元のコメントにリスト表示されます。図 132 のアウトラインでは、H-T-D(累計)と Q-T-D(四半期間累計)がアクティブです。H-T-D は世代 1 に関連付けられ、Q-T-D は世代 2 に関連付けられています。

図 132　動的時系列を示す Sample.Basic アウトライン

```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
    Qtr1 (+) (Dynamic Calc)
    Qtr2 (+) (Dynamic Calc)
        Apr (+)
        May (+)
        Jun (+)
    Qtr3 (+) (Dynamic Calc)
    Qtr4 (+) (Dynamic Calc)
```

Essbase には、次の 8 個の事前定義の動的時系列メンバーが用意されています:

- HTD(累計)
- Y-T-D(年次累計)
- S-T-D(季節累計)
- P-T-D(期間累計)
- Q-T-D(四半期間累計)
- M-T-D(月次累計)
- W-T-D(週次累計)
- D-T-D(日次累計)

これらのメンバーは、最大 8 レベルの期間累計レポートを提供します。使用するメンバーの数と種類は、データとデータベース・アウトラインによって異なります。

たとえば、データベースに 1 時間ごと、日次、週次、月次、四半期ごとおよび年次のデータが保管されている場合は、日次累計(D-T-D)、週次累計(W-T-D)、月次

累計(M-T-D)、四半期間累計(Q-T-D)、年次累計(Y-T-D)の情報をレポートできます。

データベースに過去 5 年間の月次データが保管されている場合は、特定の年までの年次累計(Y-T-D)と累計(H-T-D)の情報をレポートできます。

データベースで季節のデータを追跡している場合は、期間累計(P-T-D)または季節累計(S-T-D)の情報をレポートできます。

動的時系列メンバーは、データに関係なく、時間次元の最上位の世代番号以外であればどの世代にも関連付けることができます。たとえば、P-T-D メンバーを使用して四半期間累計情報をレポートできます。動的時系列メンバーを時間次元のレベル 0 メンバーに関連付けることはできません。

注： 動的計算用に設定されたメンバーを動的時系列計算で使用する場合は、これらのメンバーにタイム・バランス・プロパティ(期首、期末、平均、#Missing のスキップ)を割り当てないことをお薦めします。割り当てた場合、会計次元の親メンバーについて間違った値を取得する可能性があります。

動的時系列メンバーを使用可能にする

動的時系列メンバーを使用するには、そのメンバーを使用可能にする必要があります。必要に応じて、動的時系列メンバーの別名を指定できます。470 ページの「動的時系列メンバーの別名の指定」を参照してください。

➤ 動的時系列メンバーを使用可能にする方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「動的時系列メンバーを使用可能にする」を参照してください。

注： 表示される世代の数は、時間次元にある世代の数によって異なります。動的時系列メンバーを最上位の世代(レベル 0 メンバー)に割り当てることはできません。

データベース・アウトラインで動的時系列メンバーを使用可能にすると、Essbase では時間のタグが付けられた次元にコメントが追加されます。たとえば、次の Sample.Basic の年次元には、H-T-D と Q-T-D が定義されています:

```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
```

動的時系列メンバーを使用不可にする

動的時系列メンバーを使用不可にするには、事前定義のメンバーを使用しないように Essbase を設定します。

➤ 動的時系列メンバーを使用不可にする方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「動的時系列メンバーを使用不可にする」を参照してください。

動的時系列メンバーの別名の指定

事前定義の動的時系列メンバーに別名を指定できます。たとえば、Q-T-D 動的時系列メンバーの別名として QtrToDate を指定できます。その別名を使用して、Smart View またはレポートで、動的時系列メンバーを取得できます。

動的時系列メンバーごとに最大 8 個の別名を作成できます。Essbase では、各別名は指定した動的時系列別名テーブルに保存されます。

➤ 動的時系列メンバーの別名を作成するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「動的時系列メンバーの別名の作成」を参照してください。

別名の指定および表示の詳細は、152 ページの「別名の設定」を参照してください。

動的時系列メンバーに対する事前定義の世代名の適用

動的時系列メンバーを使用可能にして、そのメンバーを世代番号に関連付けると、Essbase によってその世代番号の事前定義の世代名が作成されます。158 ページの「世代およびレベルの命名」を参照してください。

➤ 世代名およびレベル名を表示するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「世代およびレベルの名前付け」を参照してください。

表 80 は、動的時系列メンバーとそれに対応する世代名を示しています:

表 80　動的時系列メンバーおよび対応する世代名

メンバー	世代名
D-T-D	日
H-T-D	履歴
M-T-D	月
P-T-D	期間
Q-T-D	四半期
S-T-D	季節
W-T-D	週

メンバー	世代名
Y-T-D	年

このメンバーと世代名は、Essbase で使用するために予約されています。いずれかの世代名を使用して時間次元に世代名を作成すると、Essbase によって対応する動的時系列メンバーが自動的に作成され、使用可能になります。

たとえば、Sample.Basic で、世代番号 2 に対して四半期という世代名を作成できます。四半期には、第 1 四半期メンバー、第 2 四半期メンバーなどの四半期ごとのデータが含まれます。世代名四半期を作成すると、Essbase で動的時系列メンバー Q-T-D が自動的に作成され、使用可能になります。

期間累計値の取得

動的時系列メンバーを取得するときには、期間累計値を計算する期間を Essbase で指定する必要があります。この期間は、最新期間と呼ばれ、時間次元のレベル 0 メンバーである必要があります。

➤ 最新期間を指定するには、次の方法を使用します:

- 特定のメンバーについて、Smart View で、最新期間のメンバー名を指定します。その名前を動的時系列メンバーまたは別名の後に置きます。たとえば、Q-T-D(May)と指定すると、4 月と 5 月の値を加算することによって、四半期間累計値が戻されます。
- 取得するには、次のいずれかの方法で最新期間を指定します:
  - レポート・ライターで<LATEST コマンドを使用します。
  - Smart View で DTS メンバーの選択オプションを指定します。

    メンバー固有の設定(Q-T-D(May)など)は、<LATEST や最新時系列オプションの設定に優先します。

図 133 の例では、「Q-T-D(May)」に、「Apr」と「May」の値の加算(8644 + 8929 = 17573)によって得られる「May」の期間累計値が表示されています。

図 133 「May」の期間累計値を表示したスプレッドシート

	Measures	Product	Market	Scenario
Qtr1	24703			
Apr	8644			
May	8929			
Jun	9534			
Qtr2	27107			
Qtr3	27912			
Qtr4	25800			
Year	105522			
Q-T-D(May)	17573			

透過パーティションでの動的時系列メンバーの使用

動的時系列メンバーを含むアウトラインの透過パーティションでのクエリー時間を最適化するには、`essbase.cfg` の設定 TARGETTIMESERIESOPT を使用します。

『Oracle Essbase テクニカル・リファレンス』および第 16 章「パーティションの作成および管理」を参照してください。

29 ブロック・ストレージ・データベース用の計算スクリプトの作成

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 30 章「ブロック・ストレージ・データベース用の計算スクリプトの例の確認」
- 第 23 章「ブロック・ストレージ・データベース用の式の作成」
- 第 58 章「計算の最適化」
- 第 64 章「集約ストレージ・データベースでのカスタム計算および割当ての実行」

この章のすべての例は、Sample.Basic データベースに基づいています。

この章で参照される計算コマンドの詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

計算スクリプトの使用

計算スクリプトには一連の計算コマンド、等式および式が記述されています。計算スクリプトを使用してデータベース・アウトラインで定義されていない計算を定義できます。

計算スクリプトでは、デフォルトの計算(CALC ALL)またはユーザーが選択した計算(データベースの一部の計算、メンバー間でのデータ値のコピーなど)を実行できます。

次のタスクを実行するには、計算スクリプトを記述する必要があります:

- データベースのサブセットを計算する場合

  495 ページの「データベースのサブセットの計算」を参照してください。
- データベース内の密次元と疎次元の計算順序を変更する場合
- 特定の順序で複雑な計算を実行する場合や、データ間で複数の繰返しを必要とする計算を実行する場合(たとえば、一部の 2 パス計算には計算スクリプトが必要です)
- 勘定科目タグが設定されていない次元で 2 パス計算を実行する場合

  980 ページの「2 パス計算の使用」を参照してください。
- 通貨換算を実行する場合

  第 14 章「通貨換算アプリケーションの設計および作成」を参照してください。
- データベース・アウトラインの式とは異なるメンバー式を計算する場合(計算スクリプトの式は、データベース・アウトラインの式を上書きします)
- API インタフェースを使用してカスタム計算を動的に作成する場合
- 計算でフロー制御ロジックを使用する場合(たとえば、IF...ELSE....ENDIF または LOOP...ENDLOOP コマンドを使用する場合)
- 特定のメンバーのデータを消去するか、コピーする場合

  494 ページの「データのコピー」を参照してください。
- データベース計算に使用する一時変数を定義する場合

  481 ページの「データ変数の宣言」を参照してください。
- データベース・アウトラインの式または勘定科目プロパティを変更した後に、強制的にデータ・ブロックを再計算する場合
- データベースを計算するときに Essbase で高機能計算を使用する方法を制御する場合

  第 26 章「高機能計算についての理解」を参照してください。

次の計算スクリプトは、年、メジャー、市場および製品次元から実績値を計算します:

```
FIX (Actual)

  CALC DIM(Year, Measures, Market, Product);
```

```
ENDFIX
```

管理サービス・コンソールで計算スクリプト・エディタを使用すると、次の操作で計算スクリプトを作成できます:

- スクリプト・エディタのテキスト領域に計算スクリプトを入力する
- スクリプト・エディタのユーザー・インタフェース機能を使用してスクリプトを作成する
- テキスト・エディタでスクリプトを作成し、そのスクリプト・テキストを計算スクリプト・エディタに貼り付ける

Oracle Essbase Administration Services Online Help の「計算スクリプト・エディタについて」を参照してください。

Administration Services を使用して作成した計算スクリプトには、デフォルトで.csc 拡張子が付きます。計算スクリプトを Administration Services または Smart View で実行する場合は、ファイルに.csc 拡張子が必要です。ただし、計算スクリプトはテキスト・ファイルなので、MaxL または ESSCMD を使用してテキスト・ファイルを計算スクリプトとして実行できます。

計算スクリプトをメモリー内に文字列として定義すると、Essbase クライアントまたは Essbase サーバー上の API からアクセスできます。このように、ダイアログ・ボックスで、ユーザーの選択に基づいて計算スクリプトを動的に作成できます。

計算スクリプトの作成プロセス

計算スクリプトを作成するには、次のプロセスを使用します:

1. 計算スクリプトを作成するか、既存の計算スクリプトを開きます。

   Oracle Essbase Administration Services Online Help の「スクリプトの作成」または「スクリプトを開く」を参照してください。

2. 計算スクリプトの内容を入力または編集します。

   次の詳細は、Oracle Essbase Administration Services Online Help の「計算スクリプト・エディタについて」を参照してください:

   - アウトラインへのスクリプトの関連付け
   - アウトライン・ツリーでのメンバーのサーチ
   - アウトライン・ツリーからスクリプトへの次元、メンバーおよび別名の挿入
   - ツリーからスクリプトへの関数とコマンドの挿入
   - 構文のオートコンプリートの使用
   - スクリプト構文の検査
   - スクリプトの実行
   - スクリプトの色分けされた要素の表示
   - スクリプトでのテキストのサーチ

- フォントの変更

3. 計算スクリプトを検証します。

   479 ページの「構文の検査」を参照してください。

4. 計算スクリプトを保存します。

   505 ページの「計算スクリプトの保存」を参照してください。

5. 計算スクリプトを実行します。

   506 ページの「計算スクリプトの実行」を参照してください。

6. 計算スクリプトの結果を確認します。

   507 ページの「計算結果の確認」を参照してください。

7. 必要に応じて、計算スクリプトで他の操作を行います。

   Oracle Essbase Administration Services Online Help で、次のトピックを参照してください:

   - 「オブジェクトのロックおよびロック解除」
   - 「スクリプトのコピー」
   - 「スクリプトの名前変更」
   - 「スクリプトの削除」
   - 「スクリプトの印刷」

計算スクリプト構文の理解

Essbase には、データベースの計算方法の制御に使用できる柔軟なコマンドのセットが用意されています。コマンドと式から計算スクリプトを作成できます。

計算スクリプトを作成するときは、次のルールを適用する必要があります:

- 各式または計算スクリプト・コマンドはセミコロン(;)で終わります。例:

  例 1

  ```
  CALC DIM(Product, Measures);
  ```

  例 2

  ```
  DATACOPY Plan TO Revised_Plan;
  ```

  例 3

  ```
  "Market Share" = Sales % Sales -> Market;
  ```

  例 4

  ```
  IF (Sales <> #MISSING)
  ```

```
 Commission = Sales * .9;

 ELSE

  Commission = #MISSING;

ENDIF;
```

次のコマンドはセミコロンで終わる必要はありません:

```
IF

ENDIF

ELSE

ELSIF

FIX

ENDFIX

EXCLUDE

ENDEXCLUDE

LOOP

ENDLOOP
```

注： 必須ではありませんが、式中の各 ENDIF ステートメントの後にセミコロンを付けることをお薦めします。

- メンバー名が次のいずれかの条件に該当する場合は、そのメンバー名を二重引用符(" ")で囲みます:
  - スペースが含まれている場合。

    たとえば、次の式では、Opening Inventory および Ending Inventory が二重引用符で囲まれています:

```
"Opening Inventory" = "Ending Inventory" - Sales + Additions;
```

- 演算子、関数名またはキーワードと同じ場合。

  1206 ページの「計算スクリプト、レポート・スクリプト、式、フィルタ、代替変数値および環境変数値での命名規則」を参照してください。

- ハイフン(-)、アスタリスク(*)、スラッシュ(/)などの英数字以外の文字が含まれている場合。

  1206 ページの「計算スクリプト、レポート・スクリプト、式、フィルタ、代替変数値および環境変数値での命名規則」を参照してください。

- すべて数字であるか、数字で始まっている場合。

  例: "100"または"10Prod"

- アンパサンド(&)で始まっている場合。先頭のアンパサンド(&)は代替変数用に予約されています。メンバー名が&で始まる場合、メンバー名を引用符で囲みます。

  注： 計算スクリプトで代替変数を引用符で囲まないでください。

- ドット(.)が含まれている場合。

  例: 1999.Jan または.100

- IF ステートメントまたは相互依存の式を使用している場合は、式を丸カッコで囲み、その式を指定したメンバーに関連付けます。

  たとえば、次の式は、データベース・アウトラインの Commission メンバーに関連付けられています:

```
     Commission
(IF(Sales < 100)
 Commission = 0;
ENDIF;)
```

- 式中の各 IF ステートメントは、ENDIF ステートメントで終わります。

  たとえば、前の式では簡単な IF...ENDIF ステートメントが使用されています。

- 別の IF ステートメントにネストされた IF ステートメントを使用している場合は、各 IF ステートメントを ENDIF ステートメントで終わらせます。

  例:

```
     "Opening Inventory"
(IF (@ISMBR(Budget))
 IF (@ISMBR(Jan))
  "Opening Inventory" = Jan;
 ELSE
  "Opening Inventory" = @PRIOR("Ending Inventory");
 ENDIF;
ENDIF;)
```

- ELSE または ELSEIF ステートメントは、ENDIF ステートメントで終了する必要はありません。

  例:

```
     Marketing
(IF (@ISMBR(@DESCENDANTS(West)) OR @ISMBR(@DESCENDANTS(East)))
 Marketing = Marketing * 1.5;
ELSEIF(@ISMBR(@DESCENDANTS(South)))
 Marketing = Marketing * .9;
ELSE
 Marketing = Marketing * 1.1;
ENDIF;)
```

  注： 式中に ELSEIF(1 語)ではなく ELSE IF(スペースあり)を使用した場合は、IF ステートメントに対応する ENDIF ステートメントを記述する必要があります。

- 各 FIX ステートメントは、ENDFIX ステートメントで終わります。

  例:

```
     FIX(Budget,@DESCENDANTS(East))
 CALC DIM(Year, Measures, Product);
ENDFIX
```

- 各 EXCLUDE ステートメントは、ENDEXCLUDE ステートメントで終わります。

計算スクリプトを記述するときは、計算スクリプト・エディタ構文チェッカを使用して、構文を検査します。479 ページの「構文の検査」を参照してください。

コメントの追加

計算スクリプトにコメントを追加して、注釈を付けることができます。Essbase では、計算スクリプトの実行時にこのコメントを無視します。

コメントを含めるには、コメントを/*で開始し、*/で終了させる必要があります。例:

```
/* This calculation script comment

 spans two lines. */
```

構文の検査

Essbase には、計算スクリプトの構文エラー(関数名の入力ミスなど)にフラグを付ける構文チェッカが用意されています。その結果は、管理サービス・コンソールのメッセージ・パネルに表示されます。

構文エラーが見つからなかった場合は、Essbase により構文の検査が正常に終了したことが示されます。

構文エラーが見つかった場合は、Essbase により構文の検査に失敗したことが示され、エラーが 1 つずつ表示されます。通常、エラー・メッセージにはエラーが発生した行の番号と簡単な説明が記載されます。たとえば、計算スクリプト・コマンドの最後にセミコロン行末文字がない場合は、Essbase で次のようなメッセージが表示されます:

```
Error: line 1: invalid statement; expected semicolon
```

最後のエラーまでくると、Essbase で次のメッセージが表示されます:

```
No more errors
```

➤ 計算スクリプト・エディタで計算スクリプトの構文を検査する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「スクリプト構文の検査」を参照してください。

注： 構文チェッカでは、計算スクリプトが予想どおりに動作しないときに発生するセマンティク・エラーを特定できません。セマンティク・エラーを見つけるには、計算を実行して、結果が予想どおりかどうかを確認します。507 ページの「計算結果の確認」を参照してください。

計算コマンドの使用

このセクションのトピックでは、機能別にグループ化された計算コマンドについて説明します。参照:

- 480 ページの「データベース・アウトラインの計算」
- 481 ページの「計算のフローの制御」
- 481 ページの「データ変数の宣言」
- 482 ページの「データベース計算用のグローバル設定の指定」

データベース・アウトラインの計算

表 81 に、データベース・アウトラインの構造と式に基づくデータベース計算を実行する計算コマンドをリストします。

表 81　データベースを計算するためのコマンドのリスト

コマンド	計算
CALC ALL	アウトラインに基づくデータベース全体
CALC DIM	指定された次元

コマンド	計算
CALC TWOPASS	勘定科目タグが付けられた次元における、2 パス・タグが付けられたすべてのメンバー
membername	データベース・アウトラインのメンバーに適用される式。membername は、式が適用されるメンバーの名前
CALC AVERAGE	勘定科目タグが付けられた次元における、平均タグが付けられたすべてのメンバー
CALC FIRST	勘定科目タグが付けられた次元における、期首タグが付けられたすべてのメンバー
CALC LAST	勘定科目タグが付けられた次元における、期末タグが付けられたすべてのメンバー
CCONV	通貨換算

計算のフローの制御

表 82 に、計算のフローを操作するコマンドをリストします:

表 82　計算のフローを制御するためのコマンドのリスト

コマンド	計算
FIX...ENDFIX	包含によってデータベースのサブセットを計算
EXCLUDE...ENDEXCLUDE	除外によってデータベースのサブセットを計算
LOOP...ENDLOOP	コマンドの繰返し回数を指定

また、IF と ENDIF のコマンドを使用して、条件付き計算を指定することもできます。

注： Essbase では、1 つの計算スクリプトから別の計算スクリプトへの分岐はできません。

データ変数の宣言

表 83 に、一時変数を宣言し、必要に応じてその初期値を設定するコマンドをリストします。一時変数には、計算の中間結果が保管されます。

計算スクリプトで代替変数も使用できます。488 ページの「計算スクリプトでの代替変数の使用」を参照してください。

表 83　データ変数を宣言するためのコマンドのリスト

コマンド	計算
ARRAY	1 次元の配列変数を宣言
VAR	単一の値を含む一時変数を宣言

一時変数に保管される値は、計算スクリプトの実行中にのみ存在します。一時変数の値についてはレポートできません。

変数および配列の名前は、次のいずれかの文字を含む文字列になります:

- 文字 a-z
- 数字 0-9
- 特殊文字: $(ドル記号)、#(シャープ記号)および_(アンダースコア)

通常、配列は、メンバー式の一部として変数を保管するときに使用します。配列変数のサイズは、対応する次元のメンバー数によって決まります。たとえば、シナリオ次元に 4 つのメンバーがある場合、次のコマンドによって、4 つのエントリを持つ Discount という配列が作成されます:

```
ARRAY Discount[Scenario];
```

一度に複数の配列を使用できます。

データベース計算用のグローバル設定の指定

表 84 に、計算動作を定義するコマンドをリストします:

表 84　計算動作を定義するためのコマンドのリスト

コマンド	計算
SET AGGMISSG	計算中の Essbase での#MISSING 値の処理方法を指定します。
SET CACHE	デフォルトの計算キャッシュ・サイズを調整します。
SET CALCPARALLEL	並列計算を使用可能にします。961 ページの「並列計算の使用」を参照してください。
SET CALCTASKDIMS	並列計算のタスクの特定に使用する次元の数を増やします。961 ページの「並列計算の使用」を参照してください。
SET CLEARUPDATESTATUS	高機能計算のデータ・ブロックを Essbase がマークする方法で制御します。432 ページの「SET CLEARUPDATESTATUS コマンドの使用」を参照してください。
SET CREATEBLOCKEQ	疎次元のメンバーに非定数値を割り当てる際のブロックの作成を制御する「等式によるブロックの作成」設定をオンおよびオフにします。974 ページの「疎次元のメンバーに割り当てられた非定数値」を参照してください。
SET CREATENONMISSINGBLK	潜在的なデータ・ブロックでの計算を使用可能にし、結果が#MISSING でない場合に、これらのブロックを保存します。
SET FRMLBOTTOMUP	大規模なデータベース・アウトラインの中の疎次元の式の計算を最適化します。973 ページの「大規模なデータベース・アウトライン内の疎次元に基づく式の最適化」を参照してください。
SET LOCKBLOCK	疎メンバー式を計算するときに Essbase で同時にロックできるブロックの最大数を指定します。
SET MSG SET NOTICE	計算を追跡するメッセージを表示します。
SET RUNTIMESUBVARS	計算スクリプトで使用されるランタイム代替変数を宣言します。488 ページの「計算スクリプトでのランタイム代替変数の使用」を参照してください。

コマンド	計算
SET UPDATECALC	高機能計算をオンおよびオフにします。430 ページの「高機能計算のオンとオフ」を参照してください。
SET UPTOLOCAL	通貨換算の場合に、定義済の同じ通貨を使用する親に集計を限定します。217 ページの「データベースの計算」を参照してください。

計算スクリプト内の SET コマンドは、同じ SET コマンドが次に検出されるまで有効です。

次の計算スクリプトでは、Essbase で年次元を計算するときに詳細レベルのメッセージ(SET MSG DETAIL;)が表示され、メジャー次元を計算するときに要約レベルのメッセージ(SET MSG SUMMARY;)が表示されます:

```
   SET MSG DETAIL;
CALC DIM(Year);

SET MSG SUMMARY;

CALC DIM(Measures);
```

SET 計算コマンドの中には、データベースへの追加パスをトリガーするものがあります。

次の計算スクリプトでは、Essbase で Qtr1 と SET AGGMISSG オンのメンバーの組合せを計算し、次にデータベースに対し 2 回目の計算パスを行い、East と SET AGGMISSG オフのメンバーの組合せを計算します:

```
   SET AGGMISSG ON;
Qtr1;
SET AGGMISSG OFF;

East;
```

980 ページの「2 パス計算の使用」も参照してください。

計算スクリプトでの式の使用

計算スクリプトにはメンバー式を入れることができます。メンバー式を入れた場合、その式はデータベース・アウトラインでメンバーに適用されている競合する式より優先されます。

計算スクリプトでは、次の両方の操作を実行できます:

- データベース・アウトラインのメンバー式を計算します
- 式を定義します

データベース・アウトラインのメンバーに適用されている式を計算するには、後にセミコロン(;)を付けたメンバー名を使用します。たとえば、次のコマンドでは、データベース・アウトラインの Variance メンバーに適用されている式を計算します:

```
Variance;
```

アウトラインを計算して得られた値を上書きするには、計算スクリプトで定義した式を手動で適用します。たとえば、次の式では、データベース内を循環して、メンバー Payroll、Marketing および Misc の値を加算し、その結果を Expenses メンバーに置きます。この式は、データベース・アウトラインで Expenses メンバーに置かれている式を上書きします:

```
Expenses = Payroll + Marketing + Misc;
```

注： 式は共有メンバーまたは「ラベルのみ」メンバーには適用できません。

参照:

- 484 ページの「基本的な等式」
- 485 ページの「条件付き等式」
- 485 ページの「相互依存の式」

第 23 章「ブロック・ストレージ・データベース用の式の作成」を参照してください。

基本的な等式

計算スクリプトで等式を使用して、値をメンバーに割り当てることができます。等式の構文は次のとおりです:

```
member
 =
mathematical_expression
;
```

member は、データベース・アウトラインのメンバー名です。
mathematical_expression は、有効な算術演算です。

Essbase によって式が評価され、その値が指定されたメンバーに割り当てられます。

次の例では、Essbase はデータベースのデータを循環しながら、Sales 内の値から COGS 内の値を減算し、結果を Margin に配置します:

```
Margin = Sales - COGS;
```

この例では、Essbase はデータベースのデータを循環しながら、Retail 内の値から Cost 内の値を減算し、結果の値を Retail の値のパーセンテージとして計算し、その結果を Markup に配置します:

```
Markup = (Retail - Cost) % Retail;
```

>(より大きい)と<(より小さい)の論理演算子を等式で使用することもできます。

次の例では、2 月の売上高が 1 月の売上高より多い場合、Sales Increase Flag の値は 1 になります。そうでない場合、値は 0 になります:

```
Sales Increase Flag = Sales -> Feb > Sales -> Jan;
```

条件付き等式

計算スクリプトでメンバー式の一部として IF ステートメントを使用する場合は、次のことを実行する必要があります:

- IF ステートメントを単一のメンバーと関連付けます
- IF ステートメントを丸カッコで囲みます

次の例では、IF...ENDIF ステートメント全体を丸カッコで囲み、Profit メンバーと関連付けています(`Profit(IF(...)...)`):

```
   Profit
(IF (Sales > 100)
  Profit = (Sales - COGS) * 2;
ELSE
  Profit = (Sales - COGS) * 1.5;
ENDIF;)
```

Essbase では、データベースを循環して、次の計算を実行します:

1. IF ステートメントによって、現在のメンバーの組合せに対する Sales の値が 100 より大きいかどうかがチェックされます。
2. Sales が 100 より大きい場合、Essbase では Sales の値から COGS の値が減算され、その差に 2 が乗算されて、結果が Profit に置かれます。
3. Sales が 100 以下の場合、Essbase では Sales の値から COGS の値が減算され、その差に 1.5 が乗算されて、結果が Profit に置かれます。

相互依存の式

計算スクリプトで相互依存の式を使用するときは、IF ステートメントの場合と同じルールが適用されます。次のことを実行する必要があります:

- 式を単一のメンバーと関連付けます
- 式をカッコで囲みます

次の例では、式全体を丸カッコで囲み、Opening Inventory メンバーと関連付けています:

```
   "Opening Inventory"
(IF(NOT @ISMBR (Jan))
 "Opening Inventory" = @PRIOR("Ending Inventory");
ENDIF;)
"Ending Inventory" = "Opening Inventory" - Sales + Additions;
```

計算スクリプトを使用した高機能計算の制御

疎次元メンバーに式があり、その式に次のいずれかのタイプの関数が含まれているとします:

- 関係(@PRIOR、@NEXT など)
- 財務(@NPV、@INTEREST など)

Essbase では、データ・ブロックが高機能計算用に消去としてマークされている場合でも、式を含んでいるデータ・ブロックを必ず再計算します。

434 ページの「データ・ブロックの計算」および第 26 章「高機能計算についての理解」を参照してください。

式と計算のグループ化

計算スクリプトで式と次元を慎重にグループ化すれば、計算パフォーマンスを大幅に改善できます。参照:

- 486 ページの「一連のメンバー式の計算」
- 487 ページの「一連の次元の計算」

一連のメンバー式の計算

式を計算するときは、必ず丸カッコを正しく使用してください。

次の例では、丸カッコが正しく使用されていないため、Essbase でデータベース内を 2 回循環し、1 回目にメンバー Qtr1 と Qtr2 の式が計算され、2 回目に Qtr3 の式が計算されます:

```
   (Qtr1;
Qtr2;)
Qtr3;
```

対照的に、次の構成では、Essbase でデータベース内を 1 回のみ循環し、メンバー Qtr1、Qtr2 および Qtr3 の式が計算されます:

```
   Qtr1;
Qtr2;
Qtr3;
```

または

```
   (Qtr1;
Qtr2;
Qtr3;)
```

同様に、次の式では、Essbase でデータベース内を 1 回循環し、1 つのパスで両方の式が計算されます:

```
   Profit = (Sales - COGS) * 1.5;
Market = East + West;
```

一連の次元の計算

一連の次元を計算するときは、可能なかぎり次元をグループ化することで、パフォーマンスを最適化できます。

たとえば、次の式では、Essbase でデータベース内を 1 回のみ循環します:

```
   CALC DIM(Year, Measures);
```

対照的に、次の構文では、Essbase でデータベース内を 2 回循環し、各 CALC DIM コマンドが 1 回ずつ計算されます:

```
   CALC DIM(Year);
CALC DIM(Measures);
```

計算スクリプトでの代替変数、ランタイム代替変数および環境変数の使用

サブトピック

- 計算スクリプトでの代替変数の使用
- 計算スクリプトでのランタイム代替変数の使用
- 計算スクリプトおよび式での環境変数の使用

代替変数は頻繁に変化する情報を参照する場合に使用されます。環境変数はユーザー固有のシステム設定のプレースホルダとして使用されます。

代替変数の一般的な説明は、118 ページの「代替変数の使用」を参照してください。

計算スクリプトでの代替変数の使用

計算スクリプトに代替変数を入れると、その代替変数は、Essbase によってユーザーがその代替変数用に指定した値に置き換えられます。代替変数は、頻繁に変更されるメンバーの情報やリストを参照する場合などに役に立ちます。

Administration Services で代替変数の値を作成し、指定します。

代替変数は、サーバー、アプリケーションおよびデータベース・レベルで作成できます。代替変数を計算スクリプトで使用するには、代替変数が計算スクリプトで使用可能である必要があります。たとえば、データベースレベルの代替変数は、データベース内の計算スクリプトでのみ使用でき、サーバーレベルの代替変数は、サーバー上のすべての計算スクリプトで使用できます。

計算スクリプトでは、代替変数の前にアンパサンド(&)を挿入します。Essbase では、アンパサンドで始まる文字列は代替変数として扱われ、計算スクリプトを解析する前に、その変数は割り当てられている値に置き換えられます。

たとえば、Sample.Basic で Qtr1 を当四半期として計算するには、次のようにします:

- 当四半期の代替変数(`&CurQtr`)を作成し、その変数に値 `Qtr1` を割り当てます
- `&CurQtr` 代替変数を使用する計算スクリプトを作成します。例:

```
     FIX(&CurQtr)
  CALC DIM(Measures, Product);
ENDFIX
```

488 ページの「計算スクリプトでのランタイム代替変数の使用」も参照してください。

計算スクリプトでのランタイム代替変数の使用

サブトピック

代替変数と同様に、代替変数が許可される場合には常に、ランタイム代替変数を計算スクリプトに含めることができます。計算スクリプトでは、ランタイム代替変数の名前の前にアンパサンド(&)を付ける必要があります。

ランタイム代替変数は、計算スクリプトで使用されるすべてのランタイム代替変数を SET RUNTIMESUBVARS 計算コマンドで宣言する必要がある点が代替変数と異なります。SET RUNTIMESUBVARS には、ランタイム代替変数の名前を指定する必要があります。ランタイム代替変数のデフォルト値の指定はオプションです。`<RTSV_HINT>rtsv_description</RTSV_HINT>`タグの文字列として指定する、ランタイム代替変数のデータ型およびデータ入力制限の説明もオプションです。

SET RUNTIMESUBVARS の次の例では、myMarket、salesNum および PointD の 3 つのランタイム代替変数が宣言されています。myMarket と pointD にはデフォルト値が設定されています(たとえば、myMarket のデフォルト値は"New York"です)。

salesNum の宣言にはデフォルト値は含まれていませんが、<RTSV_HINT>タグに説明が含まれています。

```
   SET RUNTIMESUBVARS
{
 myMarket = “New York”;
 salesNum <RTSV_HINT>salesNum: Input the value as an integer, such as 100</
RTSV_HINT>;
 pointD = "Actual"->"Final";
};
```

デフォルト値が設定されていないランタイム代替変数に対しては、次のいずれかの方法を使用してランタイムに値を指定して、SET RUNTIMESUBVARS コマンドに指定されているデフォルト値を上書きできます:

- キー/値のペアの文字列としてランタイム代替変数が指定された **with runtimesubvars** 構文を使用した MaxL execute calculation ステートメント。
- キー/値のペアの文字列としてランタイム代替変数が指定された EssCalcWithRuntimeSubVars API。
- クライアント・コンピュータ上のテキスト・ファイルまたはキー/値のペアの文字列としてランタイム代替変数を指定できる EssCalcFileWithRuntimeSubVars API。

キー/値のペアの文字列としてランタイム代替変数を指定する場合は、文字列を一重引用符で囲んで、キー/値のペアをセミコロンで区切る必要があります。このランタイム代替変数文字列の例では、4 つのランタイム代替変数の名前と値が指定されています(たとえば、"a"という名前のランタイム代替変数の値は 100 です):

```
   'a=100;b=@CHILDREN("100");c="Actual"->"Final";d="New York"'
```

ランタイム代替変数をテキスト・ファイルに指定する場合は、.rsv という拡張子のテキスト・ファイルをクライアント・コンピュータ上に作成します。(Essbase では、Essbase サーバー・コンピュータにあるランタイム代替変数ファイルはサポートされません。)ファイルの各行には、キー/値のペアとして 1 つのランタイム代替変数を指定し、最後にセミコロンを付ける必要があります。この.rsv ファイルの例では、4 つのランタイム代替変数の名前と値が指定されています(たとえば、"a"という名前のランタイム代替変数の値は 100 です):

```
   a=100;
b=200;
c=@CHILDREN("100");
d=@TODATE("DD/MM/YY","10/11/12");
```

計算が実行されると、ランタイム代替変数値が次の順序で決定されます:

1. **with runtimesubvars** 構文を使用した MaxL execute calculation ステートメント、EssCalcWithRuntimeSubVars または EssCalcFileWithRuntimeSubVars API で指定された値。

2. SET RUNTIMESUBVARS 計算コマンドに指定されたデフォルト値。

ランタイム代替変数を使用する際には、次のガイドラインを考慮してください:

- SET RUNTIMESUBVARS にランタイム代替変数を宣言したが、計算スクリプトでランタイム代替変数を使用しない場合、Essbase は未使用のランタイム代替変数の宣言を無視します(警告や例外は生成されません)。
- ランタイム代替変数は代替変数よりも優先されます。そのため、代替変数とランタイム代替変数の名前が同じ(たとえば、myProduct)の場合、ランタイム代替変数の値で、代替変数の値が上書きされます。
- 複数のランタイム代替変数の名前が同じで値が異なる場合は、ランタイム代替変数の最初のインスタンスの値のみが使用されて、後続の他のすべての値は無視されます。

ランタイム代替変数の名前と値の設定ルールは、代替変数の場合と同じです。120 ページの「代替変数の名前と値の設定ルール」を参照してください。

ランタイム代替変数のロギング

計算スクリプトに使用されているランタイム代替変数をログに書き込むには、essbase.cfg の ENABLERTSVLOGGING 構成設定を TRUE に設定します。ロギングは、Essbase サーバー、アプリケーション、またはデータベース・レベルで実装できます。

ランタイム代替変数のログ・エントリはアプリケーション・ログ・ファイルに書き込まれます。Essbase では、キー/値のペアの文字列(または、EssCalcFileWithRuntimeSubVars API を使用する場合は.rsv ファイルに指定されているキー/値のペアのリスト)ごとにアプリケーション・ログに 1 つのエントリが書き込まれます。

次の例では、2 つのランタイム代替変数(Entity と Currency)およびそれらの値が 1 つのエントリとしてログに書き込まれます:

```
   Executing calc script 'calcprofit.csc' with runtime substitution variables {Entity
=  “MyCompany” ; Currency =  “USD” ;}
```

ランタイム代替変数のデータ型および入力制限の指定

SET RUNTIMESUBVARS 計算コマンドでは、ランタイム代替変数の宣言に<RTSV_HINT>rtsv_description</RTSV_HINT>タグを含めることができます。rtsv_description は、ランタイム代替変数のデータ型およびデータ入力制限(たとえば、100 以下の整数)を記述する文字列です。</RTSV_HINT>文字列は、計算では使用されません。

EssGetRuntimeSubVars API では、SET RUNTIMESUBVARS のランタイム代替変数宣言に指定されているすべての情報(名前、値および説明)が取得されます。その後、<RTSV_HINT>文字列を使用して、ランタイムに値を入力するようにユーザーにプロンプトを表示したり、計算スクリプトに値を渡す前に入力データを検証できます。

SET RUNTIMESUBVARS の次の例では、ランタイム代替変数の名前と説明のみが各定義に指定されており、デフォルト値は指定されていません:

```
   SET RUNTIMESUBVARS
{
 myMarket <RTSV_HINT>myMarket: Input the value as a string, such as "New York"</
RTSV_HINT>;
 salesNum <RTSV_HINT>salesNum: Input the value as an integer, such as 100</
RTSV_HINT>;
 pointD <RTSV_HINT>pointD: Input the value as a member combination, such as "Actual"-
>"Final"</RTSVVAR_HINT>;
};
```

計算スクリプトおよび式での環境変数の使用

計算スクリプトでは、ユーザー固有のシステム設定のプレースホルダとしてシステム環境変数を使用できます。環境変数は、オペレーティング・システム・レベルで定義されているので、Essbase サーバー上のすべての計算スクリプトで使用できます。

注： 環境変数は MDX クエリーでは使用できません。

システム環境変数を宣言する方法は、オペレーティング・システムのドキュメンテーションを参照してください。

計算スクリプトで環境変数を使用するには、環境変数名の前にドル記号($)を挿入します。Essbase では、ドル記号で始まる文字列は環境変数として扱われ、計算スクリプトを解析する前に、その変数は割り当てられている値に置き換えられます。メンバー名が$で始まる場合は、その名前を引用符で囲みます。

計算スクリプトで環境変数を使用するときには、次のガイドラインに従います:

- 環境変数の名前:
  - 英数字またはアンダースコア(_)で構成する必要があります
  - ハイフン(-)、アスタリスク(*)、スラッシュ(/)などの英数字以外の文字を入れることはできません
  - Unicode モードのアプリケーションでは 320 バイト以下、非 Unicode モードのアプリケーションでは 320 文字以下にする必要があります
- 環境変数の値:
  - 先頭のドル記号($)以外は、どの文字でも使用できます
  - 数値かどうかに関係なく、引用符(" ")で囲む必要があります。例:

```
MY_PROD="100"

ENV_FILE="E:\temp\export1.txt"
```

環境変数を定義するときに値を引用符で囲まなかった場合、数値以外の値であれば、その環境変数が渡されるときに Essbase によって値が自動的に引用符で囲まれます。

数値の場合は、変数が渡されるときに Essbase によってその値が自動的に引用符で囲まれることはありません。これは、数値を渡すのか、それともメンバー名を渡すのかを Essbase で判断できないためです。たとえば、'Sales = $MY_SALES' where MY_SALES=700 のような計算スクリプト・ステートメントを使用する場合、目的は数値 700 を渡すことです。しかし、Essbase によって MY_SALES が引用符で囲まれると、MY_SALES はメンバー名として扱われます。その結果、数値ではなくメンバー名が渡されるので、エラーが発生します。変数の数値を文字列として扱う場合は、環境変数を定義するときにその値を引用符で囲む必要があります。

- Unicode モードのアプリケーションでは 256 バイト以下、非 Unicode モードのアプリケーションでは 256 文字以下にする必要があります

たとえば、環境変数を使用して、データ・ブロックをフラット・ファイルにエクスポートするときのエクスポート・ファイルのパスとファイル名を定義できます。次の計算スクリプトでは、パスとファイル名(E:\temp\export1.txt)が明示的に定義されています:

```
   SET DATAEXPORTOPTIONS
{
 DATAEXPORTLEVEL "ALL";
 DATAEXPORTOVERWRITEFILE ON;
};

FIX ("New York", "100-10");
 DATAEXPORT "File" "," "E:\temp\export1.txt";
ENDFIX;
```

パスとファイル名(ENV_FILE="E:\temp\export1.txt")を参照するために環境変数(ENV_FILE)を宣言し、計算スクリプトで次の構文を使用できます:

```
   DATAEXPORT "File" "," $ENV_FILE;
```

Essbase によって環境変数がユーザーの環境から取得した値に置き換えられます。

次の例では、Sales 値(CurrMbr="Sales")のみをエクスポートするために別の環境変数(CurrMbr)が定義されています:

```
   SET DATAEXPORTOPTIONS
{
 DATAEXPORTLEVEL "ALL";
 DATAEXPORTOVERWRITEFILE ON;
};

FIX ("New York", "100-10", $CurrMbr);
 DATAEXPORT "File" "," $ENV_FILE;
ENDFIX;
```

環境変数を使用して、RUNJAVA に渡された引数も解析できます。RUNJAVA は、Essbase のユーティリティで、計算スクリプトから直接カスタム定義関数を呼び出すことができます。たとえば、環境変数を使用してユーザーの電子メール・アドレスを取得できます。

次の例では、RUNJAVA ステートメントが電子メール通知を明示的に定義されたユーザー(to@somedomain.com および cc@mydomain.com)に送信します:

```
   RUNJAVA com.hyperion.essbase.calculator.EssbaseAlert "localhost"
“to@somedomain.com”  "cc@mydomain.com" "" "" "Mail Subject" "Mail Body" "";
```

ユーザー(ENV_TOMAIL="to@somedomain.com"と ENV_CCMAIL="to@mydomain.com")に対して環境変数を宣言し、計算スクリプトで次の構文を使用できます:

```
   RUNJAVA com.hyperion.essbase.calculator.EssbaseAlert "localhost" $ENV_TOMAIL
$ENV_CCMAIL "" "" "Mail Subject" "Mail Body" "";
```

データの消去およびコピー

データベースからデータのサブセットを消去したり、あるメンバーのセットから別のメンバーのセットにデータ値をコピーすることができます。参照:

- 493 ページの「データの消去」
- 494 ページの「データのコピー」

データの消去

表 85 に、データを消去するコマンドをリストします:

表 85　データを消去するためのコマンドのリスト

コマンド	計算
CLEARDATA	指定するセルの値を#MISSING に変更します。データ・ブロックは削除されません。 FIX コマンドを CLEARDATA コマンドと使用して、データベースのサブセットを消去します。
CLEARBLOCK	すべての密次元メンバーを含む、ブロックのコンテンツ全体を削除します。 Essbase では、CLEARBLOCK がブロック内のメンバーに対する FIX コマンドの内部にないかぎり、ブロック全体を削除します。
CLEARBLOCK UPPER	集計レベル・ブロックを削除します。
CLEARBLOCK NONINPUT	派生値を含んでいるブロックを削除します。計算操作によってすべて作成されたブロックには適用されますが、値がロードされたブロックには適用されません。

コマンド	計算
CLEARBLOCK DYNAMIC	動的計算および保管メンバーの組合せのブロックを削除します。 第 27 章「データ値の動的計算」を参照してください。
CLEARBLOCK EMPTY	空のブロックを削除します。

次の計算スクリプト・コマンドは、シナリオ次元が密か疎かによって、異なる結果を導き出します:

```
   FIX(Actual)
  CLEARBLOCK NONINPUT;
ENDFIX
```

- 密: コマンドは、入力データ値を含まず、シナリオ次元の Actual メンバーと交差するすべてのデータ・セルを削除します。
-

疎: コマンドは、シナリオ次元のメンバーが Actual であるブロックのみを削除します。

次の式では、Colas のすべての Actual データ値を消去します:

```
   CLEARDATA Actual -> Colas;
```

データベース全体を消去する方法は、Oracle Essbase Administration Services Online Help の「データの消去」を参照してください。

データのコピー

DATACOPY 計算コマンドは、データベース内のあるメンバー範囲から別のメンバー範囲にデータ・セルをコピーします。2 つの範囲は同じサイズである必要があります。たとえば、次の式は Actual 値を Budget 値にコピーします:

```
   DATACOPY Actual TO Budget;
```

FIX コマンドを使用すると、値のサブセットをコピーできます。たとえば、次の式は 1 月のみの Actual 値を Budget 値にコピーします:

```
   FIX (Jan)
  DATACOPY Actual TO Budget;
ENDFIX
```

495 ページの「FIX コマンドの使用」を参照してください。

データベースのサブセットの計算

データベースのサブセットを計算するには、次のいずれかの方法を使用します:

- メンバーのリストを計算するメンバー・セット関数を使用して式を作成します。

  495 ページの「メンバーのリストの計算」を参照してください。

- FIX...ENDFIX コマンドを使用して、包含によって値の範囲を計算します。

  495 ページの「FIX コマンドの使用」を参照してください。

- EXCLUDE...ENDEXCLUDE コマンドを使用して、除外によって値の範囲を計算します。

  497 ページの「除外コマンドの使用」を参照してください。

注： 高機能計算がオンの場合、新しく計算されたデータ・ブロックは、データベースの部分計算後に消去としてマークされません。データベースのサブセットを計算するときには、SET CLEARUPDATESTATUS AFTER コマンドを使用して、新しく計算されたブロックが必ず消去としてマークされるようにすることができます。このコマンドを使用すると、Essbase では高機能計算を使用してできるだけ効率的にデータベースが再計算されます。第 26 章「高機能計算についての理解」を参照してください。

メンバーのリストの計算

メンバー・セット関数は、指定したメンバーに基づいたメンバーのリストを生成します。たとえば、@IDESCENDANTS 関数を使用すると、指定したメンバーのすべての子孫のリストが生成されます。式でメンバー・セット関数を使用すると、式を計算する前に Essbase によってメンバーのリストが生成されます。

次の例では、メンバー Total Expenses で@IDESCENDANTS コマンドを使用すると、Total Expenses それ自体と、その子孫の Marketing、Payroll および Misc のリストが生成されます:

```
@IDESCENDANTS("Total Expenses");
```

FIX コマンドの使用

FIX コマンドを使用して、計算に含めるメンバーを定義します。

次の例では、East の子孫(New York、Massachusetts、Florida、Connecticut および New Hampshire)のみの Budget 値のみが計算されます:

```
   FIX(Budget,@DESCENDANTS(East))
  CALC DIM(Year, Measures, Product);
ENDFIX
```

次の例では、New Mkt という UDA が設定された East の子に対するメンバーの組合せが確定されます:

```
   FIX(@CHILDREN(East) AND @UDA(Market,"New Mkt"))
  Marketing = Marketing * 1.1;
ENDFIX
```

次の例では、ワイルドカード一致(???)を使用して、文字-10 で終わるメンバー名、つまりメンバー 100-10、200-10、300-10 および 400-10 を確定します:

```
   FIX(@MATCH(Product, "???-10"))
  Price = Price * 1.1;
ENDFIX
```

密次元で FIX コマンドのみを使用すると、Essbase によって、必要な値または指定したメンバーの値を含んだブロック全体が取得されます。I/O は影響を受けないので、計算パフォーマンス時間が向上します。

疎次元で FIX コマンドを使用すると、Essbase によって、指定した疎次元メンバーのブロックが取得されます。I/O は大幅に低下する可能性があります。

Essbase で、密次元メンバーで使用する FIX コマンドごとにデータベース内を 1 回循環します。可能な場合は、FIX ブロックを組み合せると計算パフォーマンスを向上させることができます。

たとえば、1 つの FIX コマンドを使用した場合、次の計算スクリプトでは、Essbase でデータベース内を 1 回のみ循環し、Actual と Budget の両方の値が計算されます:

```
   FIX(Actual,Budget)
  CALC DIM(Year, Measures);
ENDFIX
```

2 つの FIX コマンドを使用した場合、次の計算スクリプトでは、Essbase でデータベース内を 2 回循環し、1 回目は Actual データ値が計算され、2 回目は Budget データ値が計算されます:

```
   FIX(Actual)
  CALC DIM(Year, Measures);
ENDFIX
FIX(Budget)
  CALC DIM(Year, Measures);
ENDFIX
```

計算する次元のサブセットを FIX コマンド内で確定することはできません。たとえば、次の計算スクリプトでは、CALC DIM より前の FIX で、市場次元の特定のメンバーが確定されますが、CALC DIM の演算で市場次元全体が計算されるため、エラー・メッセージが戻されます:

```
   FIX(@CHILDREN(East) AND @UDA(Market,"New Mkt"))
  CALC DIM(Year, Measures, Product, Market);
ENDFIX
```

FIX コマンドは、他の FIX コマンド・ブロック内でネストできます。ただし、ネストされた FIX コマンドが誤って使用されると、不正確な結果が生じる場合があります。たとえば、次の計算スクリプトの目的は、East のすべての子に 1 を割り当ててから New York に 2 を割り当てることです:

```
   FIX (@CHILDREN(EAST))
  ''100-10''=1;
   FIX (''New York'')
     ''100-10''=2;
   ENDFIX
ENDFIX
```

しかし、ネストされた FIX コマンドは、そのコマンドより上の FIX コマンドで指定された次元のサブセットを確定します(これは許可されていません)。したがって、スクリプトは East のすべての子に 2 を割り当てます。これは、次のように記述されたようにスクリプトが実行されるからです:

```
   FIX (@CHILDREN(EAST),''New York'')
  ''100-10''=1;
  ''100-10''=2;
ENDFIX
```

ネストされた FIX コマンドを使用するよりも、2 つの個別の FIX コマンド・ブロックを使用してください。例:

```
   FIX (@CHILDREN(EAST))
  ''100-10''=1;
ENDFIX

FIX (''New York'')
  ''100-10''=2;
ENDFIX
```

FIX コマンドには制限があります。『Oracle Essbase テクニカル・リファレンス』を参照してください。

除外コマンドの使用

EXCLUDE...ENDEXCLUDE コマンドを使用して、計算から除外するメンバーを定義します。含めるメンバーを定義するよりも、含めないメンバーを指定する方が簡単な場合があります。

注： EXCLUDE コマンドには制限があります。『Oracle Essbase テクニカル・リファレンス』を参照してください。

DATAEXPORT コマンドを使用したデータのエクスポート

計算スクリプトで DATAEXPORT コマンドを使用すると、データをバイナリまたはテキストで、またはリレーショナル・データベースに直接エクスポートできます。データエクスポートの一連の計算コマンドでは、エクスポートするデータの種類が限定されており、出力とフォーマットの様々なオプションが用意されています。

注： バイナリ・ファイルへの DATAEXPORT は、Essbase のリリース間または 32 ビット・オペレーティング・システムと 64 ビット・オペレーティング・システムの間ではサポートされていません。

次のコマンド・シーケンスは、データをエクスポートするための一般的な計算スクリプトの構造を示しています:

```
   SET DATAEXPORTOPTIONS
 {
  DATAEXPORTLEVEL
   parameters
   ;
  DATAEXPORTDYNAMICCALC ON | OFF;
  DATAEXPORTNONEXISTINGBLOCKS ON | OFF;
  DATAEXPORTDECIMAL
   n
   ;
  DATAEXPORTPRECISION
   n
   ;
  DATAEXPORTCOLFORMAT ON | OFF;
  DATAEXPORTCOLHEADER
   dimensionName
   ;
  DATAEXPORTDIMHEADER ON | OFF;
  DATAEXPORTRELATIONALFILE ON | OFF;
  DATAEXPORTOVERWRITEFILE ON | OFF;
  DATAEXPORTDRYRUN ON | OFF;
  };
DATAEXPORTCOND
   parameters
   ;
FIX
 (
   fixMembers)

 DATAEXPORT
   parameters
   ;
```

```
ENDFIX;
```

次の表は、すべてオプションの SET DATAEXPORTOPTIONS コマンドの一覧を示しています:

- 表 86 - コンテンツ・オプション
- 表 87 - 出力フォーマット・オプション
- 表 88 - 処理オプション

表 86　SET DATAEXPORTOPTIONS コマンド: コンテンツ・オプション

コマンド	計算
DATAEXPORTLEVEL	次のいずれかのデータ値を指定します: ALL、LEVEL0 または INPUT DataExportLevel オプションにデータ値を指定する際には、次のガイドラインを使用します: ● 値では大文字と小文字は区別されません。たとえば、LEVEL0 でも level0 でも指定できます。 ● オプションで、値を引用符で囲みます。たとえば、LEVEL0 でも"LEVEL0"でも指定できます。 ● 値を指定しない場合、Essbase では、デフォルト値の ALL が使用されます。 ● 指定した値が正しくない場合(LEVEL 0 や LEVEL2 など)、Essbase では、デフォルト値の ALL が使用されます。
DATAEXPORTDYNAMICCALC	動的に計算された値のエクスポートを制御します
DATAEXPORTNONEXISTINGBLOCKS	すべての潜在的なデータ・ブロックからデータをエクスポートするのか、それとも既存のデータ・ブロックからのみデータをエクスポートするのかを指定します
DATAEXPORTDECIMAL	エクスポートした値の小数点以下の桁数を指定します
DATAEXPORTPRECISION	エクスポートした値の合計桁数を指定します

表 87　SET DATAEXPORTOPTIONS コマンド: 出力フォーマット・オプション

コマンド	計算
DATAEXPORTCOLFORMAT	縦欄フォーマットにするのか、縦欄でないフォーマットにするのかを指定します
DATAEXPORTCOLHEADER	列ヘッダーの密次元を指定します
DATAEXPORTDIMHEADER	すべての次元名を表示するヘッダー・レコードをファイルのデータと同じ順序で追加します
DATAEXPORTRELATIONALFILE	データをリレーショナル・データベースにインポートするためにテキスト・エクスポート・ファイルをフォーマットします

表 88　SET DATAEXPORTOPTIONS コマンド: 処理オプション

コマンド	計算
DATAEXPORTOVERWRITEFILE	名前と場所が同一の既存のファイルを置き換えるかどうかを指定します

コマンド	計算
DATAEXPORTDRYRUN	エクスポート・プロセス全体を実行しなくても、計算コマンドのセットを検証し、エクスポートの統計(時間の予測を含む)を表示できるようにします

➤ データのサブセットをエクスポートする計算スクリプトを作成するには、次のようにします:

1. SET DATAEXPORTOPTIONS コマンドを指定して、エクスポートのコンテンツ(表 86 を参照)、フォーマット(表 87 を参照)およびプロセス(表 88 を参照)のオプションを定義します。
2. DATAEXPORTCOND コマンドを使用して、データ値に基づいてデータを選択します。
3. FIX...ENDFIX または EXCLUDE...ENDEXCLUDE 計算コマンドを使用して、エクスポートするデータベースのスライスを選択します。
4. FIX...ENDFIX または EXCLUDE...ENDEXCLUDE グループの内部に DATAEXPORT コマンドを入れます。

   DATAEXPORT コマンドを使用して、エクスポートされたデータをリレーショナル・データベースに直接挿入する場合は、500 ページの「リレーショナル・データベースへのデータのエクスポート」を参照してください。
5. DATAIMPORTBIN 計算コマンドを使用して、以前エクスポートしたバイナリ・エクスポート・ファイルをインポートします。

   注： DATAIMPORTBIN は、Essbase のリリース間または 32 ビット・オペレーティング・システムと 64 ビット・オペレーティング・システムの間ではサポートされません。

   SET DATAIMPORTIGNORETIMESTAMP 計算コマンドを使用すると、一致するアウトライン・タイムスタンプのインポート要件を管理できます。

その他のエクスポート方法として、ESSCMD、MaxL および管理サービス・コンソールを使用してデータベースをバックアップする方法があります。レポート・ライターを使用して、出力テキスト・ファイルを作成し、データベースのサブセットを選択およびフォーマットできます(630 ページの「レポート・スクリプトを使用したテキスト・データのエクスポート」を参照)。

リレーショナル・データベースへのデータのエクスポート

DATAEXPORT コマンドを使用して、データをリレーショナル・データベースに直接挿入する場合は、次の点に注意してください:

- データが書き込まれるテーブルは、データがエクスポートされる前に存在している必要があります
- テーブルおよび列の名前にスペースを含めることはできません

デフォルトでは、エクスポートしたデータを挿入するときに、Essbase では行挿入メソッドを使用します。このメソッドでは、行が 1 つずつ挿入されます。パフォーマンスを向上させるために、バッチ挿入メソッドを使用できます(使用中のリレーショナル・データベースと ODBC ドライバがこの機能をサポートしている場合)。

注： 64 ビットの Essbase では、DATAEXPORT バッチ挿入メソッドを使用した、SQL データ・ソースへのデータの直接エクスポートをサポートしていません。

バッチ挿入を使用可能にするには、`essbase.cfg` の DATAEXPORTENABLEBATCHINSERT 構成設定を TRUE に設定します。Essbase 側でバッチ・サイズを判断するのではなく、一度に挿入する行数を制御する場合は、DEXPSQLROWSIZE 構成設定を使用して、バッチの行数(2 から 1000)を指定します。Essbase によってリレーショナル・データベースおよび ODBC ドライバがバッチ挿入をサポートするかどうかが決定できない場合は、行挿入メソッドが使用され、DEXPSQLROWSIZE(設定されている場合)は無視されます。

注： DATAEXPORTENABLEBATCHINSERT が TRUE に設定され、DEXPSQLROWSIZE が 1 に設定されている場合、バッチ挿入は使用できません(DEXPSQLROWSIZE を 1 に設定すると、行が 1 つずつ挿入されます)。

計算スクリプトを使用したデータのエクスポートの長所と短所

他の方法を使用してデータをエクスポートする場合と比べると、計算スクリプトを使用した場合には次の長所と短所があります:

- 長所
  - データのサブセットをエクスポートできます
  - 複数のエクスポート先(フラット・ファイル、リレーショナル・データベースおよびバイナリ・ファイル)をサポートしています
  - タイプ、フォーマットまたはデータのオプションが用意されています
  - 計算スクリプトの一部として、バッチ・プロセスに配置できます
  - DATAEXPORT はメモリーのカーネル・ストレージ・ブロックに直接アクセスするので、動的計算エクスポート・オプション(DATAEXPORTDYNAMICCALC)を使用しない場合は、エクスポートを非常に速く実行できます
  - バイナリのエクスポートで使用される圧縮フォーマットでは、エクスポート・ファイルに必要なストレージが他よりも少ないので、バイナリのエクスポートとインポートによって、より高速にデータをバックアップおよび復元できます
  - データの変更を追跡する他の計算コマンドの前および後に DATAEXPORT コマンドを使用することによって、デバッグ・ツールとして使用してバッチ計算結果を追跡できます

- 短所
  - レポート・ライターのフォーマットと比べると、データ・フォーマットのオプションに制限があります
  - 保管済メンバーと動的計算メンバーしか処理せず、属性メンバーと別名に対するサポートがありません
  - 集約ストレージ・データベースをサポートしていません
  - データをクライアントに直接エクスポートできません
  - DATAEXPORTNONEXISTINGBLOCKS を ON に設定している場合を除いて、動的計算データのエクスポート(DATAEXPORTDYNAMICCALC)が使用されるとパフォーマンスに重大な影響を与える可能性があります。

潜在的なブロックでの計算の有効化

密次元に設定された式を密メンバーで使用した場合、計算結果が密次元からの値で、オペランドが疎次元からの値であると、Essbase では、必要なブロックが自動的には作成されません。

次の例では、既存の実績データから売上高予算と支出データを作成するとします。Sales と Expenses はメジャー密次元のメンバーで、Budget と Actual はシナリオ疎次元のメンバーです。

```
   FIX(Budget)
  (Sales = Sales -> Actual * 1.1;
  Expenses = Expenses -> Actual * .95;)
ENDFIX
```

等式の結果である Sales と Expenses は密次元のメンバーで、オペランドである Actual は疎次元のメンバーです。Essbase では、密メンバー式は既存のデータ・ブロックに対してしか実行されないので、計算スクリプトによって必要なデータ・ブロックが作成されず、存在しないブロックに対して Budget データ値が計算されません。

次の方法を使用してこの問題を解決できます:

- 502 ページの「DATACOPY を使用した既存ブロックのコピー」
- 503 ページの「SET CREATENONMISSINGBLK を使用したすべての潜在的なブロックの計算」

DATACOPY を使用した既存ブロックのコピー

DATACOPY コマンドを使用して既存の各ブロックにブロックを作成し、新しいブロックで計算を実行します。例:

```
   DATACOPY Sales -> Actual TO Sales -> Budget;
DATACOPY Expenses -> Actual TO Expenses -> Budget;
FIX(Budget)
  (Sales = Sales -> Actual * 1.1;
```

```
  Expenses = Expenses -> Actual * .95;)
ENDFIX
```

Essbase によって、既存の対応する Actual ブロックごとに、Budget 値を含むブロックが作成されます。DATACOPY コマンドが終了すると、スクリプトの残りの部分によって値が変更されます。

次の場合に DATACOPY を使用することをお薦めします:

- 既存ブロックの値と、DATACOPY によって作成されたブロックの対応箇所の値との間に、数学的な関係がある場合。

  たとえば、前述の例では、Budget 値は既存の Actual 値に基づいて計算されます。

  注意 DATACOPY を実行すると、ソース・ブロックからのすべてのセルに、同じ値で新しいブロックが作成されます。ブロックの一部に対してのみ式を実行する場合、これらのコピーされたセルは計算終了時にそのまま残るので、結果として不適切な値となる可能性があります。

- コピーされるブロックに#MISSING 値のみを含むブロックがない場合。

  #MISSING 値が存在する場合、#MISSING 値のみを含むブロックが書き込まれます。不必要な#MISSING ブロックに、Essbase のリソースと処理時間が必要になります。

SET CREATENONMISSINGBLK を使用したすべての潜在的なブロックの計算

不適切な値が心配な場合は、DATACOPY のかわりに、SET CREATENONMISSINGBLK ON 計算コマンドを使用できます。このコマンドでは、すべての潜在的なブロックがメモリーで計算され、データ値を含む計算されたブロックのみが保管されます。SET CREATENONMISSINGBLK 計算コマンドは、密次元または疎次元で値を計算するときに便利です。

次の例では、既存の実績データから売上高予算と支出データを作成します。Sales と Expenses はメジャー密次元のメンバーで、Budget と Actual はシナリオ疎次元のメンバーです。

```
   FIX(Budget)
SET CREATENONMISSINGBLK ON
  (Sales = Sales -> Actual * 1.1;
  Expenses = Expenses -> Actual * .95;)
ENDFIX
```

注： SET CREATEBLOCKONEQ ON が疎次元に対して設定されている場合、SET CREATENONMISSINGBLK OFF コマンドが検出されるか、計算スクリプトが完了するまで、SET CREATENONMISSINGBLK ON によって一時的に SET CREATEBLOCKONEQ ON が上書きされます。974 ページの「疎次元のメンバーに割り当てられた非定数値」を参照してください。

SET CREATENONMISSINGBLK コマンドを使用する場合の長所は、密メンバーに適用したときに、メンバー式の影響を受けるデータ・セルのみが保存される点です。短所は、メモリーに生成される潜在的なブロックが多すぎて、計算パフォーマンスに影響する可能性がある点です。このコマンドを使用するときには、FIX を使用して計算するブロックのスコープを制限するなど、潜在的なブロックの数を制限します。

パーティションでの計算スクリプトの使用

参照:

- 504 ページの「パーティション用の計算スクリプトの作成」
- 504 ページの「パーティションの計算順序の制御」

パーティション用の計算スクリプトの作成

パーティション・アプリケーションは、複数のサーバー、プロセッサまたはコンピュータにスパンできます。

アプリケーションをパーティション化して、各パーティションで個別に計算を実行すれば、計算パフォーマンスを大幅に改善できます。パーティションを使用するときには、次の点に注意します:

- データベース計算全体に対するパフォーマンスの影響を評価します。パフォーマンスを向上させるために次のことを実行できます:
  - 計算全体を再デザインし、リモート・データベース内の透過パーティションにあるリモート値を参照しないようにします。
  - リモート・データベース内の値を動的に計算します。

    460 ページの「パーティション内のデータの動的計算」を参照してください。
  - 該当する式が含まれているデータベース内の値を複製します。

    たとえば、東部地域の四半期ごとのデータを複製する場合は、第 1 四半期、第 2 四半期、第 3 四半期および第 4 四半期の値のみを複製し、親の年値をローカルで計算します。
- Essbase で参照される値を取得するときにその値が最新の値であることを確認してください。前に説明したオプション(再設計、動的計算または複製)のいずれかを選択するか、式を計算する前に参照されるデータベースを計算します。

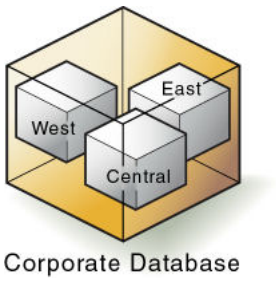
第 15 章「パーティション・アプリケーションの設計」と第 16 章「パーティションの作成および管理」を参照してください。

パーティションの計算順序の制御

Essbase で必要な結果を計算できるようにするには、データベースを特定の順序で計算する必要があります。

図 134 は、「West」、「Central」および「East」データベースの情報を「Corporate Database」から透過的に表示しているパーティションを例示しています。

図 134　パーティションの計算



「West」、「Central」および「East」には実績値のみが含まれています。「Corporate」には実績値と予算値が含まれています。「West」、「Central」および「East」のデータを「Corporate Database」で表示できますが、データが存在するのは「West」、「Central」および「East」データベースのみです。つまり、データは「Corporate Database」には複製されていません。

このため、Essbase で「Corporate」を計算するときには、「West」、「Central」および「East」から最新の値を取得する必要があります。必要な結果を得るには、「Corporate」を計算する前に「West」、「Central」および「East」を計算する必要があります。

計算スクリプトの保存、実行およびコピー

参照:

- 505 ページの「計算スクリプトの保存」
- 506 ページの「計算スクリプトの実行」
- 506 ページの「計算スクリプトのコピー」

計算スクリプトの保存

計算スクリプトは、次のいずれかの場所に保存できます:

- クライアント・コンピュータ上のファイルとして。
- Essbase サーバー上のアーティファクトとして。この場合、他のユーザーが計算スクリプトにアクセスできます。スクリプトと次のアーティファクトを関連付けることができます:
  - アプリケーションとそのアプリケーション内のすべてのデータベース。これにより、アプリケーション内の任意のデータベースに対して、スクリプトを実行できます。

    アプリケーションに関連付けられた計算スクリプトは、Essbase サーバー・コンピュータの `ARBORPATH/app/appname` ディレクトリに保存されます。
  - データベース。これにより、指定したデータベースに対してスクリプトを実行できます。

データベースに関連付けられた計算スクリプトは、Essbase サーバー・コンピュータの ARBORPATH/app/appname/dbname ディレクトリに保存されます。

786 ページの「Essbase を使用したアーティファクトの管理」を参照してください。

➤ 計算スクリプト・エディタを使用して計算スクリプトを保存する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「スクリプトの保存」を参照してください。

計算スクリプトの実行

Administration Services で計算スクリプトを実行するには、計算スクリプトを Essbase サーバー上でアーティファクトとして保存する必要があります。505 ページの「計算スクリプトの保存」を参照してください。

Administration Services を使用して計算スクリプトを実行するときには、計算をバックグラウンドで実行できるので、計算の処理中でも作業を続けることができます。バックグラウンド・プロセスのステータスをチェックして、いつ計算が完了したかを確認できます。Oracle Essbase Administration Services Online Help の「計算スクリプトの実行」を参照してください。

➤ 計算スクリプトを実行するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	計算スクリプトの実行	Oracle Essbase Administration Services Online Help
MaxL	**execute calculation**	Oracle Essbase Technical Reference
ESSCMD	RUNCALC	Oracle Essbase Technical Reference
Smart View	データの計算	Oracle Hyperion Smart View for Office User's Guide

計算スクリプトのコピー

計算スクリプトは、権限に応じて、Essbase サーバー上のアプリケーションとデータベースにコピーできます。アプリケーション移行の一部として、サーバー間でスクリプトをコピーすることもできます。

➤ 計算スクリプトをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	スクリプトのコピー	Oracle Essbase Administration Services Online Help
MaxL	**create calculation as**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYOBJECT	『Oracle Essbase テクニカル・リファレンス』

計算結果の確認

計算スクリプトの実行後に、Smart View で計算の結果を確認できます。

Essbase は、実行した計算スクリプトに関して次の情報を提供します:

- データベースの各パスに対する次元の計算順序
- 合計計算時間

詳細を表示するには、計算スクリプトで SET MSG SUMMARY、SET MSG DETAIL および SET NOTICE コマンドを使用します。482 ページの「データベース計算用のグローバル設定の指定」を参照してください。

これらのメッセージを使用することによって、計算方法を理解することができ、次回の計算で調整を行うことができます。

この情報が表示される場所は、計算スクリプトの実行に使用したツールによって異なります。

- Administration Services および Smart View - アプリケーション・ログ

  817 ページの「Essbase サーバー・ログおよびアプリケーション・ログの表示」を参照してください。
- MaxL - 標準出力(コマンドライン・ウィンドウ)

  情報の量は、MaxL シェルに設定されているメッセージ・レベルによって異なります。
- ESSCMD - ESSCMD ウィンドウまたは標準出力(コマンドライン・ウィンドウ)

  情報の量は、ESSCMD に設定されているメッセージ・レベルによって異なります。

計算に必要なディスク・サイズの判断

ESTIMATEFULLDBSIZE 構成設定を使用して、単一の CALC ALL ですべてのデータまたはデータの一部をロードするために必要なディスク・サイズを見積ることができます。

960 ページの「計算によるデータベース・サイズへの影響の見積り」を参照してください。

30 ブロック・ストレージ・データベース用の計算スクリプトの例の確認

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

これらの計算スクリプトの例について

この章のすべての例は、Sample.Basic データベースに基づいています。

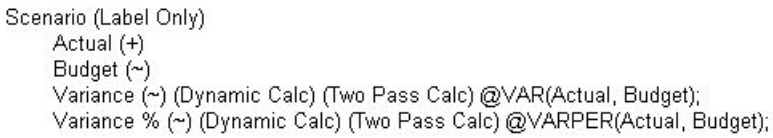
高機能計算コマンド SET UPDATECALC と SET CLEARUPDATESTATUS を使用する例は、433 ページの「SET CLEARUPDATESTATUS の使用例の確認」と 437 ページの「複数パス計算の例と解決策の確認」を参照してください。

差異の計算

次の例には、Budget 値と Actual 値の間の差異パーセンテージの計算が含まれています。

図 135 は、Variance および Variance %が動的計算、2 パス・メンバーとしてタグ付けされているアウトラインを示しています。

図 135　シナリオ次元の Variance および Variance %



デフォルト計算では、Essbase によって、市場次元と製品次元の値が集約されます。パーセンテージ値は正確に集約されないので、デフォルトの計算後に Variance %式を再計算する必要があります。

Variance %は、動的計算、2 パス・メンバーとしてタグ付けされているため、Essbase では、Variance %値の取得時にその値を動的に計算します。動的計算によって、正しくない値が正しく計算されたパーセンテージで上書きされます。

Variance %を動的計算、2 パス・メンバーとしてタグ付けしない場合は、次の計算スクリプトを使用して、デフォルト計算を実行し、Variance %の式を再計算します。この計算スクリプトでは、高機能計算がオン(デフォルト)になっていることを前提とします:

```
   CALC ALL;
SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
"Variance %";
```

Essbase では、次のアクションを実行します:

1. データベースのデフォルト計算を実行します(CALC ALL)。

   計算スクリプトを使用せずに、データベース・アウトラインのデフォルト計算を実行することもできます。

2. 高機能計算をオフにします(SET UPDATECALC OFF)。
3. 差異計算がデータベースの部分計算の場合でも、計算スクリプトの差異式によって計算されたブロックに消去のマークを付けます(CLEARUPDATESTATUS AFTER)。

   デフォルトでは、データ・ブロックは、データベースのフル計算が実行された後にのみ、消去としてマークされます。

4. データベースを循環して、Variance %の式を計算します。

983 ページの「2 パス計算タグまたは計算スクリプトの選択」と980 ページの「2 パス計算の使用」を参照してください。

統計的な差異の計算の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

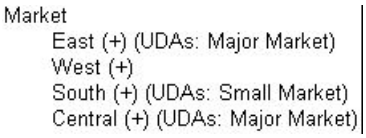
データベース・サブセットの計算

次の例は、地域のマーケティング・マネージャがデータベースの各領域を計算する方法を示しています。計算スクリプトでは、@DESCENDENTS(East)を使用して

計算を East 地域に限定し、East の各子の年次元、メジャー次元および製品次元を計算します。

図 136 は、市場次元の East、West、South および Central のメンバーのアウトラインを示しています:

図 136　市場次元の East、West、South および Central のメンバー



スクリプト例:

```
   /* Calculate the Budget data values for the descendants of East */

FIX(Budget, @DESCENDANTS(East))
 CALC DIM(Year, Measures, Product);
ENDFIX

/* Consolidate East */

FIX(Budget)
 @DESCENDANTS(East);
ENDFIX
```

Essbase では、次のアクションを実行します:

1. East の子孫の Budget 値を確定します。
2. East の子孫の各 Budget 値に対して、年次元、メジャー次元および製品次元をデータベースの 1 パスで計算します。
3. 他の次元のすべてのメンバーに対する Budget 値を確定します。
4. East の子孫を集約し、その結果を East に置きます。

新しい予算値のロード

次の例では、Budget 値を計算して、Variance と Variance %メンバーを再計算します。

スクリプト例:

```
   /* Calculate all Budget values */

FIX(Budget)
 CALC DIM(Year, Product, Market, Measures);
ENDFIX

/* Recalculate the Variance and Variance % formulas, which requires two passes */

Variance;
```

```
"Variance %";
```

Essbase では、次のアクションを実行します:

1. Budget 値を確定します。
2. すべての Budget 値を計算します。

   CALC DIM コマンドによって、Budget を含むシナリオ次元以外のすべての次元が計算されます。
3. データベース・アウトラインの Variance に適用される式を計算します。
4. データベース・アウトラインの Variance %に適用される式を計算します。

製品のシェア値と市場のシェア値の計算

次の例では、市場と製品ごとに製品のシェア値と市場のシェア値を計算します。
シェア値は次のように計算されます:

- 合計に対する各メンバーのパーセンテージ
- 親に対する各メンバーのパーセンテージ

次の 4 つのメンバーをメジャー次元に追加していることを前提とします:

- Market Share
- Product Share
- Market %
- Product %

スクリプト例:

```
   /* First consolidate the Sales values to ensure that they are accurate */

FIX(Sales)
 CALC DIM(Year, Market, Product);
ENDFIX

/* Calculate each market as a percentage of the total market for each product */

"Market Share" = Sales % Sales -> Market;

/* Calculate each product as a percentage of the total product for each market */

"Product Share" = Sales % Sales -> Product;

/* Calculate each market as a percentage of its parent for each product */

"Market %" = Sales % @PARENTVAL(Market, Sales);

/* Calculate each product as a percentage its parent for each market */
```

```
"Product %" = Sales % @PARENTVAL(Product, Sales);
```

Essbase では、次のアクションを実行します:

1. Sales 値を確定し、すべての Sales 値を集計します。

   CALC DIM コマンドを使用して、年次元、市場次元および製品次元を計算します。メジャー次元には Sales メンバーが含まれているので、集計されません。シナリオ次元はラベルのみなので、集計する必要はありません。
2. データベースを循環し、各月の各市場における各製品の Sales 値を取得し、この Sales 値を、各製品のすべての市場の合計 Sales のパーセンテージとして計算することで、Market Share を計算します(Sales -> Market)。
3. 各月の各市場における各製品の Sales 値を取得し、この Sales 値を、各市場のすべての製品の合計 Sales のパーセンテージとして計算することで、Product Share を計算します(Sales -> Product)。
4. 各月の各市場における各製品の Sales 値を取得し、この Sales 値を、市場次元の現在のメンバーの親の Sales 値のパーセンテージとして計算することで、Market %を計算します。

   @PARENTVAL 関数を使用して、市場次元の親の Sales 値を取得します。
5. 各月の各市場における各製品の Sales 値を取得し、この Sales 値を、製品次元の現在のメンバーの親の Sales 値のパーセンテージとして計算することで、Product %を計算します。

   @PARENTVAL 関数を使用して、製品次元の親の Sales 値を取得します。

製品をまたがるコストの割当て

次の例では、各月の各市場における各製品に間接費を割り当てます。間接費は、各製品の Sales 値に基づいて、すべての製品の合計 Sales のパーセンテージとして割り当てられます。

次の 2 つのメンバーをメジャー次元に追加していることを前提とします:

- OH_Costs(割り当てる間接費用)
- OH_TotalCost(合計間接費)

スクリプト例:

```
   /* Declare a temporary array called ALLOCQ based on the Year dimension */

ARRAY ALLOCQ[Year];

/* Turn the Aggregate Missing Values setting off. If this is your system default,
omit this line */

SET AGGMISSG OFF;

/* Allocate the overhead costs for Actual values */
```

```
FIX(Actual)
 OH_Costs (ALLOCQ=Sales/Sales->Product; OH_Costs =
 OH_TotalCost->Product * ALLOCQ;);

/* Calculate and consolidate the Measures dimension */

 CALC DIM(Measures);
ENDFIX
```

Essbase では次の計算を実行します:

1. 各メンバーの組合せについて、合計 Sales のパーセンテージとして Sales 値を一時的に保管するために、ALLOCQ という 1 次元の配列を作成します。

   ALLOCQ のサイズは、年次元のメンバー数によって決まります。

2. #MISSING 値はその親に集約されません(SET AGGMISSG OFF)。親レベルで保管されているデータ値は上書きされません。

   SET AGGMISSG OFF がシステムのデフォルトの場合、この行を省略します。989 ページの「#MISSING 値の集計」を参照してください。

   - Actual 値を確定します。
   - Actual に対するメンバーの組合せを循環して、OH_Costs を計算します。
   - 各月の各市場における各製品の Sales 値を取得して、その値を、各市場のすべての製品の合計 Sales のパーセンテージとして計算します(Sales -> Product)。その結果は ALLOCQ に配置されます。
   - すべての製品の合計間接費を取得して(OH_TotalCost -> Product)、その値に前の手順で ALLOCQ に配置された値を掛けます。その結果は OH_Costs に配置されます。

     両方の等式が丸カッコ( )で囲まれ、OH_Costs のメンバー: OH_Costs(equation1; equation2;)に関連付けられていることに注意してください。

3. メジャー次元を計算し、集計します。

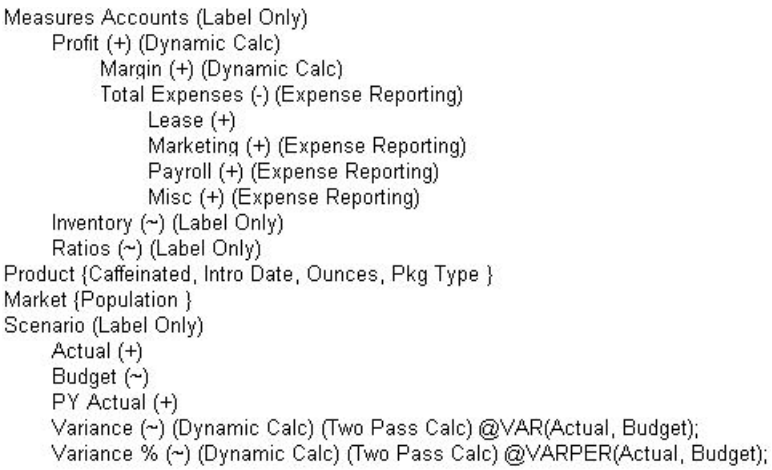
次元内での値の割当て

次の例では、@ALLOCATE 関数を使用して、2 つの製品の支出カテゴリに合計支出の予算を割り当てます。合計支出の予算は、前年の実績値に基づいて割り当てられます。

図 137 のアウトラインに示すように、次の変更が行われていることを前提とします:

- メジャー次元の Total Expenses の下に子 Lease が追加されている
- シナリオ次元に子 PY Actual が追加されている
- Total Expenses メンバーから「動的計算」タグが削除されている

図 137　変更されたメジャー次元とシナリオ次元



Colas と Root Beer について、それぞれデータ値 1000 と 2000 が Budget -> Total Expenses にロードされていることを前提とします。この値は、PY Actual の Total Expenses の欠落していない子に基づいて値を均等に分配して、各支出カテゴリに割り当てる必要があります。割当て値は、最も近いドルに丸める必要があります。

スクリプト例:

```
   /* Allocate budgeted total expenses based on prior year */

FIX("Total Expenses")
 Budget = @ALLOCATE(Budget->"Total Expenses",
 @CHILDREN("Total Expenses"),"PY Actual",,
 spread,SKIPMISSING,roundAmt,0,errorsToHigh)
ENDFIX
```

計算スクリプトの結果:

```
                Budget  PY Actual
Colas    Marketing    334 *   150
     Payroll     #MI    #MI
     Lease      333    200
     Misc       333    100

   Total Expenses

   1000

   450

Root Beer  Marketing    500    300
     Payroll     500    200
     Lease      500    200
     Misc       500    400

   Total Expenses

   2000
```

* 丸め誤差がこの値に追加されています。516 ページの手順 5 を参照してください。

➤ Essbase では、データベース内を循環して、次の計算が実行されます:

1 **Total Expenses の子を確定します。**

FIX ステートメントに@ALLOCATE を付けて使用すると、計算パフォーマンスが向上することがあります。

2 **Budget -> Colas -> Marketing の場合、各月の PY Actual -> Colas における、各支出カテゴリの欠落していない値の合計数で、1 を除算します。**

この場合は、Budget -> Colas に対する欠落していない支出値が 3 つなので、1 を 3 で除算します。

3 **手順 2 の値(.333)を取得し、その値に Budget -> Colas -> Total Expenses の値(1000)を掛け、最も近いドル(333)に丸めます。その結果は Budget -> Colas -> Marketing に配置されます。**

4 **Budget -> Colas の各支出カテゴリに対して手順 2 と手順 3 を繰り返し、次に Budget -> Root Beer に対して同じ手順を繰り返します。**

5 **計算スクリプトで指定されたとおりに、割当て値を最も近いドルの整数に丸めます。**

Essbase は、ブロックに 2 回目のパスを行い、丸められた値の合計を割当て値に等しくします(たとえば、Budget -> Colas -> Total Expenses の場合は 1000)。この例では、支出カテゴリの合計は 1000 ではなく 999 なので、Budget -> Colas -> Total Expenses に 1 の丸め誤差があります。すべての割当て値は同じ((333)なので、丸め誤差 1 が割当て範囲の最初の値 Budget -> Colas -> Marketing に追加されます(このため、値が 334 になります)。

複数の次元での値の割当て

次の例では、@MDALLOCATE 関数を使用して 3 つの次元に合計支出の予算のロードされた値を割り当てます。合計支出の予算は、前年の実績値に基づいて割り当てられます。

次の変更が行われていることを前提とします:

- シナリオ次元に子 PY Actual が追加されている
- Actual のデータが、PY Actual にコピーされている
- Budget のデータが消去されている

この例の場合、East の州全体で製品 100 の子の支出カテゴリに値 750(Budget -> Total Expenses -> Product -> East -> Jan)を割り当てる必要があります。割当てでは、PY Actual からの値を使用して、各カテゴリの比率を決定します。

スクリプト例:

```
   /* Allocate budgeted total expenses based on prior year, across 3 dimensions */

SET UPDATECALC OFF;
FIX (East, “100”, “Total Expenses”)
 BUDGET = @MDALLOCATE(750,3,@CHILDREN( “100” ),@CHILDREN("Total
Expenses"),@CHILDREN(East),"PY Actual",,share);
ENDFIX
```

PY Actual の値:

```
       Jan
    PY Actual
           Marketing   Payroll  Misc  Total Expenses
100-10  New York     94       51     0    145
    Massachusetts  23       31     1    55
    Florida      27       31     0    58
    Connecticut    40       31     0    71
    New Hampshire  15       31     1    47
100-20  New York     199      175    2    376
    Massachusetts  #MI      #MI    #MI   #MI
    Florida      #MI      #MI    #MI   #MI
    Connecticut    26       23     0    49
    New Hampshire  #MI      #MI    #MI   #MI
100-30  New York     #MI      #MI    #MI   #MI
    Massachusetts  26       23     0    49
    Florida      #MI      #MI    #MI   #MI
    Connecticut    #MI      #MI    #MI   #MI
    New Hampshire  #MI      #MI    #MI   #MI
100    New York     #MI      #MI    #MI   #MI
    Massachusetts  12       22     1    35
    Florida      12       22     1    35
    Connecticut    94       51     0    145
    New Hampshire  23       31     1    55
    East       237      220    3    460
```

➤ Essbase では、データベース内を循環して、次の計算が実行されます:

1 **East、100 の子および Total Expenses を確定します。**

   FIX ステートメントに@MDALLOCATE を付けて使用すると、計算パフォーマンスが向上することがあります。

2 **割当てを実行する前に、各製品と州の組合せについて、PY Actual での各支出カテゴリの比率を使用して、750 の各支出カテゴリへの配分比率(割り当てる値)を決定します。PY Actual -> 100-10 -> New York から始めて、Essbase では最初の支出カテゴリ Marketing の値を、PY Actual-> 100-10 -> East -> Total Expenses の値で割って、そのカテゴリの比率を計算します。**

   たとえば、Essbase で PY Actual -> 100-10 -> New York -> Marketing の値(94)を PY Actual -> 100-10 -> East -> Total Expenses の値(460)で割ると、Marketing カテゴリの比率(約 20.4%)が算出されます。

3 **各支出カテゴリ、各製品と州の組合せに対して、手順 2 を繰り返します。**

4 割当て時に Essbase では、手順 2 と手順 3 で計算した比率を使用して、各製品と州の組合せに対して、Budget の Total Expenses の各子への 750 の配分比率を決定します。

たとえば、Marketing の場合、Essbase では手順手順 2 で計算した 20.4%という数値を使用して、750 の 20.4%(約 153)を取得して、その割当て値を Budget -> 100-10 -> New York -> Marketing に配置します(この手順の下に記載されている結果を参照してください)。

5 手順 2 と手順 3 で計算した PY Actual での比率を使用して、各支出カテゴリおよび各製品と州の組合せに対して、手順 4 を繰り返します。

6 支出カテゴリを集計し、Total Expenses の値を算出します。

Budget の割当ての結果:

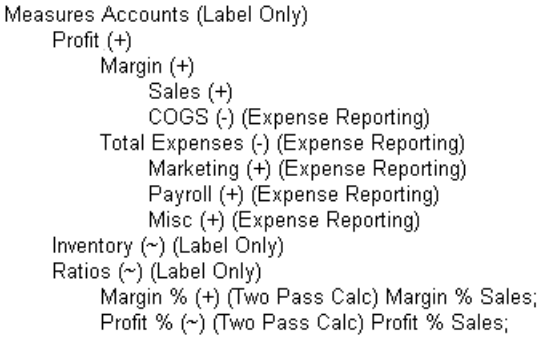
```
      Jan
    Budget
           Marketing   Payroll  Misc  Total Expenses
100-10  New York     153.26    83.15   0    236.41
    Massachusetts  37.50     50.54   1.63  89.67
    Florida      44.02     50.54   0    94.56
    Connecticut    65.22     50.54   0    115.76
    New Hampshire  24.26     50.54   1.63  76.63
100-20  New York     #MI       #MI     #MI   #MI
    Massachusetts  #MI       #MI     #MI   #MI
    Florida      42.39     37.50   0    79.89
    Connecticut    #MI       #MI     #MI   #MI
    New Hampshire  #MI       #MI     #MI   #MI
100-30  New York     #MI       #MI     #MI   #MI
    Massachusetts  #MI       #MI     #MI   #MI
    Florida      #MI       #MI     #MI   #MI
    Connecticut    #MI       #MI     #MI   #MI
    New Hampshire  19.57     35.87   1.63  57.07
100   New York     153.26    83.15   0    236.41
    Massachusetts  37.50     50.54   1.63  89.67
    Florida      86.41     88.04   0    174.46
    Connecticut    65.22     50.54   0    115.76
    New Hampshire  44.02     86.41   3.26  133.70
    East       386.41    358.70   4.89  750
```

LOOP コマンドを使用したゴールシーク計算

次の例は、特定の製品で特定の利益を得るために達成する必要がある売上高を計算する方法を示しています。この例では、計算スクリプトで、1 月の目標 15,000 Profit を達成するために Sales の Budget 値を調整します。

図 138 のアウトラインに示すように、「動的計算」のタグが付けられているメンバーがないこと、および(メジャー次元の Ratios の下にある) Profit per Ounce メンバーが計算に含まれていないことを前提とします。

図 138　メジャー次元



ゴールシーク計算スクリプトを実行する前のデータ値は次のようになっているとします:

```
   Product, Market, Budget  Jan
Profit          12,278.50
 Margin         30,195.50
  Sales        49,950.00
  COGS        19,755.00
 Total Expenses     17,917.00
  Marketing      3,515.00
  Payroll       14,402.00
  Misc        0
Inventory         Label Only member
Ratios          Label Only member
 Margin %        60.45
 Profit %        24.58
```

スクリプト例:

```
   /* Declare the temporary variables and set their initial values*/

VAR
 Target = 15000,
 AcceptableErrorPercent = .001,
 AcceptableError,
 PriorVar,
 PriorTar,
 PctNewVarChange = .10,
 CurTarDiff,
 Slope,
 Quit = 0,
  DependencyCheck,
  NxtVar;

/*Declare a temporary array variable called Rollback based on the Measures dimension
*/

ARRAY Rollback [Measures];

/* Fix on the appropriate member combinations and perform the goal-seeking
calculation*/
```

```
FIX(Budget, Jan, Product, Market)
 LOOP (35, Quit)
  Sales (Rollback = Budget;
  AcceptableError = Target * (AcceptableErrorPercent);
  PriorVar = Sales;
  PriorTar = Profit;
  Sales = Sales + PctNewVarChange * Sales;);
  CALC DIM(Measures);
  Sales (DependencyCheck = PriorVar - PriorTar;
  IF(DependencyCheck <> 0) CurTarDiff = Profit - Target;
   IF(@ABS(CurTarDiff) > @ABS(AcceptableError))
    Slope = (Profit - PriorTar) / (Sales - PriorVar);
    NxtVar = Sales - (CurTarDiff / Slope);
    PctNewVarChange = (NxtVar - Sales) / Sales;
   ELSE
    Quit = 1;
   ENDIF;
  ELSE
   Budget = Rollback;
   Quit = 1;
  ENDIF;);
 ENDLOOP
 CALC DIM(Measures);
ENDFIX
```

➤ Essbase では、次の計算を実行します:

1 **VAR コマンドを使用して必要な一時変数を宣言します。該当する場合は、初期値が設定されます。**

2 **Budget 値を保管するための Rollback という 1 次元配列を宣言します。**

Rollback のサイズは、メジャー次元のメンバー数によって決まります。

3 **Product と Market のすべてのメンバーに対して、Jan -> Budget の値を確定します。**

4 **LOOP と ENDLOOP の間のコマンドが、メンバーの組合せに対して 35 回循環されます。ただし、Quit 変数が 1 に設定された場合、LOOP は中断され、計算は ENDLOOP コマンドの後に続行されます。**

5 **メンバーの組合せを循環して、次の計算を実行します:**

1. Budget -> Sales 値を Rollback 一時配列変数に置きます。
2. Target 値(15000)と AcceptableErrorPercent 値(0.001)を乗算し、許容誤差を計算します。その結果は AcceptableError 変数に配置されます。
3. 現在の Sales 値を保持して、現在のメンバーの組合せの Sales 値を PriorVar 一時変数に置きます。
4. 現在の Profit 値を保持して、現在のメンバーの組合せの Profit 値を PriorTar 一時変数に置きます。
5. PctNewVarChange 値(0.1)と現在の Sales 値を乗算し、現在の Sales 値を加算して、新しい Sales 値を計算します。その結果は Sales に配置されます。
6. メジャー次元を計算し、集計します。

7. PriorVar 値から PriorTar 値を減算して、その結果を DependencyCheck 一時変数に置きます。
8. DependencyCheck が 0(ゼロ)でないことを確認します(IF)。
    - DependencyCheck が 0 でない場合、現在の Profit から Target 値(15000)を減算し、その結果を CurTarDiff 一時変数に置きます。

      IF コマンドによって、CurTarDiff の絶対値(+符号または-符号を無視した値)が AcceptableError の絶対値よりも大きいかどうかが検査されます:

        - AcceptableError より大きい場合、Slope、NxtVar および PctNewVarChange 一時変数を計算します。
        - AcceptableError 以下の場合、Quit の値を 1 に設定して、LOOP コマンドを中断します。計算は、ENDLOOP コマンドの後に続行されます。
    - DependencyCheck が 0 の場合、Rollback 配列の値を Budget に置きます。Essbase は Quit の値を 1 に設定して、LOOP コマンドを中断します。計算は、ENDLOOP コマンドの後に続行されます。

6 メジャー次元を計算し、集計します。

製品 100-10 の結果:

```
   Product, Market, Budget  Jan
Profit          15,000.00
 Margin         32,917.00
  Sales        52,671.50
  COGS        19,755.00
 Total Expenses     17,917.00
  Marketing      3,515.00
  Payroll       14,402.00
  Misc        0
Inventory         Label Only member
Ratios           Label Only member
 Margin %         28.47839913
 Profit %         62.49489762
```

将来値の予測

次の例では、線形回帰予測法を使用して、選択した前月の既知のデータ値から始まり、既知の値に基づく予測値へと続くトレンド(@TREND)、つまり線を作成します。また、既知のデータ値への適合度について、トレンドの結果を検査する方法を示します。この例では、計算スクリプトで、6 月から 12 月の売上高データを予測します。5 月までのデータが揃っていることを前提とします。

メジャー次元に子 ErrorLR が追加されていることを前提とします。この子には、適合度の結果が配置されます。

スクリプト例:

```
   Sales
(@TREND(@LIST(Jan,Mar,Apr),@LIST(1,3,4),,
 @RANGE(ErrorLR,@LIST(Jan,Mar,Apr)),
  @LIST(6,7,8,9,10,11,12),
   Jun:Dec,LR););
```

表 89 に、予測計算スクリプトで使用されるパラメータを示します:

表 89　将来値を予測するための計算スクリプト例で使用されるパラメータ

パラメータ	説明
@LIST(Jan,Mar,Apr)	`Ylist`、つまり既知のデータ値を含むメンバーを表します。 @LIST 関数は、3 つのメンバーをカンマ区切りのリストとしてグループ化し、そのリストを他のパラメータとは別に保管します。
@LIST(1,3,4)	`Xlist`、つまり基になる変数値を表します。2 月と 5 月はスキップされるので、Essbase では `Ylist` 値を 1、3、4 とします。
,	`Xlist` パラメータの後ろの追加のカンマは、パラメータ(`weightList`)がスキップされていることを示します。 この例では、デフォルトの加重 1 を使用します。
@RANGE(ErrorLR, @LIST(Jan,Mar,Apr)	`errorList`、つまり `Ylist` に対するトレンド線の適合度の結果が配置されるメンバー・リストを示します。 `errorList` に配置される値は、`Ylist` のデータ・ポイントと作成されたトレンド線上のデータ・ポイントの間の差異です。 @RANGE 関数では、ErrorLR メンバーと `Ylist`(Jan, Mar, Apr)を結合して、メンバー・リストを作成します。
@LIST(6,7,8,9,10,11, 12)	`XforecastList`、つまり予測をシークする基となる変数値を表します。この例では、6 月から 12 月の値を連続して予測します。このため、値は 6、7、8、9、10、11、12 となります。
Jun:Dec	`YforecastList`、つまり予測値が配置されるメンバー・リストを表します。この例では、1 月、3 月および 4 月の値に基づいて 6 月から 12 月の値を予測します。
LR	線形回帰法を指定します。

➤ Essbase では、データベース内を循環して、次の計算が実行されます:

1. `Ylist` と `Xlist` パラメータで指定されたとおりに、トレンドで基になる既知のデータ値(Jan、Mar、Apr の Sales)を検索します。
2. `YforecastList` パラメータで指定されたとおりに、線形回帰を使用してトレンド線を計算し、その結果を Jun から Dec の Sales に置きます。
3. Jan、Mar および Apr のデータ値に対するトレンド線の適合度を計算し、その結果をそれぞれの月の ErrorLR に置きます。

   たとえば、1 月の ErrorLR 値(4.57)は、Essbase によってトレンド線が計算された結果、1 月の Sales 値(2339)とトレンド線上の 1 月の値の間の差異が 4.57 であることを意味します。2 月と 5 月は、`Ylist` に含まれていないので、その ErrorLR 値は#MISSING です。

計算スクリプトの結果:

```
   100 West Actual
   Sales   ErrorLR
Jan  2339   4.57
Feb  2298   #MI
Mar  2313   -13.71
Apr  2332   9.14
May  2351   #MI
Jun  2315.14  #MI
Jul  2311.29  #MI
Aug  2307.49  #MI
Sep  2303.57  #MI
Oct  2299.71  #MI
Nov  2295.86  #MI
Dec  2292   #MI
```

31 カスタム定義計算マクロの作成

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

カスタム定義マクロの理解

カスタム定義マクロを使用すると、複数の計算関数を単一の関数に結合できます。

カスタム定義マクロを開発してテストするときには、テスト・アプリケーション内にローカルに新しいマクロを作成してテストします。カスタム定義マクロは、テスト・アプリケーションでのテストが完了し、本番環境で使用できる状態になった後にのみ、グローバルに登録します。

カスタム定義マクロを作成および管理するには、データベース・マネージャ以上のセキュリティ・レベルが必要です。

カスタム定義マクロの命名

カスタム定義マクロに名前を付けるときには次のガイドラインに従います:

- マクロ名は、たとえば@MYMACRO のように、@記号で始めます。名前の残りの部分には、文字、数字および記号(@、#、$、_)を使用します。スペースはマクロ名に使用しないでください。
- 他のマクロによってのみ呼び出されるマクロの場合、マクロ名を@_で始めて、一般用途のマクロおよび関数と区別します。

- マクロには一意の名前を付けます。また、マクロ名は、カスタム定義関数の名前、および既存の計算関数の名前とも異なる必要があります。

  注： アプリケーションにグローバル・マクロと同じ名前のローカル・マクロが含まれている場合、ローカル・マクロが優先され計算に使用されます。

- ローカル・マクロの場合、マクロ名の先頭にアプリケーション名を付けて、アプリケーション名とマクロ名をピリオドで区切る必要があります:

  ```
  AppName .@ MacroName
  ```

  例:

  ```
  Sample.@MYMACRO
  ```

- グローバル・マクロは、そのマクロが作成された Essbase サーバー上で実行されているすべてのアプリケーションで使用できるので、アプリケーション名を割り当てる必要はありません。

カスタム定義マクロの作成

カスタム定義マクロを作成すると、Essbase によってマクロ定義が記録され、マクロのカタログに保管されます。マクロは、カタログから削除されるまでの間、式と計算スクリプトで使用できます。

カスタム定義マクロは次の方法で登録します:

- ローカルとして。この場合、マクロが作成された Essbase アプリケーションでのみそのマクロを使用できます
- グローバルとして。この場合、マクロが作成された Essbase サーバー上で実行されているすべての Essbase アプリケーションでそのマクロを使用できます

➤ カスタム定義マクロを作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**カスタム定義マクロの作成**	Oracle Essbase Administration Services Online Help
MaxL	**create macro**	『Oracle Essbase テクニカル・リファレンス』

次の MaxL ステートメントによって、Sample アプリケーションで使用するための @COUNTRANGE という名前のローカル・マクロが作成されます:

```
   create macro Sample.'@COUNTRANGE'(Any) AS
'@COUNT(SKIPMISSING, @RANGE(@@S))'
spec '@COUNTRANGE(MemberRange)'
comment 'counts all non-missing values';
```

次の MaxL ステートメントによって、@COUNTRANGE という名前のグローバル・マクロが作成されます:

```
   create macro'@COUNTRANGE'(Any) AS
'@COUNT(SKIPMISSING, @RANGE(@@S))'
spec '@COUNTRANGE(MemberRange)'
comment 'counts all non-missing values';
```

カスタム定義マクロの使用

カスタム定義マクロは、計算スクリプトまたは式でネイティブ計算コマンドと同じように使用できます。

➤ カスタム定義マクロを使用するには:

1 **計算スクリプトまたは式を作成するか、既存の計算スクリプトまたは式を開きます。**

- ローカルに登録されている場合、マクロが作成されたアプリケーション内で計算スクリプトまたは式を使用する必要があります。
- グローバルに登録されている場合、Essbase サーバー上のすべてのアプリケーションですべての計算スクリプトまたは式を使用できます。

2 **カスタム定義マクロを計算スクリプトまたは式に追加します。**

たとえば、この章で作成した@COUNTRANGE カスタム定義マクロを使用するには、次の計算スクリプトを作成します:

```
CountMbr = @COUNTRANGE(Sales, Jan:Dec);
```

Sample.Basic データベースでこの計算スクリプトを使用するか、「Sales, Jan:Dec」をテスト・データベースのメンバーの範囲に置き換えます。

3 **計算スクリプトまたは式を保存したら、通常どおりに実行します。**

カスタム定義マクロの表示

カスタム定義マクロを表示して、マクロが問題なく作成されたかどうか、またはカスタム定義マクロが、ローカルまたはグローバルのどちらであるかを判断します。

➤ カスタム定義マクロを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	カスタム定義マクロの表示	Oracle Essbase Administration Services Online Help
MaxL	**display macro**	『Oracle Essbase テクニカル・リファレンス』

カスタム定義マクロの更新

➤ カスタム定義マクロを更新するには:

1 マクロがローカルとグローバルのどちらで登録されているかを確認します。

527 ページの「カスタム定義マクロの表示」を参照してください。

2 マクロの定義を更新するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	カスタム定義マクロの編集	Oracle Essbase Administration Services Online Help
MaxL	**create macro** または **replace macro**	『Oracle Essbase テクニカル・リファレンス』

次の MaxL ステートメントによって、Sample アプリケーションでのみ使用されているローカル・マクロ@COUNTRANGE が変更されます:

```
   create or replace macro Sample.'@COUNTRANGE'(Any)
as '@COUNT(SKIPMISSING, @RANGE(@@S))';
```

次の MaxL ステートメントによって、グローバル・マクロ@COUNTRANGE が変更されます:

```
   create or replace macro '@COUNTRANGE'(Any)
as '@COUNT(SKIPMISSING, @RANGE(@@S))';
```

カスタム定義マクロのコピー

カスタム定義マクロは、適切なアクセス権がある任意の Essbase サーバーとアプリケーションにコピーできます。

➤ カスタム定義マクロをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	カスタム定義マクロのコピー	Oracle Essbase Administration Services Online Help
MaxL	**create macro**	『Oracle Essbase テクニカル・リファレンス』

カスタム定義マクロの削除

グローバル・カスタム定義マクロを削除する手順は、ローカル・カスタム定義マクロを削除する手順よりも複雑なので、DBA によってのみ実行する必要があります。

➤ カスタム定義マクロを削除するには:

1 マクロがローカルとグローバルのどちらで登録されているかを確認します。

527 ページの「カスタム定義マクロの表示」を参照してください。

2 カスタム定義マクロを使用している計算スクリプトまたは式がないことを確認します。

3 マクロをマクロのカタログから削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	カスタム定義マクロの削除	Oracle Essbase Administration Services Online Help
MaxL	**drop macro**	『Oracle Essbase テクニカル・リファレンス』

4 そのマクロが関連付けられていたすべてのアプリケーションを再起動します。

529 ページの「カスタム定義マクロのカタログのリフレッシュ」を参照してください。

次の MaxL ステートメントによって、Sample アプリケーションでのみ使用されているローカル・マクロ@COUNTRANGE が削除されます:

```
drop macro Sample.'@COUNTRANGE';
```

次の MaxL ステートメントによって、グローバル・マクロ@COUNTRANGE が削除されます:

```
drop macro '@COUNTRANGE';
```

カスタム定義マクロのカタログのリフレッシュ

マクロを追加、更新または削除するときに、カスタム定義マクロのカタログをリフレッシュします。

➤ サーバー上のすべてのアプリケーションのカスタム定義マクロのカタログをリフレッシュするには、サーバーを再起動します。

➤ 1 つのアプリケーションのカスタム定義マクロのカタログをリフレッシュするには、**refresh custom definitions** MaxL ステートメントを使用します。

次の MaxL ステートメントによって、Sample アプリケーションのカスタム定義マクロのカタログがリフレッシュされます:

```
refresh custom definition on application sample;
```

32 カスタム定義計算関数の作成

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

カスタム定義関数を作成するためのプロセス

Essbase に用意されている計算関数を強化するために、計算スクリプトで使用するカスタム定義関数を作成できます。Essbase には、カスタム定義関数のサンプルは用意されていません。

カスタム定義計算関数は Java で記述します。カスタム定義計算関数は Java で記述します。Essbase には、Java のクラスとアーカイブを作成するためのツールがありません。このため、サポートされるバージョンの JDK が必要です。Oracle Hyperion Enterprise Performance Management System 動作保証マトリックス(http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html)を参照してください。

カスタム定義関数の例は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

➤ カスタム定義関数を作成するには:

1 **カスタム定義関数の要件を確認します。**

532 ページの「カスタム定義関数の要件」を参照してください。

2 **カスタム定義関数として使用するための public かつ static なメソッドを少なくとも 1 つ含む、public な Java クラスを記述します。**

533 ページの「Java クラスの作成とコンパイル」を参照してください。

3 **Java クラスをインストールします。**

534 ページの「Essbase サーバーへの Java クラスのインストール」を参照してください。

4 **カスタム定義関数を、ローカル関数またはグローバル関数として登録します。**

535 ページの「カスタム定義関数の登録」を参照してください。

カスタム定義関数の要件

カスタム定義関数として使用するために、クラスに複数のメソッドを作成できます。通常は、Essbase サーバー上のすべてのアプリケーションでカスタム定義関数として使用するメソッドを単一のクラス内に作成することをお薦めします。ただし、Essbase サーバー上の一部のアプリケーションで使用するカスタム定義関数を追加する場合は、そのカスタム定義関数を個別のクラスに作成し、個別の `.jar` ファイルに保管して Essbase サーバーに追加します。

カスタム定義関数として使用するメソッドを含んだ Java クラスを複数作成する場合は、各クラス名が一意であることを確認します。重複するクラス名があると、重複するクラスのメソッドが認識されず、そのメソッドをカスタム定義関数として登録できません。

テスト・プログラムを Java で使用して、Java のクラスとメソッドをテストします。メソッドの出力が適切な場合は、そのメソッドを Essbase サーバーにインストールして、単一のテスト・アプリケーションにそのメソッドを登録します。テストするために関数をグローバルに登録しないでください。グローバルに登録すると、問題が発生したときにその関数を更新することがより難しくなります。

カスタム定義関数のメソッドは、次のようなサポートされているデータ型のいずれの組合せでも、入力パラメータとして持つことができます:

- boolean
- byte
- char
- com.hyperion.essbase.calculator.CalcBoolean
- float、double
- java.lang.String
- short、int、long
- これらのいずれかの型の配列

CalcBoolean は Essbase 固有のデータ型で、3 つの値(TRUE、FALSE および #MISSING)を保管できます。他のリストされたデータ型の詳細は、JDK のドキュメンテーションを参照してください。

メソッドの戻りデータ型は、void または前述のデータ型のいずれかにできます。戻されたデータ型は Essbase 固有のデータ型に変換されます。文字列は string 型にマッピングされます。ブール値は CalcBoolean データ型にマッピングされます。その他のすべての値は double 型にマッピングされます。

注： Essbase では、無限値または非数(Not-a-Number)値とともに戻される double 型の変数をサポートしません。このような値が Java プログラムから戻された場合、その値は Essbase で正しく記録されていないか、表示されていない可能性があります。Essbase に戻される前に、double 型の変数に、無限値と非数値があるかどうかを検査し、有限値に設定する必要があります。JDK のドキュメンテーションのクラス Double に関する説明を参照してください。

カスタム定義関数を作成、削除および管理するために、Essbase では次のセキュリティ権限が必要です:

- ローカル、アプリケーション全体、カスタム定義関数: アプリケーション・マネージャ以上
- グローバル、サーバー全体、カスタム定義関数: 管理者

カスタム定義関数を Essbase に登録するときには、関数に名前を付けます。その名前は、計算スクリプトと式で使用され、関数によって使用される Java のクラス名とメソッド名とは異なります。

カスタム定義関数に名前を付けるときには次の要件に従います:

- 名前は@記号で始めます。関数名の残りの部分には、文字、数字および記号(@、#、$、_)を使用します。スペースは関数名に使用できません。

  例: @MYFUNCTION

- カスタム定義マクロによってのみ呼び出されるカスタム定義関数の名前は、一般用途の関数およびマクロと区別するため、@_で始めます。

  例: @_MYFUNCTION

- カスタム定義関数の名前は一意でなければなりません。関数名は、他の関数名、カスタム定義マクロの名前および既存の計算関数の名前とも異なる必要があります。
- Essbase アプリケーションにグローバル関数と同じ名前のローカル関数がある場合、計算にはローカル関数が使用されます。

Java クラスの作成とコンパイル

Java クラスを作成およびコンパイルするには、テキスト・エディタと JDK `javac` ツールを使用します。

➤ カスタム定義関数用の Java クラスを作成するには:

1 **テキスト・エディタで Java クラスを作成します。**

例:

```
    public class CalcFunc {
public static double sum (double[] data) {
 int i, n = data.length;
 double sum = 0.0d;
 for (i=0; i<n; i++) {
  double d = data [i];
  sum = sum + d;
 }
 return sum;
}
}
```

2 ファイルを.java 拡張子で保存します。

例:

```
CalcFunc.java
```

3 .java ファイルが保管されているディレクトリに移動します。コマンド・プロンプトで、次のコマンドを入力します:

```
javac
java_filename
```

例:

```
javac CalcFunc.java
```

4 コンパイラによって新しいファイルが.class 拡張子で作成されるまでに、コンパイル・エラーを修正します。

例:

```
CalcFunc.class
```

Essbase サーバーへの Java クラスのインストール

Java クラスは、JDK jar ツールを使用して JAR ファイルにコンパイルする必要があります。

➤ .jar ファイルを作成して Essbase サーバーにインストールするには:

1 .class ファイルが保管されているディレクトリに移動します。コマンド・プロンプトで、次のコマンドを入力します:

```
jar cf
```

```
jar_filename

class_filename
```

例:

```
jar cf CalcFunc.jar CalcFunc.class
```

2 **Essbase サーバーを実行しているコンピュータで、.jar ファイルを次のいずれかのディレクトリにコピーします(ディレクトリが存在しない場合は、作成します):**

- グローバル・カスタム定義関数を含んでいる.jar ファイル:

```
ARBORPATH
/java/udf/
```

- 特定のアプリケーションでのみ使用する.jar ファイル:

```
ARBORPATH
/app/
appName
/udf/
```

appName は、ローカル・カスタム定義関数が使用されるアプリケーションの名前です。

後で.jar ファイルを別の場所に移動した場合は、CLASSPATH 変数を変更して、.jar ファイルへのフルパスとファイル名を含める必要があります。

3 **特定のアプリケーションでのみ関数を使用する場合は、Essbase でそのアプリケーションを再起動します。それ以外の場合は、Essbase サーバーを再起動します。**

カスタム定義関数の登録

カスタム定義関数用の Java クラスを.jar ファイルにコンパイルし、その.jar ファイルを Essbase サーバーにインストールしたら、カスタム定義関数を登録して、計算スクリプトと式で使用できるようにします。532 ページの「カスタム定義関数の要件」を参照してください。

グローバル・カスタム定義関数を登録すると、Essbase サーバー上のすべての Essbase アプリケーションでその関数を使用できます。カスタム定義関数をグローバル関数にする前に、単一のアプリケーションでその関数をテスト(およびそのアプリケーションでのみその関数を登録)します。

マクロのカタログを更新するプロセスと同じプロセスを使用して関数のカタログを更新します。529 ページの「カスタム定義マクロのカタログのリフレッシュ」を参照してください。

注意 テストするためにグローバル関数を登録しないでください。登録すると、問題が発生した場合に変更がより難しくなります。

➤ カスタム定義関数を登録するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**カスタム定義関数の作成**	Oracle Essbase Administration Services Online Help
MaxL	**create function**	『Oracle Essbase テクニカル・リファレンス』

ローカル・スコープでカスタム定義関数を登録する場合は、アプリケーション名を接頭辞として追加します。たとえば、次の MaxL ステートメントによって、CalcFunc クラスのカスタム定義関数@JSUM が、Sample アプリケーションでのみ使用するためのローカル関数として登録されます:

```
   create function Sample.'@JSUM'
as 'CalcFunc.sum'
spec '@JSUM(memberRange)'
comment 'adds list of input members';
```

グローバル・スコープでカスタム定義関数を登録する場合は、アプリケーション名を接頭辞として追加しません。たとえば、次の MaxL ステートメントによって、CalcFunc クラスのカスタム定義関数@JSUM が、Essbase サーバー上のすべてのアプリケーションで使用するためのグローバル関数として登録されます:

```
   create function '@JSUM'
as 'CalcFunc.sum'
spec '@JSUM(memberRange)'
comment 'adds list of input members';
```

注: Java メソッドの入力パラメータの指定はオプションです。入力パラメータを指定しなかった場合、Essbase では、Java コードのメソッド定義から入力パラメータを読み取ります。ただし、複数のカスタム定義関数を同じメソッド名と異なるパラメータ・セットを使用して登録する場合は、関数のバージョンごとにパラメータを指定して、関数のバージョンごとに個別に登録する必要があります。

登録したカスタム定義関数の使用

登録したカスタム定義関数はネイティブの Essbase 計算コマンドと同じように使用できます。

➤ 登録したカスタム定義関数を使用するには:

1 **計算スクリプトまたは式を作成するか、既存の計算スクリプトまたは式を開きます。**

- カスタム定義関数をローカルに(特定のアプリケーション内に)登録した場合には、そのアプリケーション内の計算スクリプトや式を使用する必要があります。
- カスタム定義関数をグローバルに登録した場合には、Essbase サーバー上のすべての計算スクリプトまたは式を使用できます。

2 **カスタム定義関数を計算スクリプトまたは式に追加します。**

たとえば、JSUM を使用するには、次の計算スクリプトを使用します:

```
"New York" = @JSUM(@LIST(2.3, 4.5, 6.6, 1000.34));
```

この計算スクリプトを Sample.Basic サンプル・データベースで使用するか、"New York"をテスト・データベースにあるメンバーの名前に置換します。

3 **計算スクリプトまたは式を保存したら、通常どおりに実行します。**

カスタム定義関数の更新

カスタム定義関数を更新する手順は、次の条件によって異なります:

- 関数がローカルとグローバルのどちらで登録されているか
- カスタム定義関数のシグネチャ(クラス名、メソッド名または入力パラメータ)がカスタム定義関数の Java コードで変更されているかどうか

通常、カスタム定義関数を更新するには、その関数のコードを含んでいる.jar ファイルを置き換えて、その関数を再登録する必要があります。ただし、カスタム定義関数のシグネチャが変更されておらず、関数に 1 セットの入力パラメータしかない(オーバーロード・メソッドでない)場合は、関数を含んでいる.jar ファイルを置き換えることができます。

注: DBA のみによってグローバル・カスタム定義関数が更新されるようにしてください。

➤ カスタム定義関数を更新するには:

1 **関数がローカルとグローバルのどちらであるかを確認します。**

538 ページの「カスタム定義関数の表示」を参照してください。

2 **カスタム定義関数用の Java クラスに変更を加え、Java テスト・プログラムを使用してその出力をテストします。**

3 **Java クラスをコンパイルして、前の.jar ファイルと同じ名前を使用して、新しい.jar ファイルにアーカイブします。**

前の.jar ファイルに含まれていたカスタム定義関数用の他のクラスとメソッドも含めます。

4 **更新するカスタム定義関数の種類(ローカルまたはグローバル)に応じて、次のアクションを実行します:**

1. ローカル: .jar ファイルの関数を使用するすべての Essbase アプリケーションをシャットダウンします。

2. グローバル: すべての Essbase アプリケーションをシャットダウンします

どの Essbase アプリケーションが .jar ファイルのどの関数を使用するか不明な場合は、すべての Essbase アプリケーションをシャットダウンします。

5 新しい .jar ファイルを Essbase サーバーにコピーして、既存の .jar ファイルを同じ名前で置き換えます。

6 カスタム定義関数のシグネチャが変更されていない場合は、手順 8 に進みます。

7 カスタム定義関数を置き換えるには、次のツールを使用します:

- Administration Services: Oracle Essbase Administration Services Online Help の「カスタム定義関数の編集」を参照してください。
- MaxL: **create or replace function** ステートメントを使用します。例:
  - ローカル:

```
create or replace function sample.'@JSUM'
as 'CalcFunc.sum';
```

  - グローバル:

```
create or replace function '@JSUM'
as 'CalcFunc.sum';
```

8 シャットダウンしたアプリケーションを再起動して、カタログを更新します。

カスタム定義関数の表示

カスタム定義関数を表示して、関数が正常に登録されているかどうか、およびローカルとグローバルのどちらで登録されているのかを確認できます。カスタム定義関数は、作成して登録するまで表示されません。

➤ カスタム定義関数を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	カスタム定義関数の表示	Oracle Essbase Administration Services Online Help
MaxL	**display function**	『Oracle Essbase テクニカル・リファレンス』

たとえば、Sample アプリケーションに含まれているカスタム定義関数、およびすべての登録済グローバル関数を表示するには、次の MaxL ステートメントを使用します:

```
display function Sample;
```

display function ステートメントによって、アプリケーション名なしでグローバル関数がリストされ、その関数がグローバルであることが示されます。アプリケー

ションにグローバル関数と同じ名前の関数が含まれている場合は、ローカル関数のみがリストされます。

カスタム定義関数の削除

カスタム定義関数を削除するには、次の権限が必要です:

- ローカル: アプリケーションに対するアプリケーション・マネージャ権限、またはそれよりも範囲が広い権限
- グローバル: 管理者権限

カスタム定義関数を削除する前に、その関数を使用している計算スクリプトまたは式がないことを確認します。グローバル・カスタム定義関数は、Essbase サーバー全体の計算スクリプトと式で使用できるので、削除する前にこの関数を使用している Essbase サーバー上の計算スクリプトまたは式がないことを確認する必要があります。

注意 Essbase データベースにアクセスしているユーザーがおらず、計算ルーチンが実行されていないときにのみ、グローバル・カスタム定義関数を削除してください。

➤ カスタム定義関数を削除するには:

1 **関数がローカルとグローバルのどちらであるかを確認します。**

538 ページの「カスタム定義関数の表示」を参照してください。

2 **削除するカスタム定義関数の種類(ローカルまたはカスタム)に応じて、次のアクションを実行します:**

1. ローカル: `.jar` ファイルの関数を使用するすべての Essbase アプリケーションをシャットダウンします。
2. グローバル: すべての Essbase アプリケーションをシャットダウンします。

3 **カスタム定義関数を削除するには、次のツールを使用します:**

- Administration Services: Oracle Essbase Administration Services Online Help の、カスタム定義関数の削除に関する項を参照してください
- MaxL: **drop function** MaxL ステートメントを使用します。例:
- ローカル:

```
drop function Sample.'@JSUM';
```

- グローバル:

```
drop function '@JSUM';
```

4 **シャットダウンしたアプリケーションを再起動して、カタログを更新します。**

カスタム定義関数のコピー

カスタム定義関数は、適切なアクセス権がある任意の Essbase サーバーとアプリケーションにコピーできます。

カスタム定義関数をコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**カスタム定義関数のコピー**	Oracle Essbase Administration Services Online Help
MaxL	**create function**	『Oracle Essbase テクニカル・リファレンス』

カスタム定義関数のパフォーマンスの考慮事項

カスタム定義関数は、Essbase 計算機フレームワークの拡張機能として実装されるので、カスタム定義関数の動作の効率は、Essbase 計算機フレームワークにネイティブの関数よりも低いと考えられます。

パフォーマンスを最適化するため、特に計算スピードが重要となるアプリケーションにおいては、カスタム定義関数の使用は、ネイティブの Essbase 計算コマンドで実行できない計算のみに限定してください。

カスタム定義関数のメモリーの考慮事項

JVM API と Java API for XML Parsing の使用は、Essbase を実行するために必要なメモリーに最初に影響します。これらの追加コンポーネントを実行するために必要なメモリーについては、Essbase のメモリー要件で説明しています。Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

この起動メモリー要件を越えて、カスタム定義関数用に開発する Java プログラムに追加のメモリーが必要なことがあります。起動時に、Win32 オペレーティング・システム用の JVM によって、2MB のメモリーがプログラムにすぐに割り当てられます。この割当ては、JVM によって実行されるプログラムの要件に従って増加します。Win32 オペレーティング・システム上の JVM に割り当てられるメモリーのデフォルトの上限値は、64MB です。Java プログラムの実行によって、JVM のデフォルトのメモリー割当ての上限値を超えた場合、JVM によってエラーが生成されます。JVM メモリーの管理および他のオペレーティング・システムのメモリー割当ての詳細は、JDK のドキュメンテーションを参照してください。

JVM のデフォルトのメモリー要件およびサーバーを実行するハードウェアの制限を考慮して、メモリーの使用を注意深く観察します。特に、カスタム定義関数の開発者は、カスタム定義関数に大きいアーティファクトを作成するときに、JVM のメモリー制限値を超えないように注意する必要があります。

第 V 部
データの取得

データの取得の内容：

33 レポート・スクリプトの基本の理解

この章の内容

この章のすべてのレポート・スクリプトの例は、Sample.Basic データベースに基づいています。

単純なレポート・スクリプトの処理

ページ、行および列に使用する次元を、レポート・コマンドで宣言しメンバーを選択すると、単純なレポート・スクリプトのすべての要素が作成されます。

次のレポート・スクリプト・プロセスの例は、要素、次元およびメンバー選択コマンドを指定するサンプル・スクリプトを手順ごとに示します。このスクリプトには、スクリプトの動作を文書で説明するコメントおよび!出力コマンドが含まれます。このスクリプトは、Sample.Basic データベースに基づいています。

1. レポート・スクリプトを作成します。

   Oracle Essbase Administration Services Online Help の「スクリプトの作成」を参照してください。

2. 次の情報をレポート・スクリプトに入力します。

```
        // This is a simple report script example
// Define the dimensions to list on the current page, as below
<PAGE (Market, Measures)

// Define the dimensions to list across the page, as below
<COLUMN (Year, Scenario)

// Define the dimensions to list down the page, as below
```

```
<ROW (Product)

// Select the members to include in the report
Sales
<ICHILDREN Market
Qtr1 Qtr2
Actual Budget Variance
<ICHILDREN Product

// Finish with a bang
!
```

3. レポート・スクリプトを保存します。

   Oracle Essbase Administration Services Online Help の「スクリプトの保存」を参照してください。

4. レポート・スクリプトを実行します。

   Oracle Essbase Administration Services Online Help の「レポート・スクリプトの実行」を参照してください。

スクリプト例を Sample.Basic データベースに対して実行すると、次のレポートが生成されます:

```
                  East Sales

          Qtr1            Qtr2
      Actual   Budget  Variance   Actual   Budget  Variance
     ========  ========  ========  ========  ========  ========
100      9,211   6,500   2,711   10,069   6,900   3,169
200      6,542   3,700   2,842   6,697   3,700   2,997
300      6,483   4,500   1,983   6,956   5,200   1,756
400      4,725   2,800    ,925   4,956   3,200   1,756
 Product  26,961   17,500   9,461   28,678   19,000   9,678

                West Sales

          Qtr1            Qtr2
      Actual   Budget  Variance   Actual   Budget  Variance
     ========  ========  ========  ========  ========  ========
100      7,660   5,900   1,760   7,942   6,500   1,442
200      8,278   6,100   2,178   8,524   6,200   2,324
300      8,599   6,800   1,799   9,583   7,600   1,983
400      8,403   5,200   3,203   8,888   6,300   2,588
 Product  32,940   24,000   8,940   34,937   26,600   8,337

                South Sales

          Qtr1            Qtr2
      Actual   Budget  Variance   Actual   Budget  Variance
     ========  ========  ========  ========  ========  ========
100      5,940   4,100   1,840   6,294   4,900   1,394
200      5,354   3,400   1,954   5,535   4,000   1,535
300      4,639   4,000    639   4,570   3,800    770
400    #MISSING  MISSING  #MISSING  #MISSING  #MISSING  #MISSING
```

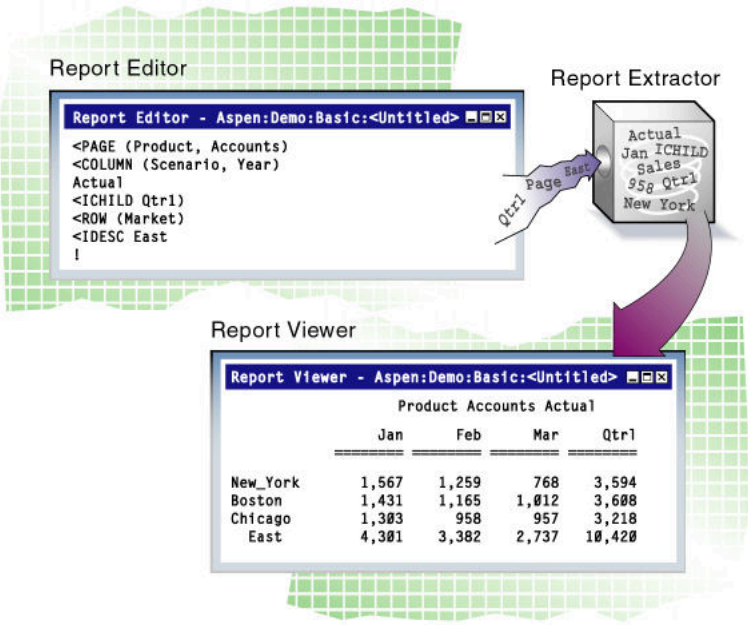
```
 Product   15,933   11,500   4,433   16,399   12,700   3,699

                 Central Sales

           Qtr1             Qtr2
      Actual   Budget  Variance   Actual   Budget  Variance
     ========  ========  ========  ========  ========  ========
100       9,246    ,500   2,746   9,974   7,300   2,674
200       7,269   6,800    469   7,440   7,000    440
300      10,405   6,200   4,205  10,784   6,800   3,984
400      10,664   5,200   5,464  11,201   5,800   5,401
 Product   37,584   24,700   12,884   39,399   26,900   12,499

                 Market Sales

           Qtr1           Qtr2
      Actual   Budget  Variance   Actual   Budget  Variance
     ========  ========  ========  ========  ========  ========
100      32,057   23,000   9,057   34,279   25,600   8,679
200      27,443   20,000   7,443   28,196   20,900   7,296
300      30,126   21,500   8,626   31,893   23,400   8,493
400      23,792   13,200  10,592   25,045   15,300   9,745
 Product  113,418   77,700   35,718  119,413   85,200   34,21
```

レポート・ライターの動作の理解

レポート・ライターは次の 3 つの主なコンポーネントで構成されています:

- レポート・スクリプト・エディタは、レポート・スクリプトを記述するときに使用するテキスト・エディタです。レポート・コマンドでは、フォーマットされたレポートを定義し、データベースからデータのサブセットをエクスポートし、フリーフォームのレポートを生成します。保存済のスクリプトを実行すると、レポートが生成されます。保存済のレポート・スクリプトのファイル拡張子は.rep です。
- レポート・エクストラクタは、レポート・スクリプトの実行時に Essbase データベースからデータ情報を取得します。
- レポート・ビューアは、完全なレポートを表示します。保存済のレポートのファイル拡張子は.rpt です。

図 139　レポート・ライターのコンポーネント



レポート・エクストラクタ

➤ レポート・エクストラクタでは、次のアクションを実行して、レポート・スクリプトを処理し、データを取得します:

1 可能なすべてのメンバーの組合せに基づいて、メンバー・リストを作成します。たとえば、<IDESCENDANTS East によって、メンバー East とそのすべての子孫が取得されます。

2 メンバーの制限を適用します。たとえば、<LINK によってメンバーの選択が限定されます。

3 メンバーの出力の順序を決めます。たとえば、<SORT によって、メンバーのソート順が決まります。

4 次の領域からデータが抽出されます:

- ローカル領域
- パーティション化された領域
- 動的に計算されるデータ

5 データを制限します。たとえば、次のコマンドによって、欠落した値のみで構成されるすべての行の表示が抑止されます:

```
{SUPMISSINGROWS}
```

6 データをソートします。たとえば、<TOP によって、指定したデータ列で一番大きい値を持つ行が戻されます。

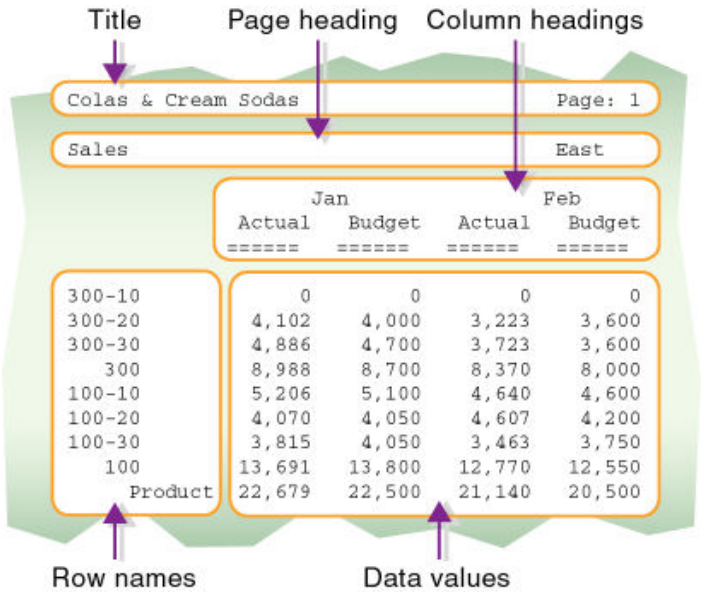
7 出力をフォーマットします。たとえば、{SKIP}によって、最終の出力レポートで 1 つ以上の行がスキップされます。

レポート・エクストラクタによるデータの取得順序は、複雑な抽出とフォーマット・コマンドの実行に影響します。たとえば、レポート・エクストラクタでは、データを制限(手順 5)してから、データをソート(手順 6)するので、条件付き取得コマンドを間違った順序で配置した場合、予期しないレポートの出力結果となる可能性があります。レポート・スクリプトを設計するときには、データ取得プロセスに注意してください。

レポートのパート

レポートについてプランし、それを設計する場合、レポートのパートを理解することが不可欠です。

図 140　典型的なレポートの要素



典型的なレポートは、次のパートで構成されています:

- ページ見出しは、現在のページに表示される次元をリストします。ページのすべてのデータ値には、ページ見出しの次元が共通のプロパティとして適用されています。

```
<PAGE (Market, Measures)
```

- 列見出しは、ページ上のメンバーをリストします。複数の次元からのデータをレポートする列を定義できます。この場合、ネストされた列見出しとなります。

```
<COLUMN (Year, Scenario)
```

- 行見出しは、ページの下方向にメンバーをリストします。1 つの次元内の複数のレベルからの行を含むメンバー・リストまたは複数の次元からの行を含むメンバー・リストを定義できます。行は、次元名の下にインデント表示されます。

```
<ROW (Product)
```

- タイトルには、ユーザー定義のテキスト、日付、タイム・スタンプ、レポートを実行しているユーザーの名前、ページ番号、ソース・データベースの名前またはその他の説明情報が表示されます。タイトルは、ユーザーが生成し、省略可能です。ページ、列および行の見出しは、レポート・ページ上のデータを明確に説明するために必要なので、自動的に生成されます。

```
     { STARTHEADING
TEXT  1 "Prepared by:"
   14 "*USERNAME"
    C "The Electronics Club"
   65 "*PAGESTRING"
TEXT  65 "*DATE"
SKIP
ENDHEADING }
```

- データ値は、データベース・セルに含まれている値です。データ値は、メンバーの組合せの検索結果、またはレポート・エクストラクタでのレポート実行時の計算結果です。各データ値は、ページ見出し、列見出しおよび行名のメンバーの組合せです。

  行のすべてのデータ値は、その行の行名のプロパティを共有します。レポートにはゼロ個以上の行名次元を使用できます。各行名次元は、行名の列を生成します。最も内側の行名の列が最も速く循環します。

レポート・スクリプトのパート

レポート・スクリプトは、一連のレポート・ライター・コマンドで構成され、感嘆符(!)レポート出力コマンドで終了します。

レポート・スクリプト・ファイルに 1 つ以上のレポート・スクリプトを入力できます。レポート・スクリプト・ファイルは、レポート・スクリプト・エディタまたは任意のテキスト・エディタで作成するテキスト・ファイルです。

レポート・スクリプトを作成するには、レイアウト、メンバーの選択およびフォーマットを定義するコマンドをレポート・スクリプト・エディタで入力または選択します。スクリプトのそれぞれの要素は、読みやすくするために色分けされます。構文のオートコンプリートを使用可能にして、スクリプトをすばやく作成できます。

レポート・ライターのコマンドでは、データ抽出およびフォーマットの 2 つの処理が実行されます:

- 抽出コマンドは、データベースから抽出された生データの選択、向き、グループ化および配列を処理します。小なり記号(<)から始まるコマンドです。
- フォーマット・コマンドでは、レポートのフォーマットと表示形式のカスタマイズ、新しい列の作成、列と行の計算を行うことができます。このコマンドは、通常、中カッコ({ })で囲まれています。ただし、一部のコマンドは小なり記号(<)で始まります。

- 感嘆符(!)は、一連のコマンドを終了し、データベースからの情報を要求します。同じレポート・ファイル内に、1 つ以上のレポート・スクリプトを、それぞれの!コマンドで終了させて配置することができます。

レポート・コマンドの詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

レポートについてのプランニング

レポートの設計は、情報の提示にとって重要です。適切な要素を入れ、優れた読みやすいレイアウトに情報を配置します。

➤ レポートをプランニングするには:

1 **レポートで求められる要件と、レポートの生成に必要な期間を考慮します。**

2 **レポートの下絵を大まかに作成します。次のアイテムを含めます:**

- レイアウト
- 列数
- メンバー
- 必要な場合はタイトル
- データ値のフォーマット

3 **下絵を確認します。ほとんどの場合、この段階で、追加のデータまたはフォーマットが必要かどうかがわかります。**

4 **レポートの実行時間を最適化する方法を決めます。**

レポート・スクリプトを最適化する方法の詳細は、第 59 章「レポートおよび他の取得機能の最適化」を参照してください。

注: レポートをプランニングするときには、できるだけ行名を数字で表さないようにしてください。あいまいにならないように、行に説明的な名前を付けてください。

セキュリティと複数ユーザーの問題に関するガイドライン

レポート・スクリプト・エディタを使用して、レポート・スクリプトを作成または変更するときには、Administration Services を使用する必要があります。任意のテキスト・エディタを使用して、スクリプト・ファイルを作成できます。レポート・スクリプト・エディタを使用した場合、デスクトップ・コンピュータだけでなく、Essbase サーバーに保管されているレポート・スクリプトも作成および変更できます。サーバーに保管されているレポート・スクリプトを変更するには、アプリケーション・マネージャまたはデータベース・マネージャのアクセス権が必要です。

Essbase では、複数のユーザーによるデータベースへの同時アクセスをサポートします。ほとんどの複数ユーザー環境と同様、Essbase では、重要なデータをセキュリティ・システムで保護します。ユーザーは、正しい権限がある場合にのみデータの読取りまたは更新ができます。

レポート・スクリプトを実行すると、Essbase のセキュリティによって、レポートの指定されたすべてのデータ・メンバーに対して読取り以上のアクセス・レベルがユーザーにあることが確認されます。メンバーをスプレッドシートに取り込むプロセスと同じフィルタ・プロセスでは、Essbase によって、ユーザーの権限が足りないメンバーが出力から除外されます。

データのレポートのみを行うユーザーには、他のユーザーによって行われたロックは透過的です。ユーザーがロックしていて、レポートに必要なデータの一部を更新している場合でも、ロックがレポートを妨げることはありません。レポートのデータは、レポートを実行した時点でのデータベースのデータを表します。同じレポートを後で実行すると、前回のレポート実行後に行われた変更が反映されます。

Essbase セキュリティ・システムの詳細は、第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」を参照してください。

レポート・スクリプト作成プロセスの確認

この項では、レポート・スクリプトを作成するプロセスについて説明します。

1. レポート・スクリプトを作成します。

   550 ページの「レポート・スクリプトの作成」を参照してください。

2. レポート・スクリプトの構文を確認します。

   Oracle Essbase Administration Services Online Help の「スクリプト構文の検査」を参照してください。

3. レポート・スクリプトを保存します。

   551 ページの「レポート・スクリプトの保存」を参照してください。

4. レポート・スクリプトを実行します。

   551 ページの「レポート・スクリプトの実行」を参照してください。

5. 必要に応じて、レポートを保存します。

   Oracle Essbase Administration Services Online Help の「レポートの保存」を参照してください。

レポート・スクリプトの作成

データベースのデータに関してレポートするには、次のいずれかの方法を使用します:

- レポート・スクリプト・エディタ。多次元データの多数のページで構成される大規模なレポートを作成するには、レポート・スクリプト・エディタを使用します。この規模のレポートは、ほとんどの場合、最も強力なスプレッドシートの機能も超える可能性があります。レポート・ライター・コマンドを

使用して、フォーマット済レポートの定義、Essbase データベースからのデータのサブセットのエクスポートおよびフリーフォーム・レポートの作成を行うことができます。Oracle Essbase Administration Services Online Help の「スクリプトの作成」を参照してください。

- テキスト・エディタ。
- スプレッドシート。フリーフォーム・モードまたはテンプレート取得モードでスプレッドシートのレポート・コマンドを使用します。
- Essbase API。592 ページの「C、Visual Basic およびグリッド API を使用したレポートの生成」および『Oracle Essbase API リファレンス』を参照してください。
- サードパーティ製レポート生成ツール。

Administration Services でのレポート・スクリプトの作成と編集の詳細は、Oracle Essbase Administration Services Online Help の「レポート・スクリプト・エディタについて」を参照してください。

レポート・スクリプトの保存

次のいずれかとして、レポート・スクリプトを保存できます:

- クライアント・コンピュータおよびネットワーク上のファイル。
- Essbase サーバー上のアーティファクト。他のユーザーがレポート・スクリプトにアクセスできるようにするには、そのスクリプトを Essbase サーバーに保存します。スクリプト・アーティファクトと次のアーティファクトを関連付けることができます:
  - アプリケーションとそのアプリケーション内のすべてのデータベース。これにより、アプリケーション内の任意のデータベースに対して、スクリプトを実行できます。
  - データベース。これにより、指定したデータベースに対してスクリプトを実行できます。

レポート・スクリプトには、デフォルトで.rep 拡張子が付いています。レポート・スクリプトを Administration Services で実行する場合には、そのスクリプトに.rep 拡張子が必要です。

➤ レポート・スクリプト・エディタを使用してレポート・スクリプトを保存するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「レポート・スクリプトの保存」を参照してください。

レポート・スクリプトの実行

Administration Services を使用してレポート・スクリプトを実行すると、その結果を「レポート・ビューア」ウィンドウ、プリンタまたはファイル(あるいは 3 つすべて)に送信できます。「レポート・ビューア」ウィンドウでは、レポートを印刷、保存およびコピーできます。

Administration Services を使用すると、レポートをバックグラウンドで実行できるので、レポートの処理中でも作業を続けることができます。バックグラウンド・プロセスのステータスを確認して、レポートが完了した時間を確認できます。

Oracle Essbase Administration Services Online Help の「レポート・スクリプトの実行」を参照してください。

レポート・スクリプトのコピー

レポート・スクリプトは、権限に応じて、Essbase サーバー上のアプリケーションとデータベースにコピーできます。アプリケーション移行の一部として、サーバー間でスクリプトをコピーすることもできます。

➤ レポート・スクリプトをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	スクリプトのコピー	Oracle Essbase Administration Services Online Help
MaxL	**alter object**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYOBJECT	『Oracle Essbase テクニカル・リファレンス』

フリーフォーム・レポートの開発

フリーフォーム・レポートは、通常、構造化レポートを作成するよりも簡単に作成できます。フリーフォームのレポート形式は、「レポート・スクリプト・エディタ」ウィンドウでのアド・ホック・レポートに最適です。

フリーフォーム・レポートでは、PAGE コマンド、COLUMN コマンドまたは ROW コマンドを使用しません。そのかわりに、レポートを実行すると、一連の内部ルールから情報が集められて、レポート・エクストラクタによってレポート・スクリプトに適用されます。

次のスクリプトおよびレポートは、フリーフォーム・レポートの例です:

```
   Sales Colas
Jan Feb Mar
Actual Budget
Illinois
Ohio
Wisconsin
Missouri
Iowa
Colorado
{UCHARACTERS}
Central
   !
```

次のレポートが生成されます:

```
                Sales 100

        Jan         Feb         Mar
     Actual  Budget  Actual  Budget  Actual  Budget
    ======= ======= ======= ======= ======= =======
Illinois    829    700    898    700    932    700
Ohio      430    300    397    300    380    300
Wisconsin   490    300    518    400    535    400
Missouri    472    300    470    300    462    300
Iowa      161     0    162     0    162     0
Colorado    643    500    665    500    640    500
========    ===    ===    ===    ===    ===    ===
Central   3,025   2,100   3,110   2,200   3,111   2,200
```

フォーマット・コマンドを使用して、特定のフォーマットをフリーフォーム・レポートに追加できます。PAGE、COLUMN および ROW コマンドが省略されている場合、Essbase では、次のルールに従ってフリーフォーム・レポートをフォーマットします:

1. レポート・エクストラクタは、レポート指定に定義されている単一の次元の最後のメンバー(レポート出力演算子!の前)を見つけます。この次元がレポートの行次元になります。残りのすべての選択は、ルール 2 と 3 に定義されているように、ページ次元または列次元になります。
2. レポート・エクストラクタは、単一メンバー選択をサーチします。ルール 1 を満たしていない単一メンバーが見つかった場合、その次元はページ次元になります。
3. レポート・エクストラクタは、ルール 1 または 2 を満たしていない残りの次元メンバーをサーチします。残りのメンバーが見つかった場合、その次元は列次元になります。列次元は、フリーフォーム・スクリプトでの選択順にネストされます。
4. レポート・エクストラクタは、レポート指定で指定されていない次元について、データベース・アウトラインをサーチします。指定されていない次元が見つかった場合、その次元はページ次元(ルール 2 に定義されている、単一メンバー選択のデフォルト)になります。
5. ある次元から 1 つ以上の連続するメンバーの後に続く選択が、その次元に対する事前のどの選択にも優先します。

たとえば、次のレポートは、California、Oregon、Washington、Utah、Nevada および West が Market のメンバーとして認識されます。

```
   Sales
Jan Feb Mar
Actual Budget
Apr May Jun
California
Oregon
Washington
Utah
Nevada
{UCHARACTERS}
```

```
West
  !
```

レポート・エクストラクタは、このレポートにフリーフォームのフォーマット・ルールを適用します:

1. California、Oregon、Washington、Utah、Nevada および West は最後にリストされているので、レポート・エクストラクタは、これらを ROW(Market)が指定されている場合と同様に扱います(ルール 1)。
2. Sales はメジャー次元からの単一メンバー選択です。レポート・エクストラクタは、このメンバーを PAGE(Measures)が指定されている場合と同様に扱います(ルール 2)。
3. レポート・エクストラクタは、残りのメンバーをサーチすると、年次元とシナリオ次元のメンバーを検出します。このメンバーを COLUMN(Year, Scenario)として扱います(ルール 3)。
4. レポート・エクストラクタは、データベース・アウトラインをサーチし、製品次元がレポート指定で指定されていないことを検出します。製品は単一メンバー選択なので、レポート・エクストラクタはこのメンバーを PAGE(Product)が指定されている場合と同様に扱います(ルール 4)。
5. 最後に、レポート・エクストラクタは、Apr May Jun が Jan Feb Mar と同じ次元に含まれ、スクリプトの後続の行に表示されていることを検出します。レポート・エクストラクタは、最初の指定(Jan Feb Mar)を破棄して、2 番目の指定(Apr May Jun)を使用します。

次のレポートが生成されます:

```
                Product Sales

          Actual          Budget
       Apr   May   Jun   Apr   May   Jun
      ======= ====== ====== ====== ====== ======
California   3,814  4,031  4,319  3,000  3,400  3,700
Oregon     1,736  1,688  1,675  1,100  1,000  1,100
Washington   1,868  1,908  1,924  1,500  1,600  1,700
Utah      1,449  1,416  1,445   900   800   800
Nevada     2,442  2,541  2,681  1,900  2,000  2,100
======     =====  =====  =====  =====  =====  =====
 West    11,309 11,584 12,044  8,400  8,800  9,400
```

注： フリーフォーム・モードでは、代替変数を使用できません。

34 レポート・スクリプトの作成

この章の内容

この章のすべてのレポート・スクリプトの例は、Sample.Basic データベースに基づいています。

関連項目:

- 第 33 章「レポート・スクリプトの基本の理解」
- 『Oracle Essbase テクニカル・リファレンス』(レポート・コマンドの構文と多数のレポート・スクリプトの例)

抽出コマンドの理解

ページ、列および行に対するメンバーの組合せを指定する抽出コマンドを使用して、レポートを作成します。

抽出コマンドは、データベースから抽出された生データ・レコードの選択、向き、グループ化および配列を指定します。これらのコマンドは、次元名かメンバー名のいずれか、またはキーワードに基づいており、名前は大なり記号(>)で始まります。

抽出コマンドは、その抽出コマンドが出現した行からレポートの最後まで、レポートに適用されます。別の抽出コマンドがレポート内の後続の行で出現する場合、その抽出コマンドは前のコマンドを上書きします。

フォーマット・コマンドの理解

フォーマット・コマンドを使用して、レポートの外観を指定したり、データ値の表示を制御できます。フォーマットされたデータ値は、抽出コマンドとレポート・コマンドの組合せに基づいて、スクリプトの実行時にレポートに表示されます。

フォーマット・コマンドでは、レポートのフォーマットと表示形式をカスタマイズし、レポート時間の計算を作成できます。フォーマット・コマンドは、通常、中カッコ({ })で囲まれています。ただし、一部のフォーマット・コマンドは小なり記号(<)で始まります。

フォーマット・コマンドはレポート・スクリプト内にグローバルに適用されるか、メンバーに固有であるかのどちらかです。

レポート・スクリプトの構文の理解

レポートを作成するには、レイアウト、メンバー選択およびフォーマットを定義するコマンドをレポート・スクリプト・エディタで入力します。スクリプトのそれぞれの要素は、読みやすくするために色分けされます。入力時にオートコンプリートを使用可能にして、スクリプトを対話式に作成できます。Oracle Essbase Administration Services Online Help の「レポート・スクリプト・エディタについて」を参照してください。

レポート・スクリプトを書く場合、次のガイドラインに従います:

- 読みやすくするために、コマンドを 1 つ以上のスペース、タブまたは新しい行で区切ります。レポート処理は、余分な空白行、スペース、タブの影響を受けません。
- コマンドは大文字または小文字で入力します。コマンドでは大文字と小文字が区別されません。データベース・アウトラインで大文字と小文字が区別される場合、レポート・スクリプトのメンバーはそのアウトラインに合わせる必要があります。
- レポート処理を開始するには、感嘆符(!)レポート出力コマンドまたは 1 つ以上の連続する数値を入力します。同じレポート・ファイル内に、1 つ以上のレポート・スクリプトを、それぞれの!コマンドで終了させて配置することができます。
- 1 組の中カッコ内で複数のフォーマット・コマンドをグループ化できます。たとえば、次のフォーマットは同じです:

```
     {UDATA SKIP}
{UDATA} {SKIP}
```

- 次のケースでは、メンバー名を引用符で囲みます:
  - アンパサンドで始まる名前(たとえば、"&Product")。
  - スペースを含む名前(たとえば、"Cost of Goods Sold")。
  - 単語 Default を含む名前(たとえば、"Default Value")。
  - 重複するメンバー名。修飾メンバー名として入力する必要があります(たとえば、"[2006].[Qtr1]")。

- 名前の先頭に 1 つ以上の数字を含む名前(たとえば、"100-Blue")。
- 名前に、表 90 にリストされる文字が含まれる場合:

表 90　メンバー名を囲む必要のある文字

文字	説明
*	アスタリスク
@	アット・マーク
{}	中カッコ
[ ]	大カッコ
,	カンマ
:	コロン
-	ダッシュ、ハイフンまたはマイナス符号
<	小なり記号
()	丸カッコ
+	プラス符号
;	セミコロン
/	スラッシュ

- フォーマット・コマンドの前にアンダースコア、等号またはハイフンが 3 つ以上あると、レポート・エクストラクタでは、これらの文字が無関係の下線付き文字であるとみなされ、無視されます。たとえば、===\{SKIP 1\}などです。
- //(ダブル・スラッシュ)を使用してコメントを記述します。コメントに続く行の内容はすべて、レポート・ライターでは無視されます。コメントの各行はダブル・スラッシュで始める必要があります。これによって、複数の行で構成されるコメントを追加できます。
- コマンド名を短縮するときは注意してください。コマンド名は同じ文字で始まることが多いので、確実に特定できる短縮形を使用しないと、予期しない結果が生じることがあります。

ページ・レイアウトの設計

レポートは、多次元データの 2 次元の表示です。ページ・レイアウト・コマンドを使用して、ネストした列または行のグループとして定義された追加の次元をページに取り込んだり、追加のページをレポートに取り込んだりできます。

ページ・レイアウトは、ページの列と行を構成する見出しで構成されます。ページ、行および列データの抽出コマンドを特定のメンバー選択と組み合せて、レポートの基本レイアウトを定義します。

ページ・レイアウトの各構成要素は、次の異なるフォーマット・コマンドを持ちます:

```
   <PAGE
<COLUMN
<ROW
```

さらに、<ASYM と<SYM コマンドでは、列次元メンバー・リストのデフォルトの解釈方法を上書きして、非対称か対称のレポート・フォーマットを生成します。

562 ページの「ページ、列および行の見出しのフォーマット」を参照してください。

ページ、列および行の見出しの作成

➤ レポートの見出しを定義するには:

1 `<PAGE(dimensionName, dimensionName)` を入力します

`dimensionName` には、現在のページに表示される次元をリストします。ページのすべてのデータ値には、ページ見出しの次元が共通のプロパティとして適用されています。例:

```
<PAGE (Measures, Market)
```

2 「[Enter]」キーを押します。

3 `<COLUMN(dimensionName)` を入力します

`dimensionName` には、ページにまたがって表示する各次元の名前を指定します。例:

```
<COLUMN (Year)
```

次元名を追加すると、ネスト列見出しが作成されます。

4 「[Enter]」キーを押します。

5 `<ROW(dimensionName)` を入力します

`dimensionName` には、ページの縦方向に表示する各次元の名前を指定します。例:

```
<ROW (Market)
```

6 「[Enter]」キーを押します。

注: 見出しコマンドに関連付ける追加のメンバーを選択できます。メンバー名を PAGE、COLUMN または ROW コマンドのパラメータとして使用すると、レポート・エクストラクタによって、そのメンバーが該当する次元に関連付けられます。

レポート・スクリプトの例:

```
   <PAGE (Product, Measures)
<COLUMN (Scenario, Year)
Actual
<ICHILDREN Qtr1
<ROW (Market)
<IDESCENDANTS East
  !
```

次のレポートが生成されます:

```
         Product Measures Actual

          Jan    Feb    Mar    Qtr1
       ======== ======== ======== ========
New York       512    601    543   1,656
Massachusetts     519    498    515   1,532
Florida       336    361    373   1,070
Connecticut      321    309    290    920
New Hampshire     44     74     84    202
 East       1,732   1,843   1,805   5,380
```

属性次元のメンバーを使用して、ページ、列および行見出しを作成できます。

レポート・スクリプトの例:

```
   <PAGE (Measures,Caffeinated)
Profit
<COLUMN (Year,Ounces)
Apr May
"12"
<ROW (Market,"Pkg Type")
Can
<ICHILDREN East
  !
```

次のレポートが生成されます:

```
        Profit Caffeinated 12 Scenario
             Apr    May
          ======== ========
New York     Can    276    295
Massachusetts  Can    397    434
Florida     Can    202    213
Connecticut    Can    107     98
New Hampshire   Can     27     31
 East      Can   1,009   1,071
```

見出しの変更

表 91 にリストされているコマンドを使用して、レポートの見出しを次のように変更できます:

表 91　レポート・ライター・コマンド: 見出し

レポート・コマンド	説明
STARTHEADING	デフォルトの見出しのかわりに、レポートの各ページの先頭または HEADING コマンドの直後に表示されるカスタム・ページ見出しを作成します。ENDHEADING コマンドを使用して、カスタム見出しの終わりを指定します。
HEADING	ページ見出し、すなわちデフォルトの見出しまたは STARTHEADING と ENDHEADING コマンドで定義された見出しのどちらかを表示します。 SUPHEADING コマンドが使用されていた場合、このコマンドを使用してページ見出しの表示を再度使用可能にできます。
IMMHEADING	見出し非表示コマンドが使用されている場合、次の非表示でないデータ行を待機せずに見出しがただちに強制表示されます。
COLHEADING	列ヘッダーの表示を自動的にオンにします。
PAGEHEADING	次のデータ出力行の前にページ見出しを表示します。

565 ページの「ページ、列および行のフォーマットの抑制」を参照してください。

対称型レポートと非対称型レポートの作成

Essbase レポートには、対称または非対称の列グループを入れることができます。Essbase では、選択したメンバーに基づいて列グループの対称性を自動的に決定します。

次のような対称型レポートでは、同じメンバーのグループが繰り戻されているのが特徴です。

```
         East                  West
  Budget      Actual       Budget     Actual
Q1  Q2  Q3    Q1  Q2  Q3     Q1  Q2  Q3   Q1  Q2  Q3
```

次のような非対称型レポートでは、ネストされたメンバーのグループの内容が異なっている(少なくとも 1 つのメンバーが異なる)のが特徴です。メンバーの数が異なる場合と、メンバーの名前が異なる場合があります。

```
         East             West
  Budget      Actual       Budget
Q1  Q2  Q3    Q1  Q2  Q3     Q1  Q2  Q3
```

デフォルトでは、Essbase では対称型レポートを作成します。ただし、すべての列次元に同じ数のメンバーを選択した場合を除きます。

非対称型レポートの例は、『Oracle Essbase テクニカル・リファレンス』の「レポート・スクリプトの例」ページにある「サンプル 13: 非対称列の作成」を参照してください。

Essbase による対称性の評価は、すべての順序付け、列の制限または計算済列の結果の適用よりも先に行われます。

デフォルトの列グループの置き換え

<SYM と<ASYM コマンドを使用して、レポート用に Essbase によって選択されるデフォルトの列グループを上書きできます。<SYM と<ASYM は、レポートでこのコマンドの後に続くメンバー選択コマンドに影響します。

1. 列メンバーの選択が非対称のルールの要件を満たしているが、対称型レポートを生成する場合に、<SYM コマンドを使用します。<SYM コマンドを使用すると、必ず各列次元のすべての組合せが作成され、対称型レポートが生成されます。
2. <ASYM コマンドを使用して、対称型フォーマットをオフにして、非対称型レポートのルールを復元します。

列見出しの変更

レポートの対称性ではなく、列見出しのみを変更する必要がある場合は、<PYRAMIDHEADERS と<BLOCKHEADERS フォーマット・コマンドが便利です。

- <BLOCKHEADERS フォーマット・コマンドを使用して、対称型レポートで使用されているピラミッド形式のヘッダーを、非対称型レポートで使用されているようなブロック形式のヘッダーに変更します。対称型レポートでは、デフォルトで列のレイアウトに<PYRAMIDHEADERS モードが使用されます。
- <PYRAMIDHEADERS フォーマット・コマンドを使用して、非対称型レポートで使用されているブロック形式のヘッダーを、対称型レポートで使用されているようなピラミッド形式のヘッダーに変更します。非対称型レポートでは、列のレイアウトに<BLOCKHEADERS モードが使用されます。

フォーマット

フォーマット・コマンドは、通常は、中カッコ({ })で囲まれ、最終レポートのデータとラベルのフォーマットを定義します。このコマンドは、グローバルかメンバー専用のいずれかに設定できます。

- グローバル・コマンドは、レポート・スクリプト・ファイル内で出現したときに実行されて、レポート・ファイルの終わりまで、または別のグローバル・コマンドに置き換わるまでその効果が持続します。

  たとえば、{SUPMISSINGROWS}コマンドでは、欠落した値のみを含むレポート・スクリプト・ファイル内のすべての行を抑制します。
- メンバー専用コマンドは、レポート・スクリプトで出現したときに実行され(通常は、レポート・スクリプトの次のメンバー)、そのメンバーにのみ適用されます。メンバーに添付されているフォーマットは、そのメンバーが処理される前に実行されます。

たとえば、{SKIP}コマンドを使用すると、レポート・スクリプトの行次元間で指定した行数がスキップされます。別の行で行をスキップする場合には、SKIPコマンドを再び使用する必要があります。

レポート・ページのフォーマット

複数のフォーマット・コマンドを使用して、最終レポートのページの外観を設計できます。参照:

- 562 ページの「ページの長さ、幅および中央揃えの設定」
- 562 ページの「改ページの挿入」

例は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

ページの長さ、幅および中央揃えの設定

表 92 にリストされているコマンドを使用して、レポート・スクリプトの次のページ仕様を設定できます:

表 92　レポート・ライター・コマンド: ページ長さ、幅および中央揃え

レポート・コマンド	説明
WIDTH	列の幅を指定します。
LMARGIN	左余白を設定します。
SETCENTER	ページの中央を設定します。

改ページの挿入

レポート・スクリプトには、表 93 にリストされているコマンドを使用して、次のタイプの改ページを設定できます:

表 93　レポート・ライター・コマンド: 改行

レポート・コマンド	説明
PAGELENGTH	各ページの行数を設定します。
NEWPAGE	現在のページに生成された行数とは無関係に改ページを強制します。
PAGEONDIMENSION	コマンド内のメンバーと同じ次元のメンバーが、レポートのある行から次の行に変更されるたびに改ページを挿入します。この機能をオフにするには、NOPAGEONDIMENSIONコマンドを使用します。
FEEDON	あるページの行数が現在の PAGELENGTH 設定より大きくなるときに、レポートの改ページを使用可能にします。

ページ、列および行の見出しのフォーマット

列と行のフォーマット・コマンドにより、特殊タイプのフォーマット設定コマンドを構成します。

列フォーマットの設定

列フォーマット・コマンドの指定は、目的のフォーマットに従って、適用する列の前または後ろに置くことができます。

たとえば、Sample.Basic データベースに基づいている次のスクリプトでは、最初の{DECIMAL}コマンドは 2 つの列(Actual と Budget)が設定された後に処理されます。ただし、この{DECIMAL}コマンドは、まだ存在しない列 3 を参照しています。Essbase では、レポートを 3 つの列に動的に拡大することでこのコマンドに対応します。レポートの指定が 6 つの列に拡大すると、{DECIMAL}のフォーマットは列 3 と 6(および 3 のすべての倍数)に適用されます。

Essbase では、別の次元が追加されたときに、前の列次元グループの繰返しが発生し、フォーマットも同様に繰り返すという前提で、このパターン拡張を実行します。2 番目の{DECIMAL}フォーマット・コマンドは、6 つの列の作成後に出現するので、列 1 と 4 にのみ適用されます。

レポート・スクリプトの例:

```
   <PAGE (Measures, Market)
Texas Sales
  <COLUMN (Scenario, Year)
  Actual Budget
{DECIMAL 2 3 }
  Jan Feb Mar
{DECIMAL 1 1 4 }
<ROW (Product)
<DESCENDANTS "100"
  !
```

次のレポートが生成されます:

```
               Sales Texas

        Actual          Budget
     Jan   Feb   Mar   Jan   Feb   Mar
     ===   ===   ===   ===   ===   ===
100-10  452.0   465  467.00  560.0   580  580.00
100-20  190.0   190  193.00  230.0   230  240.00
100-30 #MISSING #MISSING #MISSING #MISSING #MISSING #MISSING
```

次のスクリプトは、同じ結果が得られる列フォーマットの 2 つの方法を示します。最初のスクリプトでは、最初の 2 つの{DECIMAL}コマンドは、{DECIMAL}コマンドの処理後に、Jan Feb が出現したときにフォーマットを配布することで、1 番目と 3 番目の列をフォーマットするように位置付けられています。

レポート・スクリプトの例: フォーマットの配布による列のフォーマット

```
   California Sales
  <COLUMN (Scenario, Year)
  Actual Budget Variance
{DECIMAL 1 1 }
{DECIMAL 2 3 }
```

```
  Jan Feb
  //   {DECIMAL 1 1 4 }  These lines are commented; the
  //   {DECIMAL 2 3 6 }  Report Extractor ignores them.
<ROW (Product)
<DESCENDANTS "100"
  !
```

2 つの{DECIMAL}コマンドは、個々の列 1、3、4 および 6 をフォーマットするように位置付けられています。

レポート・スクリプトの例: 直接割当てによる列のフォーマット

```
   <PAGE (Measures, Market)
California Sales
  <COLUMN (Scenario, Year)
  Actual Budget Variance
  //   {DECIMAL 1 1 }   These lines are commented; the
  //   {DECIMAL 2 3 }   Report Extractor ignores them.
  Jan Feb
{DECIMAL 1 1 4 7 }
{DECIMAL 2 3 6 9 }
<ROW (Product)
<DESCENDANTS "100"
  !
```

両方のスクリプトの結果として得られるレポート:

```
               Sales California

      Actual      Budget      Variance
     Jan   Feb   Jan    Feb   Jan    Feb
    =====  ====  ====   ====  =====   ====
100-10  678.0   645  840.00  800.0  (162)  (155.00)
100-20  118.0   122  140.00  150.0   (22)  (28.00)
100-30  145.0   132  180.00  160.0   (35)  (28.00)
```

注： デフォルトでは、属性計算次元メンバー(たとえば、SUM、AVG)は、列として表示されます。これらを行の中に表示するには、それらを ROW コマンドの中に含める必要があります。

長い列名と行名への対応

長すぎて列に収まらないメンバー名は、自動的に切り捨てられます。チルダ文字(~)は、名前の一部が欠けていることを示しています。長いメンバー名は、別名を使用するときによく見られます。

表 94 にリストされているコマンドを使用して列を変更し、メンバー名全体を表示できます:

表 94　レポート・ライター・コマンド: 長い列名と行名

レポート・コマンド	説明
NAMEWIDTH	列内のすべての行メンバーの列幅を定義します。
NAMESCOL	行メンバー列が表示される場所を変更し、残りの列を左または右に移動して長いメンバー名を表示できるようにします。

ページ、列および行のフォーマットの抑制

表 95 にリストされているコマンドを使用して、レポートのページ見出し、列および行の表示を抑制できます:

表 95　レポート・ライター・コマンド: ページ見出し、列および行の抑制

レポート・コマンド	説明
SUPCOLHEADING	レポートのデフォルトの列見出し。
SUPEMPTYROWS	ゼロまたは欠落した値のみを持つ行。
SUPMISSINGROWS	欠落した値が含まれるすべての行。INCMISSINGROWS、INCZEROROWS または INCEMPTYROWS を使用して、空の行、欠落データまたはゼロを持つ行を表示します。
SUPPAGEHEADING	見出しの生成時のページ・メンバー見出し。
SUPALL	最終レポートのページ見出しと列見出し、すべてのメンバー名、改ページ、カンマおよび大カッコ。行メンバー名の列を表示するには、NAMESON コマンドを使用します。カンマと大カッコの使用をオンにするには、COMMAS と BRACKETS コマンドを使用します。
SUPHEADING	各ページの先頭のデフォルトの見出し(ページ・ヘッダーと列ヘッダー)または定義されていればカスタム・ヘッダー。
SUPNAMES	最終レポートの行メンバー名。NAMESON コマンドを使用して、行メンバー名をレポートに含めます。
SUPOUTPUT	計算、フォーマット設定などのすべての操作を処理し続けている間のすべての出力。 OUTPUT コマンドを使用して、SUPOUTPUT の動作を逆にします。
SUPCURHEADING	CURRENCY コマンドを使用して、レポートのデータ値を指定した通貨に換算するときの通貨情報。

行名の繰返し

レポートの各行で行メンバー名を繰り返すには、<ROWREPEAT コマンドを使用します。行メンバー名が次の行で変更されない場合にレポートの各行で行メンバー名を繰り返さない場合は、<NOROWREPEAT コマンドを使用します。NOROWREPEAT はデフォルトで使用可能です。

タブ区切り記号の使用

たとえば、レポート出力を別のフォームにエクスポートする場合に、レポート・スクリプトで、列間にスペースではなくタブを使用できます。

スペースをタブ区切り記号で置換するには、レポート・スクリプトの任意の場所に{TABDELIMIT}を入力します。

合計と小計の追加

列と行の計算によって、データベース・アウトラインに定義されていない追加の計算を作成できます。たとえば、列と行の計算を使用すると、選択したデータ・メンバーに基づいてレポートに追加の列または行を作成したり、追加または既存の列と行に対して計算を実行したりできます。

列と行の計算を含むレポート・スクリプトの例は、『Oracle Essbase テクニカル・リファレンス』の「レポート・スクリプトの例」ページの「サンプル 14: 列の計算」を参照してください。

列の合計

CALCULATE COLUMN コマンドでは、レポート列の作成、即時計算の実行および新規作成列での計算結果の表示を行います。

表 96 は、列計算コマンドのリストです:

表 96　レポート・ライター・コマンド: 列計算

レポート・コマンド	説明
CALCULATE COLUMN	レポート列を作成し、動的計算を実行し、新規作成列に計算結果を表示します。レポートでの列計算を一時的に使用不可にするには、OFFCOLCALCS コマンドを使用し、計算を再び使用可能にするには ONCOLCALCS コマンドを使用します。
REMOVECOLCALCS	レポートからすべての列計算定義を削除します。

CALCULATE COLUMN は、最大 499 個のアド・ホック列計算をレポートに追加します。各新しい計算済列は、作成順に既存の列の右側に追加され、次に使用可能な列番号が割り当てられます。これらの列は、列の範囲内のデータの合計または簡単な算術演算子からなる演算式を計算します。

CALCULATE COLUMN コマンドは、標準の算術演算をサポートします。

複数の列に同じ名前を使用した場合、Essbase では CALCULATE COLUMN コマンドで最後に指定した列のみが作成されます。2 番目の名前の先頭に 1 つのスペース(および 3 番目の名前の先頭に 2 つのスペースなど)を使用して、一意の列名を作成します。

または、右側に十分に長い説明テキストを追加して、列幅で切り捨てることもできます。たとえば、名前 Q1 Actual と Q1 Budget を使用して、レポートの表示に影響を与えずに似ている列名を区別できます。列名は、列ヘッダーのスペースがいっぱいになるまで、右揃えで印刷されます。超過する文字は右側で切り捨てられます。

長い列名ラベルを複数の行に分けます。ラベルを分けることができる行の最大数は、レポート指定に指定された列次元の数と同じです。列名を改行するには、改行する場所にチルダ(~)を挿入します。また、行の最大数を使用するには、列次元ごとに少なくとも 2 つのメンバーを指定する必要があります。

レポート・スクリプトの例:

```
   {CALCULATE COLUMN "Year to Date~Actual Total" = 1 : 2}
{CALCULATE COLUMN "Year to Date~Budget Total" = 3 : 4}
```

次のレポートが生成されます:

```
                  Sales East
     Actual  Year to Date    Budget   Year to Date
    Jan   Feb  Actual Total   Jan   Feb  Budget Total
   ===== ====== ============= ====== ====== =============
400-10  562   560     1,122   580   580     1,702
400-20  219   243      462    230   260      722
400-30  432   469      901    440   490     1,391
```

原則として、対称型レポートでは、計算済列名のレベルの数が列次元の数よりも少ない場合、各列次元の前のメンバー(左側)が、計算済列名で提供された最上位より上で、計算済列に属するとみなされます。通常の PYRAMIDHEADERS モードが使用されている場合、上位の列メンバーの中央揃えは、計算済列を含めるために右側に移動します。計算済列名と同じレベルの列ヘッダー・メンバーは、新しい計算済列に適用されないので、その中央揃えは移動しません。

BLOCKHEADERS モードが使用されている場合、つまり、列に適用するすべてのメンバーがその列より上で繰り返される場合、列ヘッダー・メンバーの中央揃えを移動するかわりに、メンバーが計算列名のより上位の該当するレベルで繰り返されること以外は、同じルールが適用されます。

非対称型レポートにはメンバー・プロパティを共有する列のグループがありません。このレポートでは、定義されている列レベルの数まで複数レベルの行名を使用できますが、前の列のメンバー・プロパティは、定義されていないレベルに対して自動的に共有されず、使用されません。

列ヘッダー次元の数が必要なレベルの数より少ない場合は、複数行の列ラベルを作成できます。この場合、TEXT、STARTHEADING、ENDHEADING などのフォーマット・コマンドを使用して、カスタム見出しを作成します。

列の番号付け

レポート内の複数のセクションが異なる数の列を持つため、通常(計算対象外)の列の数がレポート内で変わる場合は、次に示すように、計算済列を識別するために使用される列の番号が変わります:

- レポートの最初のセクションに 12 個の列(行名の列を含む)があり、3 つの計算済列が宣言されている場合は、列番号 0-11 は通常の列になり、列 12-14 は計算済列になります。
- レポートの 2 番目のセクションで通常の列の数を 6 に減らすと、通常の列は列 0-5 になり、同じ計算済列は列 6-8 になります。
- 同様に、通常の列の数が増えると、計算済列の番号付けはより大きい番号から始まります。

たとえば、CC1、CC2 および CC3 は 3 つの計算済列名の名前を表しています。通常の列の数が変化する 2 つの異なるセクションを持つレポートでの列の番号付けは、次のようになります:

```
   internal
col # s: 0   1    2    3   4    5    6    7
      Jan   Feb   Mar   Apr
   CC1   CC2   CC3

      ===   ===   ===   ===   ===   ===   ===
  Sales    1    3    5    3   22   55   26
  Expense   1    2    5    3   23   65   33

        same report- new section
internal
col # s: 0   1    2    3   4    5
     Qtr1   YTD
   CC1   CC2   CC3

      ===   ===   ===   ===   ===
  Sales    2    9   22   57   36
  Expense   4    8   56   45   33
```

2 番目のセクションに計算済列を入れない場合、または異なる列計算のセットを必要とする場合、REMOVECOLCALCS コマンドを使用して、古い列計算を消去します。その後、新しい列計算を定義できます。

この例では、3 つの列計算のすべては、列 1 と列 2 を除き、通常の列への参照は持たないことを前提とします。レポートの 2 番目のセクションの開始時に CC3 の計算が= 1 + 3 + 6 である場合、列計算が存在しない列(6)を参照したという内容のエラーが発生します。

行の合計

行計算では、レポートに要約行を作成します。要約行を使用して、行の範囲内のデータの合計を計算したり、簡単な算術演算子からなる演算式を計算できます。

表 97 は、行計算コマンドのリストです:

表 97　レポート・ライター・コマンド: 行計算

レポート・コマンド	説明
CALCULATE ROW	行を作成して、その行を行の名前またはラベルに関連付けます。このプロセスは、変数の宣言と似ています。CALCULATE ROW を使用して簡単な計算を実行することもできます。複雑な計算には SETROWOP を使用します。「OFFROWCALCS」と「ONROWCALCS」も参照してください。
OFFROWCALCS	レポートでの行計算を一時的に使用不可にします。「CALCULATE ROW」と「ONROWCALCS」も参照してください。
ONROWCALCS	OFFROWCALCS の使用後に、計算を再び使用可能にします。「CALCULATE ROW」と「OFFROWCALCS」も参照してください。

レポート・コマンド	説明
SETROWOP	CALCULATE ROW で指定された行の複雑な計算を定義します。SETROWOP では、後続のすべての出力データ行に適用する計算演算子を定義します。PRINTROW コマンドを使用して、新規作成行に計算結果を表示できます。
PRINTROW	CALCULATE ROW で指定された行をレポートに即時表示します。
CLEARROWCALC	計算済行の値を#MISSING にリセットします。「CLEARALLROWCALC」も参照してください。
CLEARALLROWCALC	CLEARROWCALC コマンドの使用後にすべての計算行の値をリセットします。
SAVEROW	取り込んだデータで計算済行を作成します。「SAVEANDOUTPUT」も参照してください。
SAVEANDOUTPUT	データを取り込んで、SAVEROW コマンドの使用後に結果を出力します。

列を指定するコマンドでは、列の元の順序で決定された有効なデータ列番号を使用する必要があります。

- 式内のすべての演算子の前後にはスペースを 1 つ付けます。
- Essbase では、ネストした(カッコ付きの)式をサポートしません。
- Essbase では、式で整数と浮動小数点の定数を、単一のエントリまたは配列のメンバーとしてサポートします。

CALCULATE ROW コマンドでは、定数、その他の計算済行および演算子からなる等式として、演算(+、-、*、/または OFF)を指定できます。等式は、宣言時に評価されます。メンバー名は、CALCULATE ROW コマンドの式ではサポートされません。

演算子を指定した場合、その演算子は後続の出力行に適用され、その結果は計算済行に保管されます。演算子の指定は、一連の行を集約して、小計または合計を得るときに便利です。演算子をリセットするには、SETROWOP を使用します。CALCULATE ROW コマンドで等式も演算子も指定しなかった場合、+演算子が指定されたとみなされます。

CALCULATE ROW コマンドは、標準的な算術演算をサポートします。

レポート・スクリプトの例:

```
{ CALC ROW "Total Sales" = "Sales..Group1"
+ "Sales..Group2" }
```

この例では、他の 2 つの計算行を基にして「Total Sales」を作成します。

データの表示方法の変更

フォーマット・コマンドを使用して、最終レポートでのデータの表示方法をカスタマイズできます:

下線付け

視覚的な手段として、表 98 にリストされているコマンドで、下線を使用してレポート内の情報のブロックを分割します:

表 98　レポート・ライター・コマンド: 下線

レポート・コマンド	説明
UNDERLINECHAR	レポート内で表示するデフォルトの下線文字を設定します。
UCHARACTERS	前の行にある空白以外のすべての文字に下線を付けます。
UCOLUMNS	前の行にあるすべての列に下線を付けます。
UDATA	行のすべてのデータ列に下線を付け、行名列には下線を付けません。
UNAME	前の行にあるすべての行名列に下線を付け、データ列には下線は付けません。
UNAMEONDIMENSION	コマンドで指定したメンバーと同じ次元のメンバーが変更されるたびに、行の行メンバー名に下線を付けます。新しい行に下線を付けないようにするには、NOUNAMEONDIM コマンドを使用します。

データのフォーマットの抑制

表 99 にリストされている SUPPRESS コマンドを使用して、最終レポートに表示する必要がないデータを抑制できます:

表 99　レポート・ライター・コマンド: データのフォーマットの抑制

レポート・コマンド	説明
SUPBRACKETS	負数を囲むカッコ。BRACKETS コマンドを使用すれば、カッコが再び使用可能になります。
SUPCOMMAS	999 より大きな数のカンマ。COMMAS コマンドを使用すれば、カンマが再び使用可能になります。
SUPZEROROWS	ゼロのデータ値のみを持つ行。INCZEROROWS または INCEMPTYROWS を使用すれば、再び表示できます。
SUPEUROPEAN	ヨーロッパ式の数値の表示(2.000,01 は、ヨーロッパ以外の表示では 2,000.01)。EUROPEAN コマンドを使用すれば、ヨーロッパ式の表示が再び使用可能になります。
SUPFEED	あるページの行数が現在の PAGELENGTH 設定より大きくなるときの、改ページの自動挿入。
SUPFORMATS	下線やスキップなどの出力を生成するフォーマット。INCFORMATS を使用すれば、再び表示できます。
SUPMASK	MASK コマンドを使用してレポートに定義されたテキスト・マスク。INCMASK を使用すれば、再び表示できます。

SUPPRESS コマンドを使用したフォーマットの抑制の詳細は、565 ページの「ページ、列および行のフォーマットの抑制」を参照してください。

インデント

インデントを使用して、スクリプトの行レベルに視覚的な目印を設定します。表 100 は、インデント・コマンドのリストです:

表 100　レポート・ライター・コマンド: インデント

レポート・コマンド	説明
INDENT	指定された文字数のみ、列出力順で最初の行名列を移動します。
INDENTGEN	データベース・アウトライン内の世代に基づいて、行名列の後続の行メンバーをインデントします。世代名に基づいて行メンバー名を左に揃えるには、NOINDENTGEN コマンドを使用します。

カスタム・タイトルの挿入

タイトルはユーザー生成のオプションであり、自動的に生成されるページや列見出しおよび行名とは異なり、レポート・ページ上のデータを説明します。

タイトルは各レポート・ページの先頭で繰り返され、次のようなレポートの情報を提供します:

- 日付とタイム・スタンプ
- レポートを実行しているユーザーの名前
- ページ番号
- ソース・データベースの名前
- 他の説明的な情報

タイトルをレポートに追加するには、TEXT コマンドを次のように組み合せて使用します:

- レポート内の情報を自動的に表示する、事前定義の任意のキーワード
- 定義するテキスト文字列

> **注：** レポートの終わりで TEXT コマンドを使用して、要約情報を提供することもできます。

欠落テキストまたはゼロのラベルによる置き換え

レポートを実行する場合、データを取得できなかった空のデータ・セルや、値がゼロであるセルが、多く存在することがあります。

データ値が見つからない場合、レポートではデータ・セルにデフォルトの#MISSING ラベルを表示します。

➤ #MISSING ラベルをテキスト・ラベルに置き換えるには、スクリプト内の、#MISSING ラベルをテキスト・ファイルに置き換える箇所に、次のコマンドを入力します:

```
{MISSINGTEXT ["
```

```
text
"]}
```

text は、データ・セルに表示されるテキスト文字列です。

MISSINGTEXT コマンドをレポート・スクリプトの任意の箇所に置くと、そのコマンドはスクリプト全体に適用されます。

注： レポートに#MISSING ラベルが表示されないような設定もできます。SUPPRESS コマンドを使用したラベルの抑制方法は、570 ページの「データのフォーマットの抑制」を参照してください。

➤ ゼロをテキスト・ラベルに置き換えるには、スクリプト内の、ゼロをテキスト・ラベルに置き換える箇所に、次のコマンドを入力します:

```
{ZEROTEXT ["
text
"]}
```

text は、データ・セルに表示されるテキスト文字列です。

注： 値が#MISSING と等しい場合、AFTER コマンドを使用してその値の後に挿入したすべての文字列は、印刷されません。#MISSING を他の値(0 など)に置き換えた場合でも、AFTER 文字列は印刷されません。

空白の追加

レポート内に空白を追加すると、合計などの重要な情報が読者の目を引きます。表 101 は、空白の追加に使用するコマンドのリストです:

表 101　レポート・ライター・コマンド: 空白

レポート・コマンド	説明
SKIP	最終レポート内に 1 つ以上の空白行を追加します。
SKIPONDIMENSION	コマンド内に指定されたメンバーと同じ次元のメンバーがレポートの次の行で変更されるとき、1 つの空白行を追加します。 NOSKIPONDIMENSION コマンドを使用して、新しい行の挿入をオフにします。

データ値の表示方法の変更

表 102 にリストされているコマンドを使用して、最終レポートのデータ表示方法を変更できます:

表 102　レポート・ライター・コマンド: データ値の表示

レポート・コマンド	説明
COMMAS	SUPCOMMAS または SUPALL コマンドでカンマが抑制された後で、999 よりも大きい数字に対するカンマの表示をオンにします。
BRACKETS	スクリプトで先に SUPBRACKETS コマンドを使用した場合に、負の記号のかわりに、負数を囲むカッコの表示をオンにします。
AFTER	データ値の後に、パーセンテージ記号または他の文字を組み込みます。
BEFORE	データ値の前に、ドル記号または他の文字を組み込みます。
EUROPEAN	小数点を 3 桁区切り文字として使用し、カンマを数の小数部と整数部の分離に使用して数を表示するヨーロッパ方式(非ヨーロッパ方式の 2,000.01 は 2.000,01 となる)を使用します。
MASK	各出力行のテキストを指定された文字と位置で上書きします。

メンバーの選択とソート

最終レポートのデータは、選択したメンバーとその表示順序に基づいて表示されます。また、条件付き取得を使用して、メンバーの選択とソートをさらに細かく指定できます。

メンバーの選択

メンバー選択コマンドは、兄弟、世代、レベルなどのデータベース・アウトラインの関係に基づいてメンバーの範囲を選択する抽出コマンドです。メンバー選択コマンドを使用すれば、メンバー選択コマンドが基準とするメンバー名を変更しないかぎり、アウトラインへの変更を自動的にレポートに反映できます。属性次元をメンバー選択コマンドに入れることができます。

表 103 は、メンバー選択コマンドのリストです:

表 103　レポート・ライター・コマンド: メンバーの選択

レポート・コマンド	説明
ALLINSAMEDIM	次元メンバーと同じ次元のメンバーを選択します。
ALLSIBLINGS	指定されたメンバーの兄弟を組み込みます。
ANCESTORS	指定されたメンバーの祖先を組み込みます。
ATTRIBUTE	その属性に基づく基本次元メンバーを選択します。
CHILDREN	指定されたメンバーの直下にあるレベルのメンバーを選択します。
DESCENDANTS	次元の最上位を除き、指定されたメンバーの子孫をレポートに組み込みます。
DIMBOTTOM	特定のメンバーが属する次元のすべてのレベル 0 のメンバーを選択します。
DIMTOP	次元の最上位のメンバーを組み込みます。

レポート・コマンド	説明
IANCESTORS	メンバーとその祖先を組み込みます。
ICHILDREN	指定されたメンバーとその直下にあるレベルのメンバーを選択します。
IDESCENDANTS	指定されたメンバーとその子孫を組み込みます。
IPARENT	指定されたメンバーとその親を組み込みます。
OFSAMEGEN	指定されたメンバーと同じ次元と世代のメンバーを組み込みます。
ONSAMELEVELAS	指定されたメンバーと同じ次元と同じレベルにあるメンバーを組み込みます。
PARENT	指定されたメンバーの親をレポートに組み込みます。
TODATE	特定の日付のデータまたは特定の日付の前後の期間のデータを抽出します。
WITHATTR	指定された属性次元に関連付けられた基本次元メンバーを組み込みます。

世代名とレベル名によるメンバーの選択

世代名とレベル名の選択コマンドでは、次のいずれかのアイテムに基づいて、特定のレベルまたは世代のメンバーを識別します:

- アウトラインのデフォルトの世代名またはレベル名
- アウトラインのユーザー定義の世代名またはレベル名

世代名とレベル名を使用すると、アウトラインへの変更が自動的にレポートに反映されます。独自の世代名とレベル名を定義するか、Essbase に用意されているデフォルト名を使用できます。61 ページの「世代とレベル」を参照してください。

可能な場合は世代名またはレベル名を使用すると、レポートの管理が簡単になります。レポートでメンバー名を指定する必要がないので、メンバー名が変更またはデータベース・アウトラインから削除された場合にレポートを変更する必要がありません。

注： 世代名とレベル名は独立したコマンドです。<DESCENDANTS や<CHILDREN などのレポート抽出コマンドまたはフォーマット・コマンドでメンバー名のかわりに世代名またはレベル名を使用することはできません。

➤ デフォルトのレベル名を使用するには、スクリプト内の、デフォルトのレベル名でメンバーを選択する箇所で、次のフォーマットを使用します:

```
Lev
n
,
dimensionName
```

n は、レベル番号です。

dimensionName は、メンバーを選択する次元の名前です。

注： カンマの後にスペースを置かないでください。

たとえば、Lev1,Year では、年次元のすべてのレベル 1 のメンバーを選択します。

➤ デフォルトの世代名を使用するには、スクリプト内の、デフォルトの世代名でメンバーを選択する箇所で、次のフォーマットを使用します:

```
Gen
n
,
dimensionName
```

n は、世代番号です。

dimensionName は、メンバーを選択する次元の名前です。

注： カンマの後にスペースを置かないでください。

たとえば、Gen2,Year では、年次元のすべての世代 2 のメンバーを選択します。

注： これらのデフォルトの世代名とレベル名はアウトライン・エディタでは表示されません。

次の例のレポート・スクリプトでは、デフォルトの世代名 Gen2,Year を使用して、Year 次元のメンバー Qtr1、Qtr2、Qtr3 および Qtr4 を含むレポートを生成します:

```
   <PAGE(Product)
<COLUMN(Year)
<ROW (Measures)
{OUTALTNAMES}
Cola
Gen2,Year
Sales Profit
  !
```

次のレポートが生成されます:

```
                Cola Market Scenario
          Qtr1    Qtr2    Qtr3    Qtr4
         ======== ======== ======== ========
Sales       14,585  16,048  17,298  14,893
  Profit     5,096   5,892   6,583   5,206
```

重複メンバーの選択

重複するメンバー名のアウトライン選択では、表 104 にリストされているコマンドを使用できます:

表 104　レポート・ライター・コマンド: 重複メンバーの選択

レポート・コマンド	説明
REPQUALMBR	一意のメンバー名についてはメンバー名を、重複するメンバー名については修飾名を表示します。
REPMBR	指定した次元のメンバーのメンバー名のみを表示します。
REPMBRALIAS	メンバー名の後に別名を表示します。
REPALIAS	レポート出力に含まれるメンバーに対し、別名の後にメンバー名を表示します。
REPALIASMBR	指定した次元のメンバーに対し、別名の後にメンバー名を表示します。
OUTPUTMEMBERKEY	重複するメンバー名のメンバー識別子を表示します。

動的時系列メンバーの選択

データベース・アウトラインで動的メンバーを作成し、識別します。動的メンバーは、レポート・スクリプトの生成など、ユーザーの取得要求時にのみ計算されるメンバーです。時間次元には、特別な動的時系列オプションがあります。動的時系列オプションには、次の予約世代名があります。この世代名は、表 105 にリストされている、アウトラインの別名テーブルで定義できます:

表 105　動的時系列: 予約世代名

世代名	予約名	説明
履歴	H-T-D	履歴の初めからの累計
年	Y-T-D	年次累計
季節	S-T-D	当季初めからの累計
期間	P-T-D	当期間初めからの累計
四半期	Q-T-D	四半期間累計
月	M-T-D	月次累計
週	W-T-D	当週初めからの累計
日	D-T-D	当日初めからの累計

470 ページの「動的時系列メンバーに対する事前定義の世代名の適用」を参照してください。

注： アウトラインのデータベース・ヘッダー・メッセージでは、現在のアウトラインで使用可能な動的メンバーの番号を識別します。

➤ 動的時系列メンバーを選択するには、スクリプト内の、動的時系列メンバーを選択する箇所で、次のいずれかを使用します:

- `<LATEST memberName`

  memberName は時間次元のメンバーの名前です。

  <LATEST コマンドはレポート・スクリプト全体に適用されるグローバル・コマンドであり、次元内の最下位レベルのメンバーに基づく集約です。

- `reservedName(memberName)`

  reservedName は予約された動的時系列世代名であり、memberName は時間次元のメンバーの名前です。

  この構文を使用して動的時系列を指定する場合、時系列名は引数内にリストされたメンバーにのみ関連付けられます。

レポート・スクリプトを実行すると、メンバーは動的に更新され、情報は最終レポートに組み込まれます。

注： 動的時系列の文字列は、データベース・アウトラインに表示されているとおりに入力する必要があります。文字列を作成して最終レポートに組み込むことはできません。データベース・アウトラインで動的時系列メンバーの別名テーブルを作成し、事前定義の世代名のかわりに別名を使用することはできます。

ブール演算子によるメンバーの選択

ブール演算子を使用すれば、レポートで厳密なメンバーの組合せを指定できます。これは、大規模なアウトラインを操作するときに役に立ちます。AND、OR および NOT ブール演算子を抽出コマンドと組み合せて使用し、レポート・スクリプトでメンバー選択を詳しく指定します。

- AND。すべての条件が満たされる必要がある場合。
- OR。複数の条件のうち 1 つが満たされる必要がある場合。
- NOT。選択した条件の逆を選択する場合。

➤ 演算子を使用してブール式を作成するには、スクリプト内の、リンクを使用する箇所に、次のフォーマットを入力します:

```
<LINK (
extractionCommand
 [
operator extractionCommand
])
```

extractionCommand はデータを取得するメンバー選択コマンドであり、operator は AND 演算子または OR 演算子です。

注： 同じ次元のメンバーを選択します。すべての抽出コマンドの引数は、前の例のように丸カッコで囲む必要があります。NOT は抽出コマンドにのみ関連付けることができ、式全体には適用されません。

ブール演算子は、UDA やワイルドカードなど、メンバー選択コマンドと一緒に使用できます。LINK コマンドで使用できる有効な抽出コマンドのリストは、『Oracle Essbase テクニカル・リファレンス』を参照してください。

例:

次の例では、「100」サブツリーにあるすべての菓子製品と、「100-10」と同レベルにあるすべての製品を選択します:

```
<LINK ((<IDESCENDANTS("100") AND <UDA(Product,Sweet)) OR ONSAMELEVELAS "100"-10")
```

次の例では、「100」サブツリーにある菓子以外のすべての製品と、「100-10」と同じレベルにあるすべての製品を選択します。

```
<LINK ((<IDESCENDANTS("100") AND NOT <UDA (Product,Sweet)) OR ONSAMELEVELAS
"100"-10")
```

メンバー選択条件の絞り込みの他の例は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

代替変数によるメンバーの選択

代替変数は、定期的に変化する情報のグローバル・プレースホルダとして機能します。Administration Services、MaxL または ESSCMD を使用して、代替変数をサーバーに設定し、各変数に値を割り当てます。その値はいつでも変更でき、レポート・スクリプトへの手動変更を減らすことができます。代替変数を設定するにはデータベース・マネージャ以上の役割が必要です。

たとえば、多くのレポートはレポート期間に依存します。当月に基づいてレポートを生成する場合、毎月レポート・スクリプトを手動で更新する必要があります。サーバーに設定した CurMnth などの代替変数を使用すれば、各月の割当て値を該当する期間に変更できます。最終レポートを実行すると、Essbase によって動的に情報が更新されます。

データベース・アウトラインで代替変数を作成および変更する方法は、118 ページの「代替変数の使用」を参照してください。先頭の&文字の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

代替変数は次のレベルで設定できます:

- サーバー。サーバー上のすべてのアプリケーションとデータベースから変数にアクセスできます。

- アプリケーション。アプリケーション内のすべてのデータベースから変数にアクセスできます。
- データベース。指定されたデータベースにアクセスできます。

➤ 代替変数を使用するには、スクリプト内の、変数を使用する箇所で、次のフォーマットを使用します:

```
&
variableName
```

variableName は、サーバー上に設定された代替変数と同じです。

たとえば、

```
<ICHILDREN &CurQtr
```

これは次のようになります

```
<ICHILDREN Qtr1
```

注： 代替変数では、レポートを実行するアプリケーションやデータベースからアクセス可能であることが必要です。また、変数名は英数字の組合せであり、最大サイズは付録 A「制限」で指定されています。変数名にスペースと句読点を使用することはできません。

レポート・スクリプトを実行すると、Essbase によって変数名が代替値に置換され、その情報が最終レポートに組み込まれます。

属性によるメンバーの選択

属性を使用して、基本メンバーの 1 つ以上の特性に基づいて、データを選択し、レポートできます。属性に従って基本次元のメンバーをグループ化および分析できます。また、複数の属性に基づいてクロス集計レポートを実行できます。<ATTRIBUTE コマンドを使用して、属性に関連付けられているすべての基本次元メンバーを選択できます。たとえば、ニューヨークでの 12 オンス単位のグレープ・ジュースとオレンジ・ジュースの第 1 四半期の販売量について、Sample.Basic データベースをクエリーできます。

➤ 特定の属性に基づいてメンバーを選択するには、スクリプト内の、特定の属性に基づいてメンバーを選択する箇所で、次のフォーマットを使用します:

```
<ATTRIBUTE
memberName
```

memberName は属性次元メンバーの名前です。次に例を示します:

```
<ATTRIBUTE Bottle
```

ボトルでパッケージされたすべての製品が戻されます。

複数の属性次元で同じ名前のメンバーを使用できます。たとえば、属性次元オンスと属性次元年齢で、それぞれ 24 という名前のメンバーを使用できます。クエリーによって正しい結果が戻されるようにするには、完全な属性次元メンバー名を指定します。次のフォーマットでは、24 オンス単位でパッケージされたすべての製品が戻されます:

```
<ATTRIBUTE Ounces_24
```

属性タイプは、テキスト、数値、日付、ブールのいずれかにできます。166 ページの「属性タイプの理解」を参照してください。

サンプル・レポートについては、『Oracle Essbase テクニカル・リファレンス』の、サンプル 20: メンバー選択での属性の使用に関する項を参照してください。

属性の関連付けによるメンバーの選択

<WITHATTR コマンドを使用して、1 つ以上の属性に関連付けられているすべての基本次元メンバーを選択できます。たとえば、パッケージ・タイプ属性次元のメンバーに関連付けられているすべての製品を表示できます。スクリプト内の、メンバーを選択する箇所に、次の構文を入力します:

```
<WITHATTR (
attributeDimensionName
, "
Operator
",
Value
)
```

次のコマンドでは、人口属性次元の属性小に関連付けられている、すべての基本次元メンバーが戻されます。

```
<WITHATTR (Population, "IN", Small)
```

次のコマンドでは、オンス属性次元の属性 32 に関連付けられている、すべての基本次元メンバーが戻されます。

```
<WITHATTR (Ounces, "<", 32)
```

注: <WITHATTR コマンドを LINK コマンド内で使用して、次の例に示すようにメンバー選択を細かく指定できます: `<LINK ((<WITHATTR (Ounces, "<", 32) AND <WITHATTR ("Pkg Type", "=", Can))`

サンプル・レポートについては、『Oracle Essbase テクニカル・リファレンス』の、サンプル 21: メンバー選択での WITHATTR コマンドの使用に関する項を参照してください。

日付によるメンバーの選択

<TODATE コマンドを使用して、特定の日付、特定の日付より前の期間または特定の日付より後の期間の属性データを抽出できます。たとえば、1996 年 12 月 10 日、1996 年 12 月 10 日より前、または 1996 年 12 月 10 日より後に発売されたすべての製品に関する情報を抽出できます。<TODATE コマンドは、<WITHATTR コマンド内で使用する必要があります。たとえば、次のフォーマットでは、1996 年 12 月 10 日に発売されたすべての製品のデータが戻されます。

```
<WITHATTR ("Intro Date", "=", <TODATE ("mm-dd-yyyy", "12-10-1996")
```

次のフォーマットでは、1996 年 12 月 10 日より前に発売されたすべての製品のデータが戻されます。

```
<WITHATTR ("Intro Date", "<", <TODATE ("mm-dd-yyyy", "12-10-1996")
```

次のフォーマットでは、1996 年 12 月 10 日より後に発売されたすべての製品のデータが戻されます。

```
<WITHATTR ("Intro Date", ">", <TODATE ("mm-dd-yyyy", "12-10-1996")
```

注： サポートされている日付フォーマットのタイプは、mm-dd-yyyy または dd-mm-yyyy です。日付は、1970 年 1 月 1 日から 2038 年 1 月 1 日までの範囲(両端の日付を含む)で指定する必要があります。

UDA によるメンバーの選択

UDA を使用して、共通の特性に基づいてデータを選択し、レポートできます。UDA は、不均衡な階層(この階層では、次元のメンバーによってメンバー・レベルが異なります)を持つアウトラインからメンバーを選択するときに役に立ちます。色、サイズ、性別、味などの共通のメンバー特性に関する UDA をサーバーに設定できます。サーバーに UDA を設定するにはデータベース・マネージャ権限が必要です。

UDA は属性とは異なります。UDA は、特定の特性に基づいてデータを抽出するために作成するメンバー・ラベルです。ただし、UDA を使用して、データのグループ化、クロス集計レポートの実行、選択したデータの取得を行うことはできません。データ分析では、UDA は属性ほど有効ではありません。

UDA コマンドをブール演算子とともに使用して、レポート・クエリーを細かく指定できます。例は、577 ページの「ブール演算子によるメンバーの選択」を参照してください。

158 ページの「UDA の作成」も参照してください。

多数のメンバー(たとえば、メンバーが 50 万程度)が 1 つの UDA に割り当てられている大規模なアウトライン(たとえば、メンバーが 100 万以上)では、UDA 単位のメンバー・セットのクエリーに長時間かかる可能性があります。クエリーのパフォーマンス改善のため、essbase.cfg で PRELOADUDANAMESPACE 構成設定を TRUE に設定します。この場合、アプリケーションのロードに追加メモリーが必要な場合があります。『Oracle Essbase テクニカル・リファレンス』を参照してください。

➤ UDA に基づいてメンバーを選択するには、スクリプト内の、UDA に基づいてメンバーを選択する箇所で、次のフォーマットを使用します:

```
<UDA (
dimensionName
,"
UDAstring
")
```

dimensionName は選択するメンバーの次元で、UDAstring はサーバーに設定されている UDA です。次の例では、菓子属性を持つ製品次元のメンバーを選択します。

```
<UDA (product,"Sweet")
```

レポート・スクリプトを実行すると、Essbase によって UDA のメンバーが最終レポートに組み込まれます。

注： UDA 文字列は、データベース・アウトラインに表示されているとおりに入力する必要があります。UDA 文字列を作成してレポート・スクリプトに組み込むことはできません。

ワイルドカードによるメンバーの選択

ワイルドカードを使用して、レポート・スクリプトでメンバー、世代またはレベルの名前を選択できます。メンバー名を使用すると、Essbase によってメンバーとそのメンバーのすべての子孫が検索されます。世代名またはレベル名を指定すると、Essbase によってその世代またはレベルのメンバーのみが検索されます。

ワイルドカードを使用すると、スクリプトに必要なメンバー情報の量が減るので、スクリプトのメンテナンスが簡単になります。

次の 2 つのタイプのワイルドカードがレポート・ライターでサポートされています:

- 共通のメンバー・プロパティを検索するための文字列の末尾のアスタリスク(*)
- 任意の単一文字のメンバー・プロパティを表現するための文字列中の、パターン照合疑問符(?)

➤ 末尾ワイルドカードを使用してメンバーを選択するには、スクリプト内の、末尾ワイルドカードを使用してメンバーを選択する箇所で、次のフォーマットを使用します:

```
<MATCH (
memberName
,"
character
*")
```

memberName は選択するメンバーの名前であり、character は次のメンバーの開始文字です。次のレポート・スクリプトを使用します:

```
<MATCH (Year,"J*")
```

Jan、Jun および Jul が戻されます。

➤ パターン照合ワイルドカードを使用してメンバーを選択するには、スクリプト内の、パターン照合ワイルドカードを使用してメンバーを選択する箇所で、次のフォーマットを使用します:

```
<MATCH (
memberName
,"???
characters
")
```

memberName は選択するメンバーの名前であり、characters は次のメンバーの文字です。次のレポート・スクリプトを使用します:

```
<MATCH (Product,"???-10")
```

100-10、200-10、300-10 および 400-10 が戻されます。

注： Sample.Basic データベースの例では、3 つの疑問符によって、文字列の 3 つの可変文字を表しています。この例で 2 つの疑問符を使用した場合、一致は見つかりません。疑問符のワイルドカードは一致文字列の任意の場所に置くことができます。

共有メンバーの抑制

次のアイテムとともに使用して、レポート用にデータを抽出するときに以降の共有メンバーのインスタンスを非表示にできます:

- 世代名
- レベル名

- DIMBOTTOM コマンド
- OFSAMEGEN コマンド
- ONSAMELEVELAS コマンド

共有メンバーを抑制して、レポート内の不要なデータ重複を取り除きます。

➤ 共有メンバーを抑制するには、スクリプト内の、共有メンバーを抑制する箇所で、次のように入力します:

```
<SUPSHARE
```

<SUPSHARE を使用すると、レポート・スクリプトの実行期間中に共有メンバーが非表示になります。共有メンバーを再び表示するには、<SUPSHAREOFF を使用します。

メンバーの表示方法の選択

別名を使用すると、レポートが読みやすくなり、読者はメンバー名の意味ではなく、データ値に集中できます。メンバーは、レポートで別名を使用して表示できます。たとえば、メンバー名 100-20、100-30 ではなく、Diet Cola、Caffeine Free Cola などのページ名、列名、行名を表示できます。

修飾メンバー名は、重複メンバー・データベースで特定のメンバーを識別するのに役立ちます。重複メンバー名が見つかった場合には、修飾メンバー名を表示できます。たとえば、[State].[New York]と[City].[New York]は、メンバー名 New York を一意に識別します。重複メンバー名の詳細は、第 8 章「重複メンバーのアウトラインの作成および使用」を参照してください。

表 106 は、メンバー名表示に使用するコマンドのリストです:

表 106　レポート・ライター・コマンド: メンバー名表示

レポート・コマンド	説明
<REPALIAS	指定した次元にあるメンバーの別名を表示します
<REPALIASMBR	別名にメンバー名を付けて表示します
<REPMBRALIAS	メンバー名に別名を付けて表示します
<REPMBR	指定した次元のメンバーのメンバー名のみを表示します
<REPQUALMBR	固有のメンバー名についてはメンバー名を表示し、重複するメンバー名については修飾名を表示します
<OUTPUTMEMBERKEY	重複するメンバーのメンバー識別子、メンバー名および別名を表示します。

<REPALIAS、<REPALIASMBR、<REPMBRALIAS、<REPMBR および <REPQUALMBR は、特定の次元に適用できるフォーマット・コマンドです。これらのコマンドは、お互いを上書きします。たとえば、<REPALIASMBR Product が出現した場合、レポート・ライターではメンバー名の前に別名が表示されます。

これ以降のスクリプトで<REPMBRALIAS Product が出現した場合、レポート・ライターでは別名の前にメンバー名が表示されます。重複メンバーのアウトラインの場合、<OUTPUTMEMBERKEY コマンドと表内の他のコマンドの組合せが可能です。メンバー識別子と、メンバー名または別名、あるいはその両方をレポート出力に組み込むことができます。

メンバー名と別名を表示する例

メンバー名とその別名を組み合せて、レポートにメンバーを表示できます。メンバー名と別名を組み合せることによって、Diet Cola 100-20 や 100-30 Caffeine Free Cola などのように、よりわかりやすいページ名、列名および行名を表示できます。

- メンバー名と別名を表示するには、レポート・スクリプトで、<REPALIASMBR コマンドまたは<REPMBRALIAS コマンドを使用します。

次の例では<REPALIASMBR コマンドを使用して、メンバー名の前に別名を表示します:

```
<PAGE (Product, Measures)

<COLUMN (Scenario, Year)

Actual

<ICHILDREN Qtr1

<ROW (Market)

<IDESCENDANTS "300"

<REPALIASMBR Product

  !
```

次のレポートが生成されます:

```
           Dark Cream 300-10 Measures Actual
             Jan    Feb    Mar   Qtr1
          ======== ======== ======== ========
Market         800    864    880   2,544

          Vanilla Cream 300-20 Measures Actual
            Jan    Feb    Mar   Qtr1
          ======== ======== ======== ========
Market         220    231    239    690

          Diet Cream 300-30 Measures Actual
```

```
          Jan    Feb    Mar    Qtr1
        ======== ======== ======== ========
Market       897    902    896   2,695

         Cream Soda 300 Measures Actual
          Jan    Feb    Mar    Qtr1
        ======== ======== ======== ========
Market     1,917  1,997  2,015  5,929
```

メンバーのソート

レポート内に含まれるメンバーをソートする場合、メンバーがメンバー選択コマンドによって参照されるか、静的メンバー定義によって参照されるかによって、ソート・コマンドがメンバーに与える影響が異なることに注意してください。レポート・ライター・コマンドは、メンバー名またはデータ値によってメンバーをソートします。

<CHILDREN や<DESCENDANTS などのメンバー選択コマンドは、データベース・アウトラインで指定された順序でメンバーを選択します。デフォルトでは、メンバー選択コマンドを含んでいるレポートは、データベース・アウトラインの階層順序でメンバーを表示します。ソート・コマンドを使用してソート順序を指定することで、このデフォルトを上書きできます。

ソート・コマンドはメンバー選択コマンドによって選択されたメンバーの順序に影響するので、このコマンドは、適用するメンバー選択コマンドの前に置く必要があります。ソート・コマンドを指定した場合、ソート順序は、別のソート・コマンドによって上書きされるまで維持されます。

ソート・コマンドは、<CHILDREN や<DESCENDANTS などのメンバー選択コマンドを変更します。ソート・コマンドは、フォーマット中に最終的な行のソートを実行しません。すぐにソートを開始しない場合、および次の選択コマンドの前に必要に応じてソート・コマンドをオフにするために、そのソート・コマンドを上書きする場合は、レポート・スクリプトにソート・コマンドを配置するときに注意が必要です。

ソート・コマンドは静的メンバー定義には影響しません。

表 107 は、メンバーのソートに使用するコマンドのリストです:

表 107　レポート・ライター・コマンド: メンバーのソート

レポート・コマンド	説明
SORTALTNAMES	別名がレポート・スクリプトで使用される場合、メンバーの別名によってメンバーをアルファベット順にソートします。
SORTASC	以降のメンバーを、最下位の世代から最上位の世代まで昇順にソートします。
SORTDESC	以降のメンバーを、最上位の世代から最下位の世代まで降順にソートします。
SORTGEN	データベース・アウトラインのメンバーの世代に従って、以降のメンバーをソートします。
SORTLEVEL	データベース・アウトラインのメンバーのレベルに従って、以降のメンバーをソートします。

レポート・コマンド	説明
SORTMBRNAMES	メンバーをメンバー名によってアルファベット順にソートします。
SORTNONE	レポートに追加されたメンバーが、データベース・アウトラインに基づく通常の階層順序に従うように、前のすべてのソート・コマンドを使用不可にします。

データ値の制限と順序付け

表 108 にリストされたコマンドを使用して、レポートで条件付き取得とデータのソートを実行できます:

表 108　レポート・ライター・コマンド: データ値の制限と順序付け

レポート・コマンド	説明
TOP	戻す行の数を指定します。この行には、特定のデータ列の一番上の値が含まれている必要があります。
BOTTOM	戻す行の数を指定します。この行には、特定のデータ列の一番下の値が含まれている必要があります。
RESTRICT	行が戻される前に、データ行の列が満たす必要がある条件を指定します。
ORDERBY	データ列のデータ値に基づいて、レポートの行の順序付けを指定します。

条件付き取得では構成可能な変数が使用されます。993 ページの「バッファのサイズの変更」を参照してください。

操作の順序の理解

<RESTRICT、<ORDERBY、<TOP および<BOTTOM は、レポート・スクリプトの任意の場所で、任意の順序で使用できます。これらのコマンドを使用するときには、ページ・メンバーまたは列メンバーの前、あるいはページ・メンバーまたは列メンバーに広がる<PAGE コマンドまたは<COLUMN コマンド(たとえば、IDESCENDANTS、ICHILDREN)の前に、すべてのグローバル・スクリプト・フォーマット・コマンドを置きます。

Essbase では、次の順にデータを抽出し、制限と配列を適用します:

1. RESTRICT と、SUPPMISSING、SUPZEROS、SUPEMPTYROWS などの既存の制限オプションを適用します。
2. TOP または BOTTOM、あるいは両方を適用します。
3. ORDERBY を適用します。

Essbase によって、行が戻され、出力が表示されます。

ソート・コマンドでの TOP、BOTTOM および ORDERBY の使用

<TOP、<BOTTOM および<ORDERBY コマンドは、そのデータ値によってレポートの出力をソートします。Essbase では、最初に<TOP と<BOTTOM を適用し、次に<ORDERBY を適用します。レポートに、(データではなく、メンバーをソートする)<SORTMBRNAMES などのソート・コマンドがある場合、Essbase では、最初にソート・コマンドを適用し、次に<TOP と<BOTTOM、その次に<ORDERBY を適用します。<ORDERBY が最終のソートです。

RESTRICT の使用

<RESTRICT コマンドの引数では、行選択の修飾を指定できます。Essbase では、修飾された行のみをレポート出力結果に組み込みます。

<RESTRICT では、行メンバー選択で指定する行の範囲にのみ作用します。

Essbase では、左から右に制限を処理します。引数のリストを丸かっこで囲んでグループ化することはできません。

たとえば、次の例は有効な構文ではありません:

```
   RESTRICT (... (@DATACOLUMN(1) > 300 AND @DATACOLUMN(2) < 600)...)
```

!コマンドで終了するレポートごとに、<RESTRICT を 1 つのみ使用します。レポート・スクリプトに複数のレポートが含まれている場合、各<RESTRICT は前のレポートの<RESTRICT を上書きします。例:

```
   RESTRICT (@DATACOLUMN(1) > @DATACOLUMN(2) AND 800 < @DATACOLUMN(3)
OR @DATACOLUMN(4) <> #MISSING)
```

この<RESTRICT コマンドは、操作されると次の構文と同じになります:

```
   RESTRICT (((@DATACOL(1) > @DATACOLUMN(2)) AND (800<@DATACOLUMN(3))) OR
(@DATACOLUMN(4) <> #MISSING))
```

ORDERBY の使用

<ORDERBY コマンドは、指定された列のデータ値に従って出力行を配列します。ORDERBY コマンドでオプションの方向引数を使用すると、ASC フラグで昇順を指定し、DESC フラグで降順を指定できます。同じレポートの列ごとに異なるソート方向を指定できます。方向引数を使用しなかった場合、昇順(ASC)がデフォルトで使用されます。

配列する行のセットを特定するには、コマンドで行グループ次元を指定します。デフォルトの行グループは、最も内側の行次元です。

!コマンドで終了するレポートごとに、<ORDERBY を 1 つのみ使用できます。レポート・スクリプトに複数のレポートが含まれている場合、各<ORDERBY は前のレポートの<ORDERBY を上書きします。

フォーマット・コマンドでの ORDERBY の使用

<ORDERBY コマンドのガイドラインは次のとおりです:

- レポートで<ORDERBY を使用するときには、行フォーマット・コマンドを使用しないでください。<ORDERBY によって、フォーマットするメンバーを含んでいる行が移動するので、レポートの指定の箇所で予定されているフォーマット・コマンドが予期せず出現することがあります。
- 通常、行フォーマット・コマンドは使用しないで、列メンバーまたは列メンバーを展開する列コマンド(たとえば、「ICHILDREN column dimension, <column ..., column member」)の前に全体的なスクリプトのフォーマットを置きます。

TOP および BOTTOM の使用

<TOP コマンドと<BOTTOM コマンドでは、レポートで戻される行グループ内で、それぞれ最上位の行値または最下位の列値を持つ修飾行の数を指定します。行グループ・メンバーを指定しなかった場合、最も内側の行グループ次元がデフォルトの行グループになります。

同じレポートで<TOP と<BOTTOM を一緒に使用できますが、レポートごとに使用できる<TOP と<BOTTOM はそれぞれ 1 つのみです。同時に使用する場合、混乱を防ぐため、2 つのコマンドにはその引数と同じデータ列が必要です。<TOP コマンドと<BOTTOM コマンドの結果は、コマンドで指定したデータ列の値に基づいて降順でソートされます。

<TOP と<BOTTOM は、行メンバー選択で指定する行の範囲にのみ作用します。

注： <TOP または<BOTTOM が<ORDERBY と一緒に出される場合、<ORDERBY の順序付け列は、<TOP または<BOTTOM のデータ列と同じになる必要はありません。

<ORDERBY、<TOP、<BOTTOM コマンドの任意の組合せがレポート・スクリプトに一緒に存在する場合、行グループ・メンバー(<rowGroupMember>)は同じである必要があります。この制限によって、行グループ内での行のソートと配列に関する混乱を防ぐことができます。

注意 Essbase では、TOP または BOTTOM のソートを適用する前に、抽出された 1 セットのデータ行から、ソート対象列の#MISSING 値を含む行を破棄します。

たとえば、このコマンドでは、行グループごとに col2 (Actual, Qtr2)内に最上位のデータ値を持つ 2 つの行を戻します:

```
1- TOP (2, @DATACOLUMN(2))
```

Sample.Basic データベースに対してこのコマンドを実行すると、行グループは Product になり、これによって暗黙的に、レポートが Florida に対して 100-10 と 100-30 の製品行を戻し、Maine に対して 100-10 と 100-40 の製品行を戻します:

```
               Actual       Budget
          Qtr1    Qtr2    Qtr1   Qtr2
Florida   100-10    570    670    570   650
     100-20    235    345    321   432
     100-30    655    555    455   865
     100-40    342    342    432   234
Maine    100-10    600    800    800   750
     100-20    734    334    734   534
     100-30    324    321    235   278
     100-40    432    342    289   310
New York  100-10   1010   1210   1110   910
     100-20    960    760    650   870
     100-30    324    550    432   321
     100-40    880    980    880  1080
     100-50    #MI    #MI    #MI   #MI
```

この例では、行グループが「market」であるため、レポートごとに col2 (Actual, Qtr2)内に最上位のデータ値を持つ行を戻します。

```
2- TOP("market", 3, @DATACOLUMN(2))
```

次の行が出力されます:

```
   New York  100-10   1010   1210   1110   910
      100-40   880    980    880  1080
Maine    100-10    600    800    800   750
```

このコマンドでは、行グループごとに col2 (Actual, Qtr2)内に最下位のデータ値を持つ 2 つの行を戻します。

```
3- BOTTOM ("market", 2, @DATACOLUMN(2))
```

次の行が出力されます:

```
   Maine    100-20    734    334    734    534
      100-30    324    321    235    278
```

注： <TOP と<BOTTOM は、すべての制限が適用された後に、戻される(修飾)行の数に上限を適用します。この上限は、<TOP コマンドの行数と<BOTTOM コマンドの行数を足した値に等しいです。

他のレポート構成による TOP および BOTTOM への影響についての理解

<TOP コマンドと<BOTTOM コマンドを使用するときには、その操作が他のコマンドの影響を受けることに注意してください。特に、Essbase では、<SUPPMISSING、<SUPZEROS および<SUPEMPTYROWS オプションを制限として扱い、これらのオプションを<RESTRICT コマンドの制限とともに抽出された行に適用します。Essbase では、<TOP コマンドまたは<BOTTOM コマンドを処理する前、および<ORDERBY コマンドを適用する前に、この制限をデータ行に適用します。

フォーマット・コマンドでの TOP と BOTTOM の使用

レポートにフォーマット・コマンドが出現すると、そのコマンドに続くメンバーにそのコマンドが付加されます。たとえば、このシーケンスでは、列に下線を引くコマンド{UCOLUMNS}が、その次にくるメンバーに内部で付加されます。スクリプト 1 では、このコマンドは、「フロリダと仮定すると、市場の最初の子」として説明できる行メンバーに付加されます。

SCRIPT 1

SCRIPT 2

```
....          ....
< ROW Market       {UCOLUMNS}
{UCOLUMNS }         < Florida   (row member)
<ICHILDREN Market
< TOP ....        < BOTTOM ....
```

スクリプト 2 では、{UCOLUMNS}が行メンバー Florida に付加されます。Essbase では、行メンバー Florida を持つ行が検出されるたびに、{UCOLUMNS}が実行されます。TOP コマンドまたは BOTTOM コマンドが Florida を含んでいない行を戻した場合、行に付加されたフォーマット・コマンドが実行されることはありません。

したがって、フォーマットが必ず実行されるように、<COLUMN コマンド、または列メンバーに広がるコマンドの前にすべての汎用フォーマット・コマンドを置くことをお薦めします。ただし、行は<TOP コマンドまたは<BOTTOM コマンドによって抽出されない可能性があるので、行に作用するフォーマット・コマンドは使用しないでください。また、<TOP コマンドと<BOTTOM コマンドとともに<SAVEROW と<CALCULATE ROW を使用しないでください。

異なる通貨へのデータの換算

データベースに通貨パーティションがある場合は、レポート・スクリプトで通貨換算を計算できます。<CURRENCY コマンドを使用して、出力通貨と通貨タイプを設定します。<CURHEADING コマンドを使用して、通貨換算の見出しを表示します。

注： 通貨換算は透過パーティションではサポートされません。

通貨換算アプリケーションの詳細は、213 ページの「通貨換算アプリケーションの構築および換算の実行」を参照してください。

C、Visual Basic およびグリッド API を使用したレポートの生成

表 108 にリストされているように、C、Visual Basic および C グリッド API を使用してレポートを生成できます:

表 109　レポート API

タスク	C の API 関数	Visual Basic の API 関数	C Grid の API 関数
レポート仕様をアクティブ・データベースに送信開始する。	ESSBEGINREPORT	ESBBEGINREPORT	ESSGBEGINREPORT
アクティブ・データベースに送信中のレポート仕様の終わりにマークを付ける。	ESSENDREPORT	ESBENDREPORT	該当なし
アクティブ・データベースに単一文字列としてレポート仕様を送信する。	ESSREPORT	ESBREPORT	該当なし
アクティブ・データベースにレポート仕様をファイルから送信する。	ESSREPORTFILE	ESBREPORTFILE	ESSGREPORTFILE

説明と構文は、『Oracle Essbase API リファレンス』を参照してください。

35 MDXクエリーの記述

この章の内容

はじめに

MDX は、Essbase のデータ操作言語で、XML for Analysis 設立メンバーの共同仕様です。

Sample.Basic データベースに基づいている、この章の演習を行うには、MaxL シェルを使用します。先に進む前に、Essbase を起動して MaxL シェルにログオンします。また、この章に紹介されているとおりにサンプル・クエリーを作成するためにテキスト・エディタを用意してください。

注： MaxL シェルのかわりに、管理サービス・コンソールで MDX スクリプト・エディタを使用できます。ただし、この章の説明では、MaxL シェルを使用しています。

クエリーの要素の理解

この項では、簡単なクエリーを作成するための基礎として使用するテンプレートを作成します。

ほとんどのクエリーは、次の構文フレームワークに基づいて作成できます:

```
   SELECT
{}
```

```
ON COLUMNS
FROM Sample.Basic
```

1 行目の SELECT は、すべての MDX ステートメントの本体を開始するキーワードです。

2 行目の中カッコ{ }は、セットのプレースホルダです。前のクエリーでは、セットは空ですが中カッコはプレースホルダのままです。

➤ クエリー・テンプレートを作成するには:

1 **Sample.Basic データベースに対して実行できるサンプル・クエリーを保管するフォルダを作成します。**

たとえば、Essbase インストールの ARBORPATH/app/Sample/Basic ディレクトリに「queries」というフォルダを作成します。

2 **テキスト・エディタを使用して、空のファイルに次のコードを入力します:**

```
     SELECT
 {}
ON COLUMNS
FROM Sample.Basic
```

3 **ファイルを qry_blank.txt という名前で queries フォルダに保存します。**

注: テキスト・エディタではなく、Administration Services の MDX スクリプト・エディタを使用している場合は、MDX スクリプト・エディタからクエリーを qry_blank.MDX という名前で保存します。

セットおよびタプルの概要

セットとは、同じ次元性を持つ 1 つ以上のタプルの、順序付けられた集合のことです(次元性の詳細は、596 ページの「セットの指定のルール」を参照してください)。

タプルとは、任意の数の次元からメンバーまたはメンバーの組合せを参照する方法です。たとえば、Sample.Basic データベースでは、Jan がタプルで、(Jan, Sales)、([Jan],[Sales],[Cola],[Utah],[Actual])のように使用します。

メンバー名は、次の方法で指定できます:

- 実際の名前または別名を指定します。次に例を示します:
  - Cola
  - Actual
  - COGS
  - [100]

  メンバー名が数字で始まる場合、またはメンバー名にスペースが入っている場合、[100]のように大カッコで囲む必要があります。明確にしてコードを読

みやすくするため、すべてのメンバー名で大カッコを使用することをお薦めします。

メンバー名がアンパサンド(&)で始まる場合、`["&xyz"]`のように引用符で囲む必要があります。これは、先頭のアンパサンドが代替変数用に予約されるためです(617 ページの「MDX クエリーでの代替変数の使用」を参照)。また、`StrToMbr("&100")`として指定することもできます。

属性メンバーの場合は、(メンバー名を一意に識別できる)ロング名を使用します(例: `[12]`ではなく、`[Ounces_12]`)。

- メンバー名の接頭辞として、次元名、またはいずれかの祖先メンバー名を指定します(例: `[Product].[100-10]`および`[Diet].[100-10]`)。この方法ではあいまいさがなくなり、共有メンバーを正しく参照できるため、すべてのメンバー名にこの方法を使用することをお薦めします。

  注: メンバー名の修飾には複数の祖先を使用しないでください。複数の祖先を使用した場合、Essbase によってエラーが戻されます。たとえば、`[Market].[New York]`および`[East].[New York]`は、有効な New York の名前ですが、`[Market].[East].[New York]`を指定すると、エラーが戻されます。

- WITH セクションで定義された計算済メンバーの名前を指定します。

演習: 初めてのクエリーの実行

クエリー・テンプレートの 2 行目の中カッコ{ }は、セットのプレースホルダです。この演習では、クエリーにセットを追加して、実行します。

➤ クエリーを実行するには:

1. `qry_blank.txt` を開きます。
2. セットは 1 つのタプルと同じくらい簡単なので、セットとして `Jan` をクエリー・テンプレートに追加します。中カッコはそのままにします(中カッコは、関数呼出しによって生成されるセットを除き、すべてのセット指定に必要です)。

   2 行目の中カッコの内側に Jan と入力します:

```
      SELECT
{
      Jan
      }
ON COLUMNS
FROM Sample.Basic
```

3. クエリーを `qry_first.txt` という名前で保存します。
4. Essbase が起動していること(`essbase.exe` プロセスが実行中であること)を確認します。
5. Essbase に MDX ステートメントを渡すには、MaxL シェルまたは Administration Services の MDX スクリプト・エディタを使用して、ステートメントを Essbase に渡します。この章の例では、MaxL シェルを使用します。

MaxL シェルを起動して、有効なユーザー名とパスワードでログオンします。たとえば、

```
essmsh -l admin passwd
```

6 **SELECT クエリー全体をコピーして MaxL シェルに貼り付けますが、まだ「[Enter]」キーは押さないでください。**

7 **Basic の後で、「[Enter]」キーを押す前に、最後にセミコロンを入力します。セミコロンは、MDX 構文の要件ではありませんが、MaxL シェルでは、実行する準備ができているステートメントの終わりを示すために必要です。**

注： Administration Services の MDX スクリプト・エディタを使用している場合は、末尾にセミコロンを入力しないでください。

8 **「[Enter]」キーを押します。**

結果: 結果は次のようになります:

```
   Jan
 8024
```

セットの指定のルール

前述のとおり、セットは、1 つ以上のタプルの順序付けられた集合です。

たとえば、次のクエリーでは、{[100-10]}は 1 つのタプルで構成されたセットです。

```
   SELECT
 {[100-10]}
ON COLUMNS
FROM Sample.Basic
```

次のクエリー{([100-10], [Actual])}も 1 つのタプルで構成されたセットですが、この場合のタプルは 1 つのメンバー名ではありません。([100-10], [Actual])は、2 つの次元(製品とシナリオ)のメンバーで構成されたタプルを表しています。

```
   SELECT
 {([100-10], [Actual])}
ON COLUMNS
FROM Sample.Basic
```

セットに複数のタプルが含まれる場合は、次のルールが適用されます: セットの各タプルのメンバーは、セットの他のタプルのメンバーと同じ次元を表す必要があります。さらに、次元は同じ順序で表す必要があります。つまり、セットの各タプルには同じ次元性が必要です。例:

- 次のセットは、同じ次元性を持つ 2 つのタプルで構成されています:

```
{(West, Feb), (East, Mar)}
```

- 次のセットでは、Feb と Sales が異なる次元のメンバーであるため、次元性ルールが守られていません:

```
{(West, Feb), (East, Sales)}
```

- 次のセットでは、2 つのタプルには同じ次元が含まれていますが、2 番目のタプルの次元の順序が逆であるため、次元性ルールが守られていません:
-

```
{(West, Feb), (Mar, East)}
```

- セットは、セットの集合または空(タプルが含まれていない)であることが可能です。
- セットは、中カッコ{}で囲まれている必要があります(セットを戻す MDX 関数でセットが表される場合を除く)。

軸の指定の概要

軸とは、データベースからのクエリー結果のレイアウトの指定です。軸は、次のように MDX クエリーに指定します:

```
   SELECT
   <axis>
   [,
   <axis>
   ...]
FROM <database>
```

少なくとも 1 つの軸を MDX クエリーに指定する必要があります。

最大 64 個の軸を指定でき、AXIS(0)で始まり、AXIS(1)...AXIS(63)と続きます。一般的に、4 個以上の軸を使用することはありません。軸の順序は重要ではありませんが、0 から n までの軸のセットを指定するときに、0 と n の間の軸をスキップしないでください。また、次元は複数の軸上に表示できません。

表 110 に示すように、最初の 5 個の軸にはキーワードの別名があります:

表 110　軸のキーワード別名

軸のキーワード別名	軸
ON COLUMNS	AXIS(0)のかわりに使用可能
ON ROWS	AXIS(1)を置換え可能
ON PAGES	AXIS(2)を置換え可能
ON CHAPTERS	AXIS(3)を置換え可能

軸のキーワード別名	軸
ON SECTIONS	AXIS(4)を置換え可能

たとえば、次のクエリーでは、軸の指定は{Jan}ON COLUMNS です:

```
   SELECT
 {Jan} ON COLUMNS
FROM Sample.Basic
```

演習: 2 軸クエリーの実行

➤ 2 軸クエリーを実行するには:

1 qry_blank.txt を開きます。

2 ON COLUMNS の後ろにカンマを追加してから、ON ROWS を追加することで 2 番目の軸のプレースホルダを追加します:

```
      SELECT
 {}
ON COLUMNS,

       {}


      ON ROWS

FROM Sample.Basic
```

3 新しいクエリー・テンプレートを qry_blank_2ax.txt という名前で保存します。

4 列軸のセットの指定として、製品メンバー 100-10 および 100-20 を入力します。例:

```
      SELECT

      {[100-10],[100-20]}

ON COLUMNS,
 {}
ON ROWS
FROM Sample.Basic
```

これらのメンバー名には特殊文字が含まれているため、大カッコを使用する必要があります。メンバー名に特殊文字が含まれていない場合でも、メンバー名を大カッコで囲むという、ここで使用されているルールに従うことをお薦めします。

5 行軸のセットの指定として、年メンバー Qtr1 から Qtr4 までを入力します。

```
      SELECT
 {[100-10],[100-20]}
ON COLUMNS,
 {
      [Qtr1],[Qtr2],[Qtr3],[Qtr4]
      }
ON ROWS
FROM Sample.Basic
```

6 クエリーを qry_2ax.txt という名前で保存します。

7 演習: 初めてのクエリーの実行の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

クエリーの結果を表 111 に示します:

表 111　結果: 2 軸クエリーの実行

	100-10	100-20
Qtr1	5096	1359
Qtr2	5892	1534
Qtr3	6583	1528
Qtr4	5206	1287

演習: 単一軸上での複数の次元のクエリー

➤ 単一軸上で複数の次元をクエリーするには:

1 qry_blank_2ax.txt を開きます。

2 列軸で、2 つのタプルを指定します。各タプルは、1 つのメンバーではなく、メンバーの組合せです。各タプルで複数のメンバーを表しているので、各タプルを丸カッコで囲みます。

```
      SELECT
 {
      ([100-10],[East]), ([100-20],[East])
      }
ON COLUMNS,
 {}
ON ROWS
FROM Sample.Basic
```

3 行軸で、2 つのメンバーを持つタプルを 4 つ指定します。各 Quarter を Profit でネストします:

```
      SELECT
 {([100-10],[East]), ([100-20],[East])}
ON COLUMNS,
```

```
{
     ([Qtr1],[Profit]), ([Qtr2],[Profit]),

     ([Qtr3],[Profit]), ([Qtr4],[Profit])
}
ON ROWS
FROM Sample.Basic
```

4 クエリーを qry_1ax.txt という名前で保存します。

5 演習: 初めてのクエリーの実行の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

クエリーの結果を表 112 に示します:

表 112　結果: 単一軸上での複数の次元のクエリー

		100-10	100-20
		East	East
Qtr1	Profit	2461	212
Qtr2	Profit	2490	303
Qtr3	Profit	3298	312
Qtr4	Profit	2430	287

キューブの指定

キューブの指定は、クエリー対象のデータベースを決定するクエリーの一部です。キューブの指定は、次のように MDX クエリーに配置します:

```
SELECT <axis> [, <axis>...]

FROM <database>
```

<database>セクションは、FROM キーワードの後に置き、最初にアプリケーション名を指定し次にデータベース名を指定する、区切られたまたは区切られていない識別子で構成する必要があります。たとえば、次の指定は有効です:

- FROM Sample.Basic
- FROM [Sample.Basic]
- FROM [Sample].[Basic]
- FROM'Sample'.'Basic'

関数を使用したセットの作成

メンバーごとまたはタプルごとにセットを作成するのではなく、セットを戻す関数を使用できます。MDX には、セットを戻す関数および他の値を戻す関数が用意されています。Essbase でサポートされる MDX 関数の詳細は、『Oracle Essbase テクニカル・リファレンス』の MaxL に関する項を参照してください。

演習: MemberRange 関数の使用

MemberRange 関数では、同世代の指定された 2 つのメンバーの間(これら 2 つのメンバーを含む)の範囲のメンバーが戻されます。この構文は次のとおりです:

```
MemberRange (
member1
,
member2
, [,
layertype
])
```

指定する最初の引数は、範囲の始まりとなるメンバーで、2 番目の引数は、範囲の終わりとなるメンバーです。`layertype` 引数はオプションです。『Oracle Essbase テクニカル・リファレンス』を参照してください。

注: MemberRange のもう 1 つの構文として、関数名を使用するかわりに、`member1 : member2` のように 2 つのメンバー間にコロンを使用する方法があります。

➤ MemberRange 関数を使用するには:

1. `qry_blank.txt` を開きます。
2. 中カッコ`{}`を削除します。セットを戻す関数を使用するときには必要ありません。
3. コロン演算子を使用して、次のように Qtr1 から Qtr4 までのメンバー範囲を選択します:

```
    SELECT
    [Qtr1]:[Qtr4]
ON COLUMNS
FROM Sample.Basic
```

4. 演習: 初めてのクエリーの実行の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

   Qtr1、Qtr2、Qtr3 および Qtr4 が戻されます。

5. MemberRange 関数を使用して、同じメンバー範囲である Qtr1 から Qtr4 までを選択します。

```
        SELECT

        MemberRange([Qtr1],[Qtr4])

ON COLUMNS
FROM Sample.Basic
```

6 クエリーを gry_member_range_func.txt という名前で保存します。

7 演習: 初めてのクエリーの実行の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

結果: 手順 3 および手順 5 で示されたクエリーの実行時に、同じ結果が戻されるはずです。

演習: CrossJoin 関数の使用

CrossJoin 関数では、異なる次元の 2 つのセットのクロス積が戻されます。この構文は次のとおりです:

```
CrossJoin(
set
,
set
)
```

CrossJoin 関数では、異なる次元の 2 つのセットを入力として、2 つの入力セットのクロス積であるセットが作成されます。これは、対称型レポートを作成するときに便利です。

➤ CrossJoin 関数を使用するには:

1 qry_blank_2ax.txt を開きます。

2 列軸の中カッコ{}を CrossJoin()に置き換えます。

```
        SELECT

        CrossJoin ()

ON COLUMNS
        ,

 {}
ON ROWS
FROM Sample.Basic
```

3 CrossJoin 関数に指定する 2 つのセット引数のプレースホルダとして、中カッコ 2 つのペアをカンマで区切って追加します:

```
        SELECT
```

```
 CrossJoin (
      {}, {}
      )
ON COLUMNS,
 {}
ON ROWS
FROM Sample.Basic
```

4 最初のセットでは、製品メンバー[100-10]を指定します。2 番目のセットでは、市場メンバー[East]、[West]、[South]および[Central]を指定します。

```
      SELECT
 CrossJoin ({
      [100-10]
      }, {
      [East],[West],[South],[Central]
      })
ON COLUMNS,
 {}
ON ROWS
FROM Sample.Basic
```

5 行軸で CrossJoin を使用して、メジャーのメンバーのセットと Qtr1 が含まれるセットをクロスします:

```
       SELECT
 CrossJoin ({[100-10]}, {[East],[West],[South],[Central]})
ON COLUMNS,

       CrossJoin (


         {[Sales],[COGS],[Margin %],[Profit %]}, {[Qtr1]}



        )

ON ROWS
FROM Sample.Basic
```

6 クエリーを qry_crossjoin_func.txt という名前で保存します。

7 演習: 初めてのクエリーの実行の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

CrossJoin を使用する場合は、引数の順序が、出力のタプルの順序に影響を与えます。

クエリーの結果を表 113 に示します:

表 113　結果: CrossJoin 関数の使用

		100-10	100-10	100-10	100-10
		East	West	South	Central
Sales	Qtr1	5731	3493	2296	3425
COGS	Qtr1	1783	1428	1010	1460
Margin %	Qtr1	66.803	59.118	56.01	57.372
Profit %	Qtr1	45.82	29.974	32.448	24.613

演習: Children 関数の使用

Children 関数では、特定のメンバーのすべての子メンバーのセットが戻されます。次の構文を使用します:

```
Children (member)
```

注： Children のもう 1 つの構文として、`member.Children` のように、Children を入力メンバーで演算子のように使用する方法があります。この演習では、この演算子構文を使用します。

➤ Children 関数を使用して、最初の軸の指定にショートカットを取り入れるには、次の手順に従います:

1. `qry_crossjoin_func.txt` を開きます。
2. 列軸の指定の 2 番目のセットで、`[East],[West],[South],[Central]` を `[Market].Children` に置き換えます。

```
      SELECT
 CrossJoin ({[100-10]}, {
      [Market].Children
      })
ON COLUMNS,
 CrossJoin (
  {[Sales],[COGS],[Margin %],[Profit %]}, {[Qtr1]}
 )
ON ROWS
FROM Sample.Basic
```

3. クエリーを `gry_children_func.txt` という名前で保存します。
4. 演習: 初めてのクエリーの実行の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

結果: 表 113 で戻された結果と同じ結果が戻されるはずです。

レベルおよび世代の処理

MDX では、レイヤーは、Essbase 階層の世代とレベルを意味します。

Essbase では、世代番号のカウントは、次元名を 1 として開始され、世代番号が大きいほど階層内のリーフ・メンバーに近くなります。

レベル番号は、階層の最もリーフ側に近い部分の 0 から始まります。最上位のレベル番号は次元名です。

数多くの MDX 関数では、指定したレイヤーが入力引数として受け取られ、レイヤー引数で示された世代またはレベルに基づいて、セット操作が実行されます。

レイヤー引数を指定する方法は、次のとおりです:

- 世代名またはレベル名(例: `States`、`Regions`)。
- 世代名またはレベル名を付けた次元名(例: `Market.Regions`、`[Market].[States]`)。
- 入力として次元とレベル番号を使用する Levels 関数たとえば、`[Year].Levels(0)`。
- 入力としてメンバーを使用する Level 関数。たとえば、`[Qtr1].Level` によって、Sample.Basic の四半期のレベルが戻されます。それは、市場次元のレベル 1 です。
- 入力として次元と世代番号を使用する Generations 関数(例: `[Year].Generations (3)`)。
- 入力としてメンバーを使用する Generation 関数。たとえば、`[Qtr1].Generation` によって、Sample.Basic の四半期の世代が戻されます。それは、市場次元の世代 2 です。

注： Sample.Basic データベースでは、Qtr1 と Qtr4 は同じレイヤーにあります。つまり、Qtr1 と Qtr4 は同じ世代に属します。ただし、不規則階層を持つ別のデータベースでは、Qtr1 と Qtr4 は、同じ世代に属していても、必ずしも同じレベルにあるとは限りません。たとえば、Qtr1 の階層は週にドリル・ダウンし、Qtr4 の階層は月で止まることがあります。Qtr1 は、Qtr4 よりも 1 つレベルが高いですが、この 2 つは同じレイヤーにあります。

演習: Members 関数の使用

Members 関数を使用して、指定した世代またはレベルのすべてのメンバーを戻すことができます。この構文を layer 引数とともに使用すると、次のようになります:

```
Members ( layer )
```

指定する `layer` 引数は、戻されるメンバーの世代またはレベルを示します。

注： Members のもう 1 つの構文は、`layer.Members` です。

➤ Members 関数を使用するには:

1 qry_blank.txt を開きます。

2 中カッコ{}を削除します。

3 Members 関数と Levels 関数を使用して、Sample.Basic の市場次元のすべてのレベル 0 メンバーを選択します:

```
      SELECT

      Members(Market.levels(0))

ON COLUMNS
FROM Sample.Basic
```

4 クエリーを qry_members_func.txt という名前で保存します。

5 演習: 初めてのクエリーの実行の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

結果: 市場次元のすべての州が戻されます。

スライサ軸を使用したクエリーの視点の設定

スライサ軸は、データベースの特定の領域のみに適用されるようにクエリーを制限する方法です。オプションのスライサを使用する場合は、MDX クエリーの WHERE セクションで指定する必要があります。さらに、WHERE セクションは、キューブの指定(FROM セクション)の後に置き、クエリーの最後の構成要素にする必要があります:

```
   SELECT {
   set
   }
ON
   axes

FROM
   database

WHERE slicer
```

スライサ軸を使用して、クエリーのコンテキストを設定します。通常は、他のすべての軸のデフォルトのコンテキストです。

たとえば、クエリーで Sample.Basic データベース内の売上高実績のみ(売上高予算を除く)を選択する場合、WHERE 句は次のようになります:

```
   WHERE ([Actual], [Sales])
```

スライサ軸に(Actual, Sales)が指定されているため、ON AXIS(n)のセットの指定に含める必要はありません。

演習: スライサ軸を使用した結果の制限

➤ スライサ軸を使用して結果を制限するには:

1 gry_crossjoin_func.txt を開きます。

2 演習: 初めてのクエリーの実行の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

データ・セルの 1 つに表示される結果に注意してください。たとえば、最初のタプル([100-10],[East],[Sales],[Qtr1])の値は、5731 です。

3 戻されるデータを予算値のみに制限するために、スライサ軸を追加します。

```
      SELECT
 CrossJoin ({[100-10]}, {[East],[West],[South],[Central]})
ON COLUMNS,
 CrossJoin (
  {[Sales],[COGS],[Margin %],[Profit %]}, {[Qtr1]}
 )
ON ROWS
FROM Sample.Basic

      WHERE (Budget)
```

4 クエリーを MaxL シェルに貼り付けて実行します。

5 クエリーを qry_slicer_axis.txt という名前で保存します。

結果: 手順 1 および手順 3 で示されたクエリーに対して異なる結果が表示されるはずです。

共通関係関数

表 114 に、データベース・アウトライン内のメンバーの関係に基づいてセットを戻す関係関数をリストします:

表 114　セットを戻す関係関数のリスト

関係関数	説明
Children	入力メンバーの子が戻されます。
Siblings	入力メンバーの兄弟が戻されます。
Descendants	メンバーの子孫が戻されます。異なるオプションが使用できます。

表 115 に、セットではなく単一メンバーを戻す関係関数をリストします:

表 115　単一メンバーを戻す関係関数のリスト

関係関数	**説明**
Ancestor	指定したレイヤーに存在する祖先が戻されます。
Cousin	別の祖先のメンバーと同じ位置の子メンバーが戻されます。
Parent	入力メンバーの親が戻されます。
FirstChild	入力メンバーの最初の子が戻されます。
LastChild	入力メンバーの最後の子が戻されます。
FirstSibling	入力メンバーの親の最初の子が戻されます。
LastSibling	入力メンバーの親の最後の子が戻されます。

関係関数を使用する例は、『Oracle Essbase テクニカル・リファレンス』の MaxL に関する項に記載されている MDX の例を参照してください。

セット演算の実行

表 116 に、データベースから追加情報を抽出せずに、入力セットに作用するセット関数をリストします:

表 116　純粋なセット関数のリスト

純粋なセット関数	**説明**
CrossJoin	複数の次元のセット 2 つがクロス(交差)する部分が戻されます。
Distinct	セットから重複するタプルが削除されます。
Except	2 つのセットの相違を含むサブセットが戻されます。
Generate	反復関数です。`set1` 内のタプルごとに `set2` が戻されます。
Head	セットの先頭から `n` 個のメンバーまたはタプルが戻されます。
Intersect	2 つの入力セットの交差が戻されます。
Subset	セットからサブセットが戻されます。戻されるサブセットは、数値で指定されたタプルの範囲です。
Tail	セットの最後から `n` 個のメンバーまたはタプルが戻されます。
Union	2 つの入力セットの和集合が戻されます。

演習: Intersect 関数の使用

Intersect 関数では、2 つの入力セットの交差が戻されます。必要に応じて、重複が保持されます。構文:

```
Intersect (
set
,
set
 [,ALL])
```

Intersect 関数を使用すると、両方のセットに存在するタプルを検索して、セットを比較できます。

➤ Intersect 関数を使用するには:

1 qry_blank.txt を開きます。

2 軸から中カッコ{}を削除し、かわりに Intersect()を入力します。例:

```
     SELECT

     Intersect (

 )

ON COLUMNS
FROM Sample.Basic
```

3 Intersect 関数に指定する 2 つのセット引数のプレースホルダとして使用するために、中カッコ 2 つのペアをカンマで区切って追加します。例:

```
     SELECT
 Intersect (

     { },
 { }

 )
ON COLUMNS
FROM Sample.Basic
```

4 最初のセット引数として East の子を指定します。例:

```
     SELECT
 Intersect (
 {
     [East].children
     },
 { }
 )
ON COLUMNS
```

```
FROM Sample.Basic
```

5 2 番目のセット引数として、Major Market の UDA を持つ市場次元のすべてのメンバーを指定します。例:

```
     SELECT
 Intersect (
 { [East].children },
 {
     UDA([Market], "Major Market")
      }
 )
ON COLUMNS
FROM Sample.Basic
```

6 クエリーを qry_intersect_func.txt という名前で保存します。

7 595 ページの「演習: 初めてのクエリーの実行」の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

結果:「Major Market」の UDA を持つ East のすべての子が戻されます。例:

```
New York  Massachusetts  Florida



8202    6172      5029
```

演習: Union 関数の使用

Union 関数では、2 つの入力セットが結合されます。必要に応じて、重複が保持されます。構文:

```
Union (set, set [,ALL])
```

Union 関数を使用すると、2 つのセットを 1 つのセットにまとめることができます。

➤ Union 関数を使用するには:

1 qry_intersect_func.txt を開きます。

2 Intersect を Union に置き換えます。

3 クエリーを qry_union_func.txt という名前で保存します。

4 クエリーを MaxL シェルに貼り付けて実行します。

結果: これらのクエリーの結果と Intersect 関数を使用した結果(609 ページの「演習: Intersect 関数の使用」を参照)を比較すると、Intersect では、「Major Market」の UDA を持つ、East の子のみが含まれているセットが戻されるのに対して、Union では、(East のすべての子) + (「Major Market」の UDA を持つ市場のすべてのメンバー)が戻されることがわかります。

純粋なセット演算関数のその他の例は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

名前付きセットおよび計算済メンバーの作成と使用

計算済メンバーと名前付きセットは、クエリーの存続期間中に複数回使用できる、クエリー内の論理エンティティです。計算済メンバーと名前付きセットによって、記述されたコードの行および実行時間で時間を節約できます。MDX クエリーの先頭に置くオプションの WITH セクションでは、計算済メンバーまたは名前付きセット、あるいはその両方を定義できます。

計算済メンバー

計算済メンバーとは、クエリーの実行期間中に存在する仮想メンバーです。計算済メンバーを使用すれば、新しいメンバーをデータベース・アウトラインに追加する必要なく、複雑な分析を行えます。計算済メンバーは、実際のメンバーに対して実行された計算結果の格納場所です。

計算済メンバーの名前には次のガイドラインを使用します:

- 計算済メンバーは次元に関連付けます。たとえば、メンバー MyCalc をメジャー次元に関連付けるには、`[Measures].[MyCalc]`という名前を付けます。
- 実際のメンバーの名前を使用して、計算済メンバーに名前を付けないでください。たとえば、Sales はすでにメジャー次元に存在するため、計算済メンバーに`[Measures].[Sales]`という名前を付けないでください。

複数の計算済メンバーを使用して、比率またはカスタム合計を作成するときには、各計算済メンバーに解決順を設定することをお薦めします。解決順の詳細は、『Oracle Essbase テクニカル・リファレンス』の MDX に関する項を参照してください。

演習: 計算済メンバーの作成

この演習では、計算の共通関数である Max 関数を使用します。Max 関数では、セットのタプル内で見つかった最大値が戻されます。この構文は次のとおりです:

```
Max (
set
,
numeric_value
)
```

➤ 計算済メンバーを作成するには:

1 qry_blank_2ax.txt を開きます。

2 行軸のセットで Product の子を指定します。例:

```
     SELECT
 {}
ON COLUMNS,
 {[
     Product].children
     }
ON ROWS
FROM Sample.Basic
```

3 クエリーの先頭に、計算済メンバーの指定のプレースホルダを追加します。例:

```
     WITH MEMBER [].[]
AS ''

SELECT
 {}
ON COLUMNS,
 {[Product].children}
ON ROWS
FROM Sample.Basic
```

4 計算済メンバーをメジャー次元に関連付け、メンバー名を Max Qtr2 Sales にするには、この情報を計算済メンバーの指定に追加します。例:

```
     WITH MEMBER [
     Measures
     ].[
     Max Qtr2 Sales
     ]
 AS ''
SELECT
 {}
ON COLUMNS,
 {[Product].children}
ON ROWS
FROM Sample.Basic
```

5 AS キーワードの後の一重引用符の内側に、Max Qtr2 Sales という名前の計算済メンバーの論理を定義します。

Max 関数で、セットを使用して(Qtr)を最初の引数として評価し、メジャーを使用して(Sales)を 2 番目の引数として評価します。例:

```
     WITH MEMBER [Measures].[Max Qtr2 Sales]
 AS '

     Max (
```

```
        {[Year].[Qtr2]},

        [Measures].[Sales]

       )
      '
SELECT
 {}
ON COLUMNS,
 {[Product].children}
ON ROWS
FROM Sample.Basic
```

6 計算済メンバー Max Qtr2 Sales は、WITH セクションに定義されています。クエリーで使用するには、クエリーの SELECT 部分の軸のいずれかに、この計算済メンバーを追加します。列軸で事前定義の計算済メンバーを選択します。例:

```
     WITH MEMBER [Measures].[Max Qtr2 Sales]
 AS '
 Max (
  {[Year].[Qtr2]},
  [Measures].[Sales]
 )'
SELECT
 {
      [Measures].[Max Qtr2 Sales]
      }
ON COLUMNS,
 {[Product].children}
ON ROWS
FROM Sample.Basic
```

7 クエリーを gry_calc_member.txt という名前で保存します。

8 595 ページの「演習: 初めてのクエリーの実行」の説明のとおりに、クエリーを MaxL シェルに貼り付けて、実行します。

クエリーの結果を表 117 に示します:

表 117　結果: 計算済メンバーの作成

	Max Qtr2 Sales
100	27187
200	27401
300	25736
400	21355

	Max Qtr2 Sales
Diet	26787

名前付きセット

名前付きセットは、クエリーの WITH セクションに定義します。定義すると、クエリーの SELECT セクションを作成するときにセットを名前で参照できるので便利です。

たとえば、名前付きセット Best5Prods は、12 月に最もよく売れている 5 つの製品のセットを識別します:

```
   WITH
SET
   [Best5Prods]

 AS
 'Topcount (
  [Product].members,
  5,
  ([Measures].[Sales], [Scenario].[Actual], [Year].[Dec])
 )'
SELECT [
   Best5Prods
   ] ON AXIS(0),
 {[Year].[Dec]} ON AXIS(1)
FROM Sample.Basic
```

反復関数の使用

表 118 に、データのセットがループされて、指定した検索条件と結果が実行される関数をリストします:

表 118　反復関数のリスト

関数	説明
Filter	検索条件の値が TRUE であるセットの、タプルのサブセットが戻されます。
IIF	条件付きテストが実行されます。テストが TRUE または FALSE のいずれに評価されるかに応じて、適切な数値式またはセットが戻されます。
Case	条件付きテストが実行され、指定した結果が戻されます。
Generate	反復関数です。set1 内のタプルごとに set2 が戻されます。

Filter 関数の例

次のクエリーでは、式 IsChild([Market].CurrentMember,[East]) によって TRUE が戻される市場次元のすべてのメンバーが戻されます。つまり、このクエリーでは、East のすべての子が戻されます。

```
   SELECT
 Filter([Market].Members,
  IsChild([Market].CurrentMember,[East])
 )
ON COLUMNS
FROM Sample.Basic
```

MDX 内の Filter 関数は、レポート・ライターの RESTRICT コマンドと互換性があります。

Filter 関数および他の反復関数のその他の例は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

欠落データの処理

データベースをクエリーする場合、軸の指定の先頭に NON EMPTY キーワードを使用すると、値を含まないセルをクエリー結果から除外できます。

NON EMPTY を含む軸の指定の構文は、次のとおりです:

```
   <axis_specification> ::=

   [NON EMPTY]
    <set> ON
 COLUMNS | ROWS | PAGES | CHAPTERS |
 SECTIONS | AXIS (<unsigned_integer>)
```

軸でのセットの指定の前にオプションのキーワード NON EMPTY を指定すると、この軸で#MISSING 値全体を含むスライスが抑制されます。

軸上の特定のタプル(`(Qtr1, Actual)`など)の場合、スライスは、このタプルと他のすべての軸のすべてのタプルの組合せからなるセルで構成されます。これらのセルの値がすべて#MISSING の場合、NON EMPTY キーワードによってそのタプルが削除されます。

たとえば、行のいずれの値も空でない場合は、その行全体が戻されます。行軸指定の先頭に NON EMPTY を挿入すると、クエリーで戻されるセットから次の行スライスが削除されます:

Qtr1					
実績	#MISSING	#MISSING	#MISSING	#MISSING	#MISSING

NON EMPTY を使用して欠落した値を抑制する他にも、#MISSING の結果を処理する関数として、次の MDX 関数があります:

- CoalesceEmpty は、数値式から#MISSING 値以外を探します。
- IsEmpty は、入力の数値式の評価結果の値が#MISSING の場合に、TRUE を戻します。

- Avg は、オプションの IncludeEmpty フラグが使用されていないかぎり、欠落した値を平均から除外します。

NonEmptyCount MDX 関数は、セットのタプルのうち、値が#Missing 値でないと評価されたタプル数のカウントを戻します。タプルごとに評価され、この関数によって戻されるカウントに含まれます。数値式が指定されている場合は、すべてのタプルとの関連で式が評価され、#Missing でない値のカウントが戻されます。

集約ストレージ・データベースの場合のみ、NonEmptyCount MDX 関数が最適化されるため、データベースを一度スキャンするだけですべてのセルの個別カウントを計算できます。この最適化がない場合、個別カウントに対応するセルの数と同じ回数データベースがスキャンされます。アウトライン・メンバー式に次の構文がある場合、NonEmptyCount 最適化がトリガーされます:

```
NONEMPTYCOUNT(
set
,
measure
, exclude_missing)
```

exclude_missing パラメータは、個別カウント計算を実行するメトリックを問い合せるクエリーのパフォーマンスを向上させることで、集約データベースでの NonEmptyCount 最適化をサポートします。

NONEMPTYMEMBER および NONEMPTYTUPLE プロパティを使用すると、Essbase の MDX では、メンバーまたはタプルの大きなセットでクエリーを実行でき、一方、#MISSING データのみが含まれた、使用しない値での式の実行はスキップできます。

- **nonempty_member_list** で指定されたメンバーのいずれかが空のとき、式または計算済メンバーの値が空であることを Essbase に通知するには、計算済メンバーまたは式の先頭で NONEMPTYMEMBER プロパティ句を単独で使用します。
- nonempty_member_list で指定されたタプルのセル値が空のとき、式または計算済メンバーの値が空であることを **Essbase** に通知するには、計算済メンバーまたは式の先頭で NONEMPTYTUPLE プロパティ句を単独で使用します。

入力セットが指定されている場合、NonEmptySubset MDX 関数では、すべてのタプルが空でないと評価される入力セットのサブセットが戻されます。空でないことの確認に対してオプションの値式を指定できます。この関数は、空でない組合せのセットが小規模であることがわかっている大規模なセットに基づいたクエリーを最適化する際に役立つことがあります。NonEmptySubset を使用すると、メトリックが存在する場合にセットのサイズが減少します; たとえば、特定の Units に対して空でない子孫のサブセットを要求できます。

詳細および例は、『Oracle Essbase テクニカル・リファレンス』の MDX に関する項を参照してください。

MDX クエリーでの代替変数の使用

代替変数は、定期的に変化する情報のグローバル・プレースホルダとして機能します。Administration Services、MaxL または ESSCMD を使用して、代替変数をサーバーに設定し、各変数に値を割り当てます。その値はいつでも変更できます。代替変数を設定するにはデータベース・マネージャ以上の役割が必要です。118 ページの「代替変数の使用」を参照してください。

➤ MDX 式で代替変数を使用する場合は、次の点に注意してください:

- 代替変数は、クエリーを実行するアプリケーションやデータベースからアクセス可能でなければなりません。
- 代替変数は、名前と値の 2 つのコンポーネントで構成されます。
- 変数名は英数字の組合せであり、最大サイズは付録 A「制限」で指定されています。MDX で使用する代替変数名にスペース、句読点、大カッコ([ ])を使用しないでください。
- 式の中では変数を使用する場所で、変数名の前にアンパサンド(&)を付けて指定します。たとえば、サーバー上に設定した代替変数の名前が CurMonth である場合は、MDX 式では、&CurMonth と指定します。
- 取得を実行すると、Essbase によって変数名が代替値に置換されて、MDX 式ではこの値が使用されます。

たとえば、次のように変数名 CurQtr の先頭に&を付けた式を作成します:

```
   SELECT
 {[
   &CurQtr
   ]}
ON COLUMNS
FROM Sample.Basic
```

この式を実行するときは、変数名が現在の値(Qtr1)に置換されるので、次の式が実行されます:

```
   SELECT
 {[
   Qtr1
   ]}
ON COLUMNS
FROM Sample.Basic
```

プロパティのクエリー

MDX では、プロパティでデータとメタデータの特性を記述します。MDX では、プロパティを使用してデータを取得および分析するクエリーを作成できます。プロパティは、固有かカスタムかのいずれかです。

組込みプロパティは、すべての次元のメンバーに対して定義されます。Essbase のデータベース・アウトラインのすべてのメンバーに対して定義されている組込みメンバー・プロパティは、MEMBER_NAME、MEMBER_ALIAS、LEVEL_NUMBER およびGEN_NUMBER です。

Essbase の MDX では、属性プロパティと UDA プロパティという 2 種類のカスタム・プロパティをサポートします。属性プロパティは、アウトラインの属性次元によって定義されます。Sample.Basic データベースのパッケージ・タイプ属性次元は、製品次元のメンバーの容器特性を表しています。この情報は、プロパティ名[Pkg Type]を使用して MDX でクエリーできます。

属性プロパティは、特定の次元に対してのみ、および各次元の特定のレベルに対してのみ定義されます。たとえば、Sample.Basic アウトラインでは、[Ounces]は、製品次元のメンバーに対してのみ定義された属性プロパティで、このプロパティは、製品次元のレベル 0 メンバーにのみ有効な値です。[Ounces]プロパティは、市場などの他の次元にはありません。製品次元の非レベル 0 メンバーの[Ounces]プロパティは、NULL 値です。アウトラインの属性プロパティは、そのアウトラインにある属性次元の名前によって識別されます。

カスタム・プロパティには UDA も含まれます。たとえば、[Major Market]は、市場次元のメンバーに対して定義された UDA プロパティです。[Major Market] UDA がメンバーに対して定義されている場合は TRUE 値が戻され、それ以外の場合は FALSE が戻されます。

メンバー・プロパティのクエリー

MDX クエリー内部でプロパティを使用するには、2 つの方法があります。

- 軸セットごとに次元とプロパティの組合せをリストできます。クエリーを実行すると、指定したプロパティが、指定した次元のすべてのメンバーに対して評価され、結果セットに追加されます。

  たとえば、次のクエリーによって、列軸では市場次元のすべてのメンバーの GEN_NUMBER 情報が戻されます。行軸では、クエリーによって各製品次元メンバーに対して MEMBER_ALIAS 情報が戻されます。

```
    SELECT
 [Market].Members
  DIMENSION PROPERTIES [Market].[GEN_NUMBER] on columns,
 Filter ([Product].Members, Sales > 5000)
  DIMENSION PROPERTIES [Product].[MEMBER_ALIAS] on rows
FROM Sample.Basic
```

  軸の DIMENSION PROPERTIES セクションを使用してメンバー・プロパティをクエリーするときに、プロパティは、次元名とプロパティ名によって識別するか、プロパティ名のみを使用して識別できます。プロパティ名が単独で使用される場合、そのプロパティが適用される、対象軸上のすべての次元からのすべてのメンバーに対してプロパティ情報が戻されます。次のクエリーでは、MEMBER_ALIAS プロパティは、年次元と製品次元に関して行軸で評価されます。

```
     SELECT [Market].Members
 DIMENSION PROPERTIES [Market].[GEN_NUMBER] on columns,
 CrossJoin([Product].Children, Year.Children)
  DIMENSION PROPERTIES [MEMBER_ALIAS] on rows
FROM Sample.Basic
```

- プロパティは、MDX クエリーの値式の内部で使用できます。たとえば、入力セットのメンバーのプロパティを使用する値式に基づいて、セットをフィルタできます。

  次のクエリーでは、缶にパッケージされたすべてのカフェイン製品が戻されます。

```
     SELECT
 Filter([Product].levels(0).members,
  [Product].CurrentMember.Caffeinated and
  [Product].CurrentMember.[Pkg Type] = "Can")
   xDimension Properties
   [Caffeinated], [Pkg Type] on columns
FROM Sample.Basic
```

  次のクエリーでは、UDA [Major Market]を使用し、現在の Market が主要な市場であるかどうかに基づいて値[BudgetedExpenses]が計算されます。

```
     WITH
 MEMBER [Measures].[BudgetedExpenses] AS
  'IIF([Market].CurrentMember.[Major Market],
  [Marketing] * 1.2, [Marketing])'

SELECT  {[Measures].[BudgetedExpenses]} ON COLUMNS,
 [Market].Members ON ROWS
FROM Sample.Basic
WHERE ([Budget])
```

プロパティの値のタイプ

Essbase の MDX プロパティの値タイプは、数値、ブール、文字列のいずれかです。MEMBER_NAME および MEMBER_ALIAS プロパティでは、文字列値が戻されます。LEVEL_NUMBER および GEN_NUMBER プロパティでは、数値が戻されます。

属性プロパティでは、属性次元のタイプに基づいて、数値、ブール値、文字列値のいずれかが戻されます。たとえば、Sample.Basic の[Ounces]属性プロパティは、数値プロパティです。[Pkg Type]属性プロパティは、文字列プロパティです。[Caffeinated]属性プロパティは、ブール・プロパティです。

Essbase では、属性次元に日付型を使用できます。日付型プロパティは、MDX で数値プロパティとして処理されます。このプロパティ値を日付と比較するときには、TODATE 関数を使用して、比較する前に日付文字列を数値に変換します。

次のクエリーでは、2007 年 3 月 25 日に発売された製品次元のすべてのメンバーが戻されます。プロパティ[Intro Date]は日付型であるため、比較を行う前に TODATE 関数を使用して、日付の文字列「03-25-2007」を数値に変換する必要があります。

```
   SELECT
 Filter ([Product].Members,
  [Product].CurrentMember.[Intro Date] =
  TODATE("mm-dd-yyyy","03-25-2007"))ON COLUMNS
FROM Sample.Basic
```

プロパティを値式で使用する場合は、その値の型(文字列、数値またはブール)に基づいて適切にプロパティを使用する必要があります。

数値の範囲を持つ属性次元をクエリーすることもできます。

次のクエリーでは、Small、Medium および Large という人口範囲の Sales データが取得されます。

```
   SELECT
 {Sales} ON COLUMNS,
 {Small, Medium, Large} ON ROWS
FROM Sample.Basic
```

値式で属性をプロパティとして使用する場合は、範囲メンバーを使用して、メンバーのプロパティ値が特定の範囲内であるかどうかを確認できます。この場合は IN 演算子を使用します。

たとえば、次のクエリーでは、人口範囲が Medium である市場次元のすべてのメンバーが戻されます:

```
   SELECT
 Filter(
  Market.Members, Market.CurrentMember.Population
  IN "Medium"
 )
ON AXIS(0)
FROM Sample.Basic
```

NULL プロパティ値

すべてのメンバーが特定のプロパティ名に対して有効な値を持っているとは限りません。たとえば、MEMBER_ALIAS プロパティを指定した場合、アウトラインに定義されているとおりにメンバーの別名が戻されます。ただし、すべてのメンバーに別名が定義されているとは限りません。この場合、別名が定義されていないメンバーに関しては NULL 値が戻されます。

次のクエリーを見てください。

```
   SELECT
 [Year].Members
 DIMENSION PROPERTIES [MEMBER_ALIAS]
```

```
ON COLUMNS
FROM Sample.Basic
```

年次元のどのメンバーにも別名が定義されていません。したがって、クエリーでは、年次元のメンバーの MEMBER_ALIAS プロパティについて NULL 値が戻されます。

属性プロパティは、特定の次元のメンバーおよびその次元の特定のレベルに対して定義されます。Sample.Basic データベースでは、[Ounces]プロパティは、製品次元のレベル 0 メンバーに対してのみ定義されています。

したがって、次のクエリーに示すように、市場次元のメンバーの[Ounces]プロパティをクエリーすると、構文エラーとなります:

```
   SELECT
 Filter([Market].members,
  [Market].CurrentMember.[Ounces] = 32) ON COLUMNS
FROM Sample.Basic
```

その次元の非レベル 0 メンバーの[Ounces]プロパティをクエリーすると、NULL 値が戻されます。

値式でプロパティ値を使用する場合、関数 IsValid()を使用して、NULL 値があるかどうかを確認できます。次のクエリーでは、NULL 値を持つメンバーが削除されてから、[Ounces]プロパティ値 12 を持つ製品次元のすべてのメンバーが戻されます。

```
   SELECT
 Filter([Product].Members,
  IsValid([Product].CurrentMember.[Ounces]) AND
  [Product].CurrentMember.[Ounces] = 12)
ON COLUMNS
FROM Sample.Basic
```

36 データ・サブセットのコピーとデータのエクスポート

この章の内容

データベース・サブセットの作成プロセス

移動するデータの出力ファイルを抽出することで、Essbase データベース間または他のプログラムにデータを移動できます。他のほとんどのプログラムのインポート・フォーマット指定を満たすには、レポート・スクリプトまたは計算スクリプトを使用してテキスト・ファイルを作成します。

ここでは、ある Essbase サーバー(Essbase サーバー 1)から別の Essbase サーバー(Essbase サーバー 2)にデータベース・サブセットをコピーするプロセスについて説明します。

1. Essbase サーバー 2 で、データベース・サブセットを入れるアプリケーションとデータベースを作成します。

   624 ページの「アプリケーションおよびデータベースの作成」を参照してください。

2. アウトライン・ファイル(たとえば、`source_dbname.otl`)を、Essbase サーバー 1 にあるソース・データベースから Essbase サーバー 2 にある新しいデータベースにコピーします。

   Essbase サーバー 2 のデータベースの名前と一致するように、アウトライン・ファイルの名前を変更(たとえば、`target_dbname.otl`)して、ターゲット・データベースの既存のアウトライン・ファイルを上書きする必要がある場合もあります。

   624 ページの「ソース・データベースからのアウトライン・ファイルのコピー」を参照してください。

3. 必要なデータ・サブセットが含まれる出力ファイル(たとえば、プレーン・テキスト・ファイル)を作成します。

   625 ページの「必要なデータ・サブセットが含まれる出力ファイルの作成」を参照してください。

4. 出力ファイルを、先に作成した新しいデータベースにロードします。

   627 ページの「新しいデータベースへの出力ファイルのロード」を参照してください。

必要に応じて、手順 3 および 4 を繰り返して、Essbase サーバー 2 のデータベースから出力ファイルを作成し、別のコンピュータにあるメイン Essbase データベースにデータをロードします。

次の項の例は、Sample.Basic データベースに基づいています。例のデータ・サブセットは、West 市場の Actual, Measures データです。この例では、データ・サブセットを Essbase サーバー 2 および West Westmkts データベースにコピーします。

アプリケーションおよびデータベースの作成

アプリケーションとデータベースを Essbase サーバー 2 に作成します。必要なデータ・サブセットをこの新しいデータベースにコピーします。このアプリケーションとデータベースには任意の名前を付けることができます。

➤ アプリケーションとデータベースを作成する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アプリケーションの作成」と「データベースの作成」を参照してください。

新しい空のデータベースが起動していないことを確認します。

➤ データベースを停止する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「データベースの停止」を参照してください。

ソース・データベースからのアウトライン・ファイルのコピー

ソース・データベースのアウトライン・ファイル(.otl)を、新しく作成したデータベースにコピーします。この例では、Sample.Basic データベースの `basic.otl` アウトライン・ファイルをコピーして、Essbase サーバー 2 でそのファイル名を `wesmkts.otl` に変更します。

アウトライン・ファイルのコピー方法は、Essbase サーバー 2 からソース Essbase データベースに接続できるかどうかによって異なります。

- 接続可能な場合は、次のいずれかの方法を使用してアウトラインをコピーします:

ツール	トピック	場所
Administration Services	アウトラインのコピー	Oracle Essbase Administration Services Online Help
MaxL	**create database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYDB	『Oracle Essbase テクニカル・リファレンス』

- 接続できない場合は、オペレーティング・システムを使用して、アウトライン・ファイルをコピーします。
  1. オペレーティング・システムを使用してソース・アウトライン・ファイルをコピーします。たとえば、`basic.otl` を `westmkts.otl` にコピーします。

2. コピーしたアウトラインに、新規データベースと完全に同一の名前を付けます。
3. Essbase サーバー 2 コンピュータ上の ARBORPATH/app/appname/dbname ディレクトリにアウトライン・ファイルを保存します。ARBORPATH は、Essbase をインストールしたディレクトリで、appname と dbname は、新しく作成したアプリケーションとデータベースです。

   たとえば、basic.otl をディスクにコピーし、その名前を westmkts.otl に変更します。次に、ディスクの westmkts.otl を Essbase サーバー 2 の次のディレクトリにコピーします:

   ```
   Oracle/Middleware/EPMSystem11R1/products/Essbase/EssbaseServer/app/west/westmkts/westmkts.otl
   ```

   これで、既存の空の westmkts.otl ファイルを上書きしても安全です。

   注： 既存の空のアウトライン・ファイル(Essbase で新しいアプリケーションとデータベースを作成したときに自動的に作成されたもの)を、新しいアウトライン・ファイルで上書きしていることを確認します。
4. 新しいデータベースを停止して再起動します。

   Oracle Essbase Administration Services Online Help の「データベースの開始」と「データベースの停止」を参照してください。

これで、Essbase サーバー 2 にデータベース・アウトラインのコピーが作成されました。

必要なデータ・サブセットが含まれる出力ファイルの作成

必要なデータ・サブセットが含まれる出力ファイルを作成します。出力ファイルは、テキスト・ファイルかスプレッドシート・ファイルのいずれかです。次のいずれかの方法を使用して、データ・サブセットを作成します。

➤ データベースのサブセットの出力ファイルを作成するには、次のツールを使用します:

次の例では、レポート・ライターを使用して、Westmkts データベースのサブセットを作成します。レポート・スクリプトを使用して、データ・サブセットをエクスポートすることもできます。

➤ 必要なデータ・サブセットが含まれるテキスト・ファイルを作成するには:

1 **ソース・データベースを選択します。たとえば、West Westmkts を選択します。**

Oracle Essbase Administration Services Online Help の「オブジェクトのナビゲートと選択」を参照してください。

- Essbase サーバー 2 コンピュータから Essbase サーバー 1 データベースに接続できる場合は、Essbase サーバー 2 からソース・データベースを選択できます。
- 接続できない場合は、Essbase サーバー 2 コンピュータ以外のコンピュータを使用して、ソース・データベースを選択します。

2 **レポートを作成します。**

Oracle Essbase Administration Services Online Help の「スクリプトの作成」を参照してください。

3 **必要なデータ・サブセットを選択するレポート・スクリプトを作成します。レポート・スクリプトの作成方法は、第 33 章「レポート・スクリプトの基本の理解」を参照してください。**

たとえば、次のレポート・スクリプトでは、Sample.Basic の West 市場の Actual, Measures データを選択します:

```
     {TABDELIMT}
QUOTEMBRNAMES
Actual
<IDESC West
<IDESC Measures
```

- データ間にタブ・ストップを配置するには、スペースではなく TABDELIMIT を使用し、メンバー名またはデータ値が切り捨てられないようにしてください。
- QUOTEMBRNAMES 使用して、空白スペースが含まれるメンバー名を引用符(" ")で囲みます。これによって、Essbase でデータのロード時にメンバー名が認識されます。

4 **レポート・スクリプトを実行します。**

Oracle Essbase Administration Services Online Help の「レポート・スクリプトの実行」を参照してください。

5 **レポート・スクリプトを、`westout.txt` などの `.txt` 拡張子を付けて保存します。**

データをロードするには、出力ファイルが Essbase サーバー 2 の `ARBORPATH/app/appname/dbname` ディレクトリに存在する必要があります。`appname` と `dbname` は、新しく作成したアプリケーションとデータベースのディレクトリです。

Essbase サーバー 2 コンピュータを使用している場合は、`ARBORPATH/app/appname/dbname` ディレクトリに出力ファイルを直接保存できます。次に例を示します:

```
     Oracle/Middleware/EPMSystem11R1/products/Essbase/EssbaseServer/app/west/
westmkts/westout.txt
```

Essbase サーバー 2 コンピュータを使用していない場合は、現在のコンピュータの任意の場所に出力ファイルを保存します。デフォルトでは、Essbase では、サーバーではなく、Essbase クライアント・コンピュータにファイルを保存し

ます。レポートを実行するときには、オペレーティング・システムを使用して、ファイルを Essbase サーバー 2 の ARBORPATH/app/appname/dbname ディレクトリにコピーします。たとえば、ディスクを使用してファイルをコピーします。

Essbase サーバー 2 コンピュータを使用していない場合は、Essbase クライアントのディレクトリからファイルをダウンロードし、Essbase サーバー 2 の ARBORPATH/app/appname/dbname ディレクトリにコピーします。たとえば、出力ファイルを次のディレクトリにコピーします:

```
Oracle/Middleware/EPMSystem11R1/products/Essbase/EssbaseServer/app/west/
westmkts/westout.txt
```

新しいデータベースにテキスト・ファイルをロードする準備が整いました。

新しいデータベースへの出力ファイルのロード

出力ファイルを、Essbase サーバー 2 にある新しいデータベースにロードします。

➤ ファイルをデータベースにロードする方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「データ・ロードまたは次元構築の実行」を参照してください。

次の例は、Westmkts データベースにデータをロードする手順を示しています:

1. 新しいデータベースを選択します。たとえば、Westmkts を選択します。
2. 作成した westout などのテキスト・ファイルを使用して、データ・ロードを開始します。

注: westout が表示されない場合は、そのファイルに.txt 拡張子が付いていて、ARBORPATH/app/West/Westmkts ディレクトリに保管されていることを確認します。625 ページの「必要なデータ・サブセットが含まれる出力ファイルの作成」を参照してください。

第 20 章「データ・ロードまたは次元構築の実行およびデバッグ」を参照してください。

これで、Essbase サーバー 2 コンピュータでデータを表示できるようになりました。場合によっては、データベース・サブセットを再計算する必要があります。データベース・サブセットを表示しているので、データ値のパーセンテージは #MISSING です。

必要であれば、作成したデータベース・サブセットで使用するレポート・スクリプトやその他のアーティファクト・ファイルを Essbase サーバー 2 コンピュータにコピーできます。

データのエクスポート

➤ Essbase データベース間でデータを移動するためにデータベースからデータをエクスポートする場合、または Essbase 以外のプログラムで読取り可能なフォーマットにデータをエクスポートする場合は、次のツールを使用します:

ツール	トピック	場所
Smart View	「クエリー・デザイナの使用」	Oracle Hyperion Smart View for Office User's Guide
MaxL	**export data**	Oracle Essbase Technical Reference
計算スクリプト	498 ページの「DATAEXPORT コマンドを使用したデータのエクスポート」	Oracle Essbase Database Administrator's Guide
レポート・スクリプト	630 ページの「レポート・スクリプトを使用したテキスト・データのエクスポート」	Oracle Essbase Database Administrator's Guide

注： MaxL、計算スクリプトまたは管理サービス・コンソールを使用してテキスト・データをエクスポートするときに、最後のレコードの終了フィールドに値がない場合は、これらのフィールドに#MI などの欠落した値のマーカーは追加されません。全体の処理時間を向上させるため、最後の出力レコードではフィールドが空白のまま処理されます。

Unicode モードのアプリケーションでは、データベースからのエクスポート・ファイルは UTF-8 エンコード方式になります。Essbase によって、エクスポート・ファイル用のディスク・スペースが足りないことが検出された場合、データはエクスポートされません。

Essbase データを特殊な区切り記号が必要なプログラムにインポートする場合は、レポート・ライターの MASK コマンドまたは DATAEXPORT 計算コマンドの区切り記号パラメータを使用して、区切り記号を指定します。

バックアップするためにデータをエクスポートする方法の詳細は、『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

レポート・スクリプトまたは計算スクリプトを使用して、テキスト・フォーマットで Essbase データをエクスポートできます。両方のツールでは、ほとんどのプログラムのインポート・フォーマット指定に適合したテキスト・ファイルを作成できます。

MaxL を使用したデータのエクスポート

export data MaxL ステートメントを使用して、Essbase データベースからデータをエクスポートできます。データは、連続してエクスポートすることも、並列してエクスポートすることも可能です。スレッドのデータが 2GB を超える場合、Essbase によってエクスポート・データが複数のファイルに分割され、ファイル名には数値が追加されます。

- ブロック・ストレージ・データベース: 計算値を含まないすべてのデータ、レベル 0 データまたは入力レベル・データをエクスポートできます。

  データを並列エクスポートするには、カンマで区切られたエクスポート・ファイルのリスト(最大 1024 のファイル名まで)を指定します。ファイル名の数により、エクスポート・スレッドの数が決定されます。使用可能なブロック・アドレス範囲の数により、Essbase が実際に使用するエクスポート・スレッドの数が制限されます。Essbase では、実際のデータ・ブロックの数を、指定されたファイル名(エクスポート・スレッド)の数で除算します。実際のデータ・ブロックが、指定されたエクスポート・スレッドの数より少ない場合、作成されるエクスポート・スレッドの数は、実際のデータ・ブロックの数に基づきます。この方法によって、エクスポート・スレッド間でデータ・ブロックがより均等に配分されます。

  注： エクスポート・ファイルの数を指定する際には、Essbase サーバーが稼働しているコンピュータの使用可能な CPU コアの数および I/O 帯域幅を考慮することが重要です。指定する値が大きすぎると、パフォーマンスが低下する場合があります。

- 集約ストレージ・データベース: 計算値を含まない、レベル 0 データのみをエクスポートできます。(レベル 0 データは、集約ストレージ・データベースでの入力データと同じです。)集約ストレージ・データベースでは、上位レベルのデータのエクスポート、または縦欄式のエクスポートは実行できません。

  データを並列でエクスポートするには、カンマで区切られたエクスポート・ファイルのリスト(1 から 8 までのファイル名)を指定します。Essbase で使用されるスレッドの数は、通常、指定するファイル名の数によって異なります。ただし、データ・ブロックの少ない、非常に小規模な集約ストレージ・データベースでは、複数ファイルへの並列エクスポートが要求された場合でも、単一のファイルのみが作成される(実際にはシリアル・エクスポートが実行される)可能性があります。

絶対パスが指定されていない場合、エクスポート・ファイルはサーバーの ARBORPATH/app ディレクトリに保管されます。

計算スクリプトを使用したテキスト・データのエクスポート

次の計算コマンドを使用して、テキスト・インポート・ファイルを選択およびフォーマットできます: DATAEXPORT、DATAEXPORTCOND、SET DATAEXPORTOPTIONS、FIX...ENDFIX および EXCLUDE...ENDEXCLUDE。計算スクリプトの作成と実行の一般的な情報は、第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。

計算スクリプトベースのデータ・エクスポートは、保管され、動的に計算されるメンバーに対してのみ使用でき、レポート・スクリプトよりもフォーマット・オプションの数が少ないです。ただし、計算スクリプトベースのデータ・エクスポートでは、レポート・スクリプトよりも速く処理できる小数ベースおよび精度ベースのフォーマット・オプションを使用できます。DATAEXPORT 計算コマンドを使

用すると、通常途中で必要なインポート手順を実行せずにリレーショナル・データベースに直接エクスポートできます。

次の計算スクリプト例では、データベースのサブセットを含んだテキスト・ファイルが生成されます。

```
   SET DATAEXPORTOPTIONS
{  DATAEXPORTLEVEL "ALL";
DATAEXPORTCOLFORMAT ON;
DATAEXPORTCOLHEADER Scenario;
};
FIX ("100-10","New York","Actual","Qtr1");
 DATAEXPORT "File" "," "C:\exports\actual.txt" "NULL";
ENDFIX;
```

これらのコマンドでは、すべてのレベルのデータを含めるように指定し、列でデータが繰り返されることを示し、シナリオ次元を出力用の密次元の列ヘッダーとして設定しています。FIX コマンドでは、データ・スライスを定義し、データをテキスト・ファイルとして `C:\exports\actual.txt` にエクスポートすることを定義しています。区切り記号としてカンマが使用され、欠落したデータ値は連続する区切り記号によって示されます。Sample.Basic に対してこのスクリプトを実行すると、次のデータが生成されます:

```
   "Actual"
"100-10","New York","Sales","Qtr1",1998
"100-10","New York","COGS","Qtr1",799
"100-10","New York","Margin","Qtr1",1199
"100-10","New York","Marketing","Qtr1",278
"100-10","New York","Payroll","Qtr1",153
"100-10","New York","Misc","Qtr1",2
"100-10","New York","Total Expenses","Qtr1",433
"100-10","New York","Profit","Qtr1",766
"100-10","New York","Opening Inventory","Qtr1",2101
"100-10","New York","Additions","Qtr1",2005
"100-10","New York","Ending Inventory","Qtr1",2108
"100-10","New York","Margin %","Qtr1",60.01001001001001
"100-10","New York","Profit %","Qtr1",38.33833833833834
"100-10","New York","Profit per Ounce","Qtr1",63.83333333333334
```

DATAEXPORT 計算コマンドの詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

レポート・スクリプトを使用したテキスト・データのエクスポート

レポート・ライターでは、柔軟に出力データを選択し、他のプログラムの要件に合わせてそのデータをフォーマットできます。レポート・スクリプトは、すべてのメンバー・タイプ(たとえば、保管済メンバー、動的計算メンバー、属性、ラベルのみメンバー、共有メンバー、別名)に使用できます。ただし、レポート・スクリプトは、クエリーベースのデータ抽出(データベースに生成されていないデー

タ・ブロックを詳しく調べる方法)を使用するので、一般的に速度が遅くなります。Oracle Essbase Administration Services Online Help の「レポート・スクリプトの実行」および 551 ページの「レポート・スクリプトの実行」を参照してください。レポート・スクリプトベースのエクスポートでは、より柔軟にデータをフォーマットできます。このエクスポート方法は、プロ級のレポートを作成するのに最適です。

2 次元(固定フィールド・フォーマット)を使用するプログラムにデータをエクスポートする場合は、ページ次元または列次元を指定する必要がありません。2 次元レポートを作成するときには、すべての次元を行次元として指定できます。ROWREPEAT コマンドを使用して、指定した各メンバーの名前を各行に追加します(デフォルトのネストしたスタイルは使用しません)。次のスクリプト例とレポートでは、この状態の 5 次元データベースを示しています:

```
   <ROW (Year, Measures, Product, Market, Scenario)
{ROWREPEAT}
<ICHILDREN Year
Sales
<ICHILDREN "400"
East
Budget
  !
```

次のレポートが生成されます:

```
   Qtr1      Sales     400-10    East    Budget    900
Qtr1      Sales     400-20    East    Budget   1,100
Qtr1      Sales     400-30    East    Budget    800
Qtr1      Sales      400     East    Budget   2,800
Qtr2      Sales     400-10    East    Budget   1,100
Qtr2      Sales     400-20    East    Budget   1,200
Qtr2      Sales     400-30    East    Budget    900
Qtr2      Sales      400     East    Budget   3,200
Qtr3      Sales     400-10    East    Budget   1,200
Qtr3      Sales     400-20    East    Budget   1,100
Qtr3      Sales     400-30    East    Budget    900
Qtr3      Sales      400     East    Budget   3,200
Qtr4      Sales     400-10    East    Budget   1,000
Qtr4      Sales     400-20    East    Budget   1,200
Qtr4      Sales     400-30    East    Budget    600
Qtr4      Sales      400     East    Budget   2,800
 Year     Sales     400-10    East    Budget   4,200
 Year     Sales     400-20    East    Budget   4,600
 Year     Sales     400-30    East    Budget   3,200
 Year     Sales      400     East    Budget  12,000
```

最下位レベル(レベル 0)データのみを含む 2 次元レポートを作成するには、CHILDREN または DIMBOTTOM を使用して、レベル 0 メンバーを選択します。

- 特定のメンバーのレベル 0 のデータのみをリストするには、CHILDREN コマンドで、印刷するデータより上のパラメータとしてレベル 1 メンバーを指定します。

- 特定のメンバーが属する次元のすべてのレベル 0 のデータをリストするには、DIMBOTTOM コマンドで、印刷するデータを含んでいる次元の任意のメンバーを指定します。

たとえば次のスクリプトでは、CHILDREN コマンドを使用して Qtr1 (レベル 1 のメンバー)の子を選択し、DIMBOTTOM コマンドを使用して、製品次元のレベル 0 のデータをすべて選択しています。

```
   <ROW (Year, Measures, Product, Market, Scenario)
{ROWREPEAT}
{DECIMAL 2}
<CHILDREN Qtr1
Sales
<DIMBOTTOM Product
East
Budget
   !
```

次のレポートが生成されます:

```
   Jan   Sales  100-10   East    Budget    1,600.00
Jan   Sales  100-20   East    Budget     400.00
Jan   Sales  100-30   East    Budget     200.00
Jan   Sales  200-10   East    Budget     300.00
Jan   Sales  200-20   East    Budget     200.00
Jan   Sales  200-30   East    Budget    #MISSING
Jan   Sales  200-40   East    Budget     700.00
Jan   Sales  300-10   East    Budget    #MISSING
Jan   Sales  300-20   East    Budget     400.00
Jan   Sales  300-30   East    Budget     300.00
Jan   Sales  400-10   East    Budget     300.00
Jan   Sales  400-20   East    Budget     400.00
Jan   Sales  400-30   East    Budget     200.00
Feb   Sales  100-10   East    Budget    1,400.00
Feb   Sales  100-20   East    Budget     300.00
Feb   Sales  100-30   East    Budget     300.00
Feb   Sales  200-10   East    Budget     400.00
Feb   Sales  200-20   East    Budget     200.00
Feb   Sales  200-30   East    Budget    #MISSING
Feb   Sales  200-40   East    Budget     700.00
Feb   Sales  300-10   East    Budget    #MISSING
Feb   Sales  300-20   East    Budget     400.00
Feb   Sales  300-30   East    Budget     300.00
Feb   Sales  400-10   East    Budget     300.00
Feb   Sales  400-20   East    Budget     300.00
Feb   Sales  400-30   East    Budget     300.00
Mar   Sales  100-10   East    Budget    1,600.00
Mar   Sales  100-20   East    Budget     300.00
Mar   Sales  100-30   East    Budget     400.00
Mar   Sales  200-10   East    Budget     400.00
Mar   Sales  200-20   East    Budget     200.00
Mar   Sales  200-30   East    Budget    #MISSING
Mar   Sales  200-40   East    Budget     600.00
Mar   Sales  300-10   East    Budget    #MISSING
```

```
Mar    Sales   300-20   East    Budget     400.00
Mar    Sales   300-30   East    Budget     300.00
Mar    Sales   400-10   East    Budget     300.00
Mar    Sales   400-20   East    Budget     400.00
Mar    Sales   400-30   East    Budget     300.00
```

データ・エクスポート用のフォーマットのその他の例は、『Oracle Essbase テクニカル・リファレンス』の「レポート・ライター・コマンド」の項の「レポート・スクリプトの例のサンプル 12」ページを参照してください。

37 リレーショナル・データの取得

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- Oracle Essbase Integration Services System Administrator's Guide
- Oracle Essbase Administration Services Online Help

リレーショナル・データベースと Essbase の統合

リレーショナル・データベースには数テラバイトのデータを保管することができるため、無限ともいえる拡張容易性があります。一方、多次元データベースは、一般的にリレーショナル・データベースよりも小規模ですが、高度な分析能力を提供します。リレーショナル・データベースと Essbase データベースを統合することによって、多次元データベースの概念的な能力で拡張容易性の高いリレーショナル・データベースを活用できます。

デフォルトでは、Integration Server で Essbase アウトラインを作成するときに、メタアウトラインで指定されたすべてのメンバー・レベルを多次元データベースにロードします。ただし、指定のメンバー・レベルに構築(ハイブリッド分析)する、または次元レベルにのみ構築(拡張リレーショナル・アクセス)するように Integration Server を設定できます。指定のレベルまでの構築の方が、多次元データベースと Essbase アウトラインは小さくなります。

636 ページの「ハイブリッド分析」および642 ページの「拡張リレーショナル・アクセス」を参照してください。

Extended Online Analytic Processing (XOLAP)を使用して、ソース・リレーショナル・データベースと Essbase データベースを統合できます。これは、ビジネス・インテリジェンスでの OLAP の役割の変形です。具体的には、XOLAP は、アウトライ

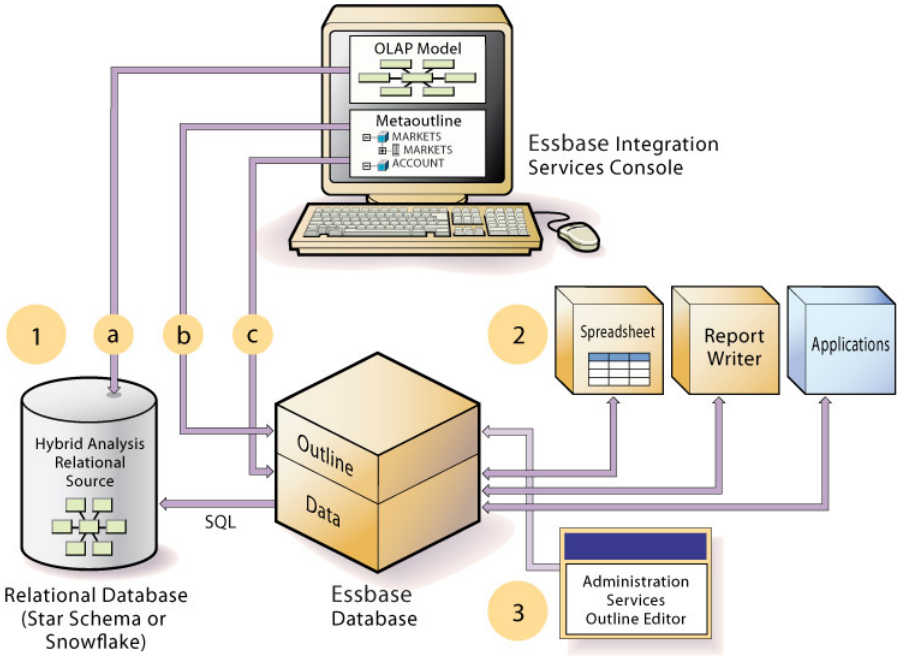
ン・メタデータのみを保管して、クエリー時にリレーショナル・データベースからデータを取得する Essbase 多次元データベースです。したがって、XOLAP は、Essbase データベースと統合して、多次元データベースの高度な分析機能とともにリレーショナル・データベースの拡張容易性を活用します。OLAP と XOLAP のどちらがビジネス環境に適しているかを判断する必要があります。

643 ページの「XOLAP の概要」を参照してください。

ハイブリッド分析

ハイブリッド分析によって、下位レベルのメンバーおよびそれらのデータを Essbase データベースにロードして保管する必要がなくなります。この機能によって、アウトライン・サイズに関する制限をほとんど受けることなく Essbase を使用できるようになります。また、Essbase データベースとリレーショナル・データベースとの間のデータ転送も迅速に行えるようになります。ハイブリッド分析によって、リレーショナル・データベースと Essbase 多次元データベースが統合されるので、アプリケーションとレポート・ツールは、両方のデータベースから直接データを取得できます。図 141 に、ハイブリッド分析のアーキテクチャを示します:

図 141　ハイブリッド分析のアーキテクチャ



ハイブリッド分析リレーショナル・ソース

ハイブリッド分析を設定する最初の手順は、リレーショナル・データベースをハイブリッド分析リレーショナル・ソースとして定義することです(図 141 の 1)。

ハイブリッド分析リレーショナル・ソースは Integration Services コンソールで定義します。Integration Services コンソールでは、最初に OLAP モデルのリレーショナル・データ・ソースを指定します。OLAP モデルは、リレーショナル・データベースのテーブルと列から作成するスキーマです。モデルを構築するため、Integration

Services では、リレーショナル・データベースのスター・スキーマにアクセスします(図 141 の a)。

このモデルを使用して、階層、およびメンバーをハイブリッド分析に対して使用可能にするタグ・レベルを定義します。メタアウトライン(Essbase アウトラインを作成するための構造とルールを含んだテンプレート)を、目的のハイブリッド分析レベルまで作成します。ハイブリッド分析の使用可能情報は、OLAP メタデータ・カタログに保管されます。このカタログには、ハイブリッド分析リレーショナル・ソースのデータの性質、ソース、場所およびタイプが記述されます。

次に、メンバー・ロードを実行して、次元とメンバーを Essbase アウトラインに追加します(図 141 の b)。メンバー・ロードが完了した後で、データ・ロードを実行して Essbase データベースにデータを入れます(図 141 の c)。この時点で、ハイブリッド分析のアーキテクチャは次の状態になります:

- 下位レベル・メンバーおよびその関連付けられたデータは、リレーショナル・データベース内に残されます。
- リレーショナル・データベースのデータは、ハイブリッド分析によって定義された Essbase アウトラインにマップされます。
- アウトラインは、Essbase データベースに保管されます。

  メタデータは、データベース内の値を記述したデータです。ハイブリッド分析データを定義するメタデータは、Essbase アウトラインおよび Essbase アウトラインの基となる Integration Services メタアウトラインに存在します。Essbase アウトラインに関連付けられている OLAP モデルまたはメタアウトラインのハイブリッド分析データが変更された場合は、アウトラインを更新して、Essbase で報告されるデータの正確さを確保する必要があります。641 ページの「ハイブリッド分析でのデータの一貫性の管理」を参照してください。
- 上位レベルのメンバーおよびその関連付けられたデータは、Essbase データベースに保管されます。

データの取得

スプレッドシートやレポート・ライターのインタフェースなどのアプリケーションとレポート・ツールは、両方のデータベースから直接データを取得できます(図 141 の 2)。アウトラインに定義されている次元とメンバー構造を使用して、Essbase では、メンバーの場所を特定し、Essbase データベースまたはハイブリッド分析リレーショナル・ソースからデータを取得します。データがハイブリッド分析リレーショナル・ソースに存在する場合は、Essbase では SQL コマンドを使用して取得します。638 ページの「ハイブリッド分析データの取得」を参照してください。

アウトラインを変更するには、Administration Services のアウトライン・エディタを使用して、必要に応じてハイブリッド分析の次元を使用可能または使用不可にできます(図 141 の 3)。640 ページの「ハイブリッド分析でのアウトライン・エディタの使用」および『Oracle Essbase Integration Services システム管理者ガイド』を参照してください。

ハイブリッド分析データの取得

Integration Services コンソールでのハイブリッド分析リレーショナル・ソースの定義方法は、Oracle Essbase Integration Services System Administrator's Guide を参照してください。

ハイブリッド分析では、アプリケーションとレポート・ツールは、次のツールを使用してリレーショナル・データベースと Essbase データベースから直接データを取得できます:

- Smart View
- レポート・ライター
- Oracle Hyperion Web Analysis
- サードパーティ製アプリケーション

注： Essbase データベースとリレーショナル・データベースは、同じ ODBC データ・ソースに登録する必要があります。Integration Services では、両方のデータベースに同じソース名を使用する必要があります。

計算を実行するとき、またはレポートを生成するときに、データはハイブリッド分析リレーショナル・ソースと Essbase データベースからアクセスされるので、ハイブリッド分析によってデータ取得にかかる時間が増える可能性があります。ただし、ピボット、ドリルスルー、その他のメタデータベースのメソッドを含む、Essbase のデータ取得操作のほとんどの機能は、ハイブリッド分析で使用できます。

ハイブリッド分析の取得環境の定義

表 119 にリストされた構成設定を使用して、Essbase サーバーまたは特定のアプリケーションとデータベースに対してハイブリッド分析を使用可能にし、定義します:

表 119　ハイブリッド分析の取得構成設定

設定	説明
HAENABLE	ハイブリッド分析リレーショナル・ソースからのメンバーの取得を可能にします
HAMAXNUMCONNECTION	Essbase がリレーショナル・データベースに対してアクティブな状態を保持できる、データベースごとの接続の最大数を設定します
HASOURCEDSNOS390	OS/390 システム上の DB2 データ・ソースへのアクセスを可能にします
HAMAXNUMSQLQUERY	1 つの Essbase クエリー・セッションにつきリレーショナル・データベースのファクト・テーブルに対して発行できる、SQL クエリーの最大数を設定します
HAMAXQUERYTIME	ハイブリッド分析リレーショナル・ソースからの SQL クエリーのクエリーごとの最大時間制限を設定します
HAMAXQUERYROWS	Essbase クエリーのために発行された SQL クエリーごとに戻される行の最大数を設定します

設定	説明
HARETRIEVENUMROW	一度に処理するために SQL クエリーから戻される最大行数が設定されます
HARAGGEDHIERARCHY	ハイブリッド分析対応のアウトライン用の次元の作成に使用する次元テーブルの列で、NULL 値を使用可能にします
HAMEMORYCACHESIZE	ハイブリッド分析リレーショナル・ソースからのクエリー済のメンバーをキャッシュするために予約するメモリー量を設定します

『Oracle Essbase テクニカル・リファレンス』を参照してください。

Smart View を使用したハイブリッド分析データの取得

Smart View では、ハイブリッド分析は自動的に使用可能になります。

ハイブリッド分析では、Smart View で「親ドリルアップ」・オプションをサポートします。リレーショナル・メンバーに対するドリルアップでは、そのリレーショナル・メンバーの直接の親ではなく、常に Essbase アウトラインのリーフレベル・メンバーが得られます。

Oracle Hyperion Smart View for Office User's Guide を参照してください。

レポート・ライターを使用したハイブリッド分析データの取得

レポート・ライターでは、コマンドを使用してハイブリッド分析を使用可能または使用不可にします:

- <HYBRIDANALYSISON を使用すると、ハイブリッド分析を使用可能な次元のメンバーをレポート・スクリプトで取得できるようになります。
- <HYBRIDANALYSISOFF を使用すると、ハイブリッド分析を使用可能な次元のメンバーをレポート・スクリプトで取得できなくなります。

<ASYM コマンドと<SYM コマンドは、ハイブリッド分析でサポートされません。この 2 つのコマンドがレポートにある場合は、エラーが発生する可能性があります。<SPARSE コマンドは、ハイブリッド分析リレーショナル・ソースからデータを取得するレポートで無視され、エラーは発生しません。

次は、IDESCENDANTS コマンドを使用してハイブリッド分析データを戻すレポート・ライター・スクリプトのサンプルです:

```
   <PAGE (Accounts, Scenario, Market)
Sales
Actual
<Column (Time)
<CHILDREN Time
<Row (Product)
<IDESCENDANTS 100-10
!
```

Web Analysis を使用したハイブリッド分析データの取得

Web Analysis を使用してハイブリッド分析データを取得する手順は、ハイブリッド分析に定義されていないデータを取得する手順と同じです(Web Analysis のドキュメンテーションを参照してください)。

Web Analysis を使用してハイブリッド分析データを取得するときのパフォーマンスを最適にするため、次のガイドラインを考慮してください:

- Web Analysis でクエリーを作成するときに、ハイブリッド分析を使用可能な次元を行に沿って配置します。
- ソート・オプションを使用しないようにします。
- 階層をドリル・ダウンするときに子演算子を使用します。子孫演算子は使用しないでください。
- 制限または最上位/最下位を取得する形での追加処理は、クエリーの時間が遅くなる可能性があります。

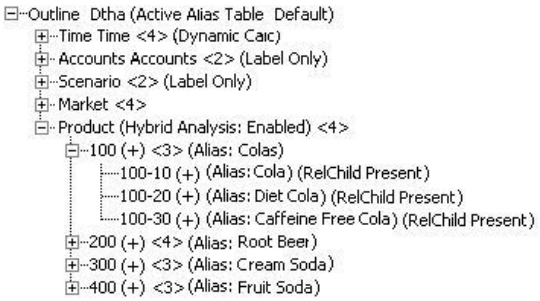
ハイブリッド分析でのアウトライン・エディタの使用

アウトライン・エディタでは、「ハイブリッド分析」ボタンを切り替えることによって、Integration Services コンソールでハイブリッド分析に対して定義されている各次元のハイブリッド分析を使用可能または使用不可にできます。ハイブリッド分析に定義されていないアウトラインを開いた場合、「ハイブリッド分析」ボタンはツールバーに表示されません。

注： 次元のハイブリッド分析が使用不可の場合、ユーザーは、その次元に関連するハイブリッド分析データを表示およびドリル・スルーすることができません。ただし、次元のメンバーは、アウトライン・エディタに表示されたままです。

図 142 は、アウトライン・エディタに表示されたハイブリッド分析に定義されているアウトラインの例です。ハイブリッド分析が使用可能な次元は、ハイブリッド分析を使用できない次元と区別するために特定されています。

図 142　アウトライン・エディタでのハイブリッド分析の例



ハイブリッド分析でのデータの一貫性の管理

ハイブリッド分析リレーショナル・ソースを作成すると、データとメタデータは、リレーショナル・データベースと Essbase データベースで次のように保管および管理されます:

- 下位レベルのメンバーおよびその関連データは、リレーショナル・データベースに残されたままです。
- リレーショナル・データベースのデータは、ハイブリッド分析に定義されている Essbase アウトラインにマップされます。
- アウトラインは、Essbase データベースに保管されます。
- 上位レベルのメンバーおよびその関連付けられたデータは、Essbase データベースに保管されます。

データとメタデータは別の場所にあるので、両者の情報が同期していない状態になることがあります。

Essbase は、ハイブリッド分析リレーショナル・ソースにアクセスするときに Integration Services の OLAP メタデータ・カタログに依存します。Essbase データベースの起動時に、Essbase サーバーでは、Essbase アウトラインの次元とメンバーの数を関連するメタアウトラインと照合します。

Integration Services セッション中に、関連する OLAP モデルまたはメタアウトラインに対して行われた変更は、Essbase データベースが再び起動するまで Essbase サーバーによって検出されません。検出されていない変更によって、Essbase データベースとハイブリッド分析リレーショナル・ソースの間にデータの不整合が生じる可能性があります。

ハイブリッド分析リレーショナル・ソースで変更が行われたり、OLAP モデルまたはメタアウトラインにメンバーが追加または削除された場合、そのような変更によって、Essbase アウトラインはそれが基にしているメタアウトラインと同期しなくなる可能性があります。このような変更と次元の階層構造への変更の影響は、アウトライン構築とデータ・ロード・プロセスが Integration Services コンソールによって完了するまで、Essbase データベースに反映されません。

Administration Services の「データベースの再構築」ダイアログ・ボックスには、再構築がハイブリッド分析リレーショナル・ソースを含んでいるアウトラインに影響する場合はいつでも警告するように設定できるチェック・ボックスがあります。このような問題は、たとえば、リレーショナル上の子を持つメンバーが移動または削除された場合に発生します。

警告は、アプリケーション・ログにリストされます。警告で、データの一貫性を脅かすものを反映するのかどうかを指定する必要があります。アプリケーション・ログを表示する方法は、817 ページの「Essbase サーバー・ログおよびアプリケーション・ログの表示」を参照してください。

Essbase の管理者は、Essbase 多次元データベース、リレーショナル・データベースおよび Integration Services OLAP モデルとメタアウトラインが常に同期を維持するようにする必要があります。Administration Services と Integration Services コンソールには、管理者が一貫性検査を実行して、該当する更新を実行するためのコマンドが用意されています。

第 51 章「データの整合性の確保」および第 56 章「データベースの再構築の最適化」を参照してください。

ハイブリッド分析でのセキュリティの管理

Essbase の管理者は、個々の Essbase ユーザー・レベルでのハイブリッド分析リレーショナル・ソースへのアクセスを設定します。ハイブリッド分析のアクセスは、Essbase 全体のセキュリティに影響する要因と同じ要因で制御されます:

- 個々のユーザーおよびユーザー・グループに対する権限およびアクセス・レベル
- サーバー、アプリケーションまたはデータベースに対する権限およびアクセス・レベル
- 特定のデータベース・メンバーに対する特定のデータベース・アクセス・レベル

セキュリティ・フィルタにおいて、Essbase で表示できる、レベル 0 メンバーのリレーショナル上の子が制限されている場合、Essbase でアクセスできないレベル 0 メンバーのリレーショナル上の子は表示できません。

次のような市場次元のアウトラインがあると仮定します。このアウトラインでは、San Francisco と San Jose は California のリレーショナル上の子であり、Miami と Orlando は Florida のリレーショナル上の子です:

```
Market
  West
    California
      San Francisco
      San Jose
  East
    Florida
      Miami
      Orlando
```

この例で、フィルタによってレベル 0 メンバー California とその子孫のみが表示できるようになっている場合、California とそのリレーショナル上の子である San Francisco および San Jose は表示できます。しかし、レベル 0 メンバー Florida の子は表示できません。

次の章を参照してください:

- 第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」
- 第 41 章「セキュリティ・フィルタを使用したデータベース・セルへのアクセスの制御」
- 第 42 章「ネイティブ・セキュリティ・モードでのセキュリティの例」

拡張リレーショナル・アクセス

Integration Services は、拡張リレーショナル・アクセスを使用して、Essbase ユーザーがリレーショナル・データベースまたはデータ・ウェアハウスのデータに直接アクセスできるようにします。Integration Services コンソールでは、拡張リレーショナル・ストレージは、メタアウトライン・レベルで使用可能です。リレーショ

ナル・ストレージ・オプションを選択すると、非会計次元のすべてのメンバーに対してリレーショナル・ストレージが使用可能になります。選択した非会計次元に対してリレーショナル・ストレージを使用可能にすることもできます。

メタアウトラインで拡張リレーショナル・アクセスが使用可能な場合、ユーザーは、リレーショナルに保管されたメンバーに直接クエリーを実行できます。データベースへのクエリーは、多次元式(MDX)を使用して実行します。この式は、SQL文に変換されます。次元メンバーはリレーショナル・データ・ソースから直接アクセスされます。

注： 拡張リレーショナル・アクセスの使用可能化に関する詳細は、『Oracle Essbase Integration Services オンライン・ヘルプ』および Oracle Essbase Integration Services System Administrator's Guide を参照してください。

XOLAP の概要

サブトピック

- XOLAP のワークフロー
- XOLAP の使用に関するガイドライン
- XOLAP の最適化

Extended Online Analytic Processing (XOLAP)は、ビジネス・インテリジェンスでの OLAP の役割の変形です。具体的には、XOLAP は、アウトライン・メタデータのみを保管して、クエリー時にリレーショナル・データベースからデータを取得する Essbase 多次元データベースです。したがって、XOLAP は、ソース・リレーショナル・データベースと Essbase データベースを統合して、多次元データベースの高度な分析機能とともにリレーショナル・データベースの拡張容易性を活用します。OLAP と XOLAP のどちらがビジネス環境に適しているかを判断する必要があります。

OLAP と XOLAP では、メタデータ・アウトラインおよび基になるデータを保管する場所が異なります:

- OLAP では、メタデータは Essbase データベースに配置され、基になるデータも Essbase データベースに配置されます。
- XOLAP では、メタデータは Essbase データベースに配置されますが、基になるデータはソース・リレーショナル・データベースに配置されます。

メタデータとデータの場所の違いは OLAP と XOLAP の機能に影響するので、XOLAP のメリットを理解する上でこの違いは手がかりとなります。

OLAP は、従来のリレーショナル・データ・ストレージとデータ分析に適しています。XOLAP は、(Essbase と Integration Services のユーザーによく知られている)ハイブリッド分析や拡張リレーショナル・アクセスなどの混合(ハイブリッド)環境でサポートされる操作に適しています。ハイブリッド分析と拡張リレーショナル・アクセスの基本概念の多くは、Essbase Studio の XOLAP キューブの機能に組み込まれています。

XOLAP の使用に関するガイドラインと制限、および XOLAP 用のモデルの指定方法は、Oracle Essbase Studio User's Guide を参照してください。

XOLAP のワークフロー

XOLAP 環境でのデータ取得ワークフローは、XOLAP 以外の環境でのワークフローによく似ています:

1. モデルは、Essbase Studio で XOLAP 対応に指定されます。
2. キューブは、Essbase Studio に配置されますが、この時点ではデータはロードされていません。
3. Essbase データベースは、Smart View、または Essbase データベースにアクセス可能なその他のレポート作成ツールを使用してクエリーされます。
4. Essbase によって、ソース・リレーショナル・データベースからデータを取得するために必要な SQL が自動的に生成されます。

XOLAP の使用に関するガイドライン

サブトピック

- XOLAP の制限
- XOLAP でサポートされない使用方法

XOLAP には、いくつかの制限があります。また、XOLAP でサポートされない使用方法もあります。

XOLAP の制限

XOLAP には次の制限があります:

- XOLAP キューブの編集は許可されていません。アウトラインを変更する場合は、Essbase Studio で新しいアウトラインを作成する必要があります。アウトラインの作成後に、次元テーブルの構造とコンテンツの変更は XOLAP 操作によって自動的に組み込まれません。
- 導出テキスト・メジャーが Essbase モデルを構築するためにキューブ・スキーマで使用されている場合、そのモデルに XOLAP は使用できません。
- XOLAP は、集約ストレージでのみ使用できます。データベースは自動的に重複メンバー対応になります。
- XOLAP では、カタログに対応するスキーママッピングのない次元をサポートしますが、そのような次元では、1 つのメンバーしか保管済メンバーにすることができません。

XOLAP でサポートされない使用方法

XOLAP では、次の使用をサポートしません

- フラット・ファイル
- 不規則階層
- 代替階層
- 再帰階層
- カレンダ階層
- フィルタ
- 型付きメジャー
- リーフ・レベルのユーザー定義メンバー
- 複数のリレーショナル・データ・ソース

XOLAP の最適化

Essbase の次の構成設定を使用して、XOLAP のパフォーマンスを最適化します:

- SSOPTIMIZEDGRIDPROCESSING: 作成する対称クエリーを少なくするために入力グリッドを対称グリッドに分ける最適化グリッド処理を、スプレッドシート操作で使用可能にするかどうかを指定します。
- SSBULKGRIDPROCESSING: 非対称グリッド API クエリーを XOLAP 用に最適化します。
- XOLAPENABLEHEURISTICS: アプリケーションの SQL を最適化する範囲を指定します。

第 VI 部

セキュリティ・システムの設計および管理

セキュリティ・システムの設計および管理の内容：

- EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ
- Essbase ネイティブ・セキュリティ・モードでのユーザーの管理およびセキュリティ
- Essbase セキュリティ・ファイル(essbase.sec)の管理
- セキュリティ・フィルタを使用したデータベース・セルへのアクセスの制御
- ネイティブ・セキュリティ・モードでのセキュリティの例

38 EPM Systemセキュリティ・モードでのユーザーの管理およびセキュリティ

この章の内容

関連項目:

- 第 39 章「Essbase ネイティブ・セキュリティ・モードでのユーザーの管理およびセキュリティ」
- 付録 G「異なるセキュリティ・モデルにおけるセキュリティ関連操作の相違」
- Oracle Enterprise Performance Management System Security Configuration Guide
- Oracle Enterprise Performance Management System User Security Administration Guide

EPM System セキュリティ・モードでの Essbase の使用

Essbase のユーザー管理とセキュリティは、Oracle Hyperion Shared Services を使用してユーザー管理、ユーザー・プロビジョニングおよび外部認証の定義を行う EPM System 経由で提供されます。プロビジョニングとは、役割とアクセス権を Essbase アプリケーションのユーザーに割り当てるプロセスのことです。

EPM System セキュリティを実装している製品には、Shared Services クライアントとサーバー・ソフトウェアを実行している Shared Services サーバーへのアクセス、および Shared Services 専用のデータベースへのアクセスが必要です。

デフォルトの配置オプションを使用した場合、Essbase は EPM System セキュリティ・モードで配置されます。

Essbase を EPM System セキュリティ・モードで配置しない場合、Essbase では、独自のネイティブ・セキュリティ・モードを使用して、Essbase アプリケーション、データベースおよびアーティファクトのセキュリティが管理されます。ネイティブ・セキュリティ・モードでの Essbase の動作は、ネイティブ・セキュリティ・モードで外部認証を使用する場合を除いて変わりません。672 ページの「Essbase を引き続きネイティブ・セキュリティ・モードで使用する」を参照してください。

Essbase ネイティブ・セキュリティ・モードの Essbase 配置に EPM System セキュリティを使用するには、Essbase サーバー・アプリケーションと既存の Essbase ユーザーおよびグループをすべて Shared Services に移行する必要があります。661 ページの「Essbase の EPM System セキュリティへの移行」を参照してください。

EPM System セキュリティにおけるユーザーとグループのセキュリティ

Essbase を EPM System セキュリティ・モードで実行中に、Essbase はユーザーおよびグループの詳細(ユーザーおよびグループ情報と Essbase アプリケーションへのプロビジョニングを含む)を、Shared Services から取得します。Essbase はすべてのユーザーおよびグループを Essbase セキュリティ・ファイル(essbase.sec)には格納しないため、Essbase の管理者は Essbase と Shared Services 間でセキュリティを明示的に同期化する必要がありません。

ユーザーが Essbase にログオンする際、Essbase は Shared Services にそのユーザーの情報を問い合せます。ユーザーがセッションを開始する権限は、セッション中にユーザーの権限が Shared Services で変更されるかどうかに関係なく、セッション全体で保持されます。

ユーザーまたはグループは、次の状況の場合のみ、Essbase セキュリティ・ファイルに格納されます:

- ユーザーまたはグループが、Shared Services に正常に移行されなかった。
- ユーザーまたはグループが、データベース計算またはフィルタ・アクセスを割り当てられている。
- ユーザーまたはグループが、クエリー・ガバナー除外リストで指定されている。
- ユーザーがアプリケーションまたはデータベースの作成者である。
- ユーザーがデータベース関連アーティファクトをロックしている。
- PERSISTUSERATLOGIN 構成設定が TRUE に設定されている。

  ユーザーが Essbase にログオンする際、そのユーザーが essbase.sec ファイルにまだ存在しなければ、ユーザーをファイルに追加するかどうかが PERSISTUSERATLOGIN により指定されます。

ユーザーまたはグループを Essbase セキュリティ・ファイルから削除する(Shared Services からプロビジョニング解除せずに)には、**from security_file** 文法を使用した **drop user** または **drop group** MaxL ステートメントを使用します。計算およびフィルタの関連付けも削除されます。

表示操作については、Essbase は Shared Services に問い合せます。参照:

- 651 ページの「EPM System セキュリティの移行およびアップグレードに関する考慮事項」
- 652 ページの「EPM System セキュリティの移行およびアップグレード後のユーザー・セキュリティに関する考慮事項」

EPM System セキュリティの移行およびアップグレードに関する考慮事項

Essbase を Essbase ネイティブ・セキュリティから EPM System セキュリティに移行する際、正常に移行できないユーザーは Essbase セキュリティ・ファイルに保持されます。

Shared Services への移行中にユーザーのデータベース権限が変更された場合、情報は `AccessModifiedUsers_n.txt` というテキスト・ファイルに書き込まれます。ここで `n` は、Shared Services に登録されている Essbase のインスタンスに対するシーケンス ID を表します。(シーケンス ID は Essbase が外部化されるたびに増加します。)`ARBORPATH/bin` ディレクトリにあるこのファイルには、次の情報が含まれます:

- ユーザーの名前
- ユーザーがアクセス権を持っているアプリケーションおよびデータベース
- Shared Services に移行する前の、特定のデータベースに対するユーザーのアクセス・レベル
- Shared Services へ移行後のユーザーのアクセス・レベル。グループその他の方法を使用して取得したアクセス権も含まれる
- ユーザーのフィルタ割当て

Shared Services では、Essbase アプリケーションに複数のデータベースが含まれている場合、それらのデータベースのユーザー・セキュリティ・アクセス・レベルが同じである必要があります(ただしユーザーは、同じアプリケーション内のデータベースに対して異なる計算スクリプトおよびデータベース・フィルタを割り当てることができます)。Shared Services への移行時に、同じアプリケーション内の 2 つのデータベースに対してユーザーが所持するアクセス・レベルが異なっていると、それらの両データベースに対するユーザーのアクセス・レベルはより制限が厳しくなります。

Shared Services ユーザーを、ユーザーのパスワードが自動的に生成されるオプションとともに移行すると、このユーザーの名前およびパスワードは `ARBORPATH/bin` ディレクトリにある `MigratedUsersPassword.txt` というテキスト・ファイルに書き込まれます。管理者が特定のパスワードを移行ユーザーに対して指定した場合は、パスワードはファイルにリストされません。このファイルは、アップグレードまたは移行の実行時に作成されます。

EPM System セキュリティの移行およびアップグレード後のユーザー・セキュリティに関する考慮事項

ユーザーの Shared Services への移行、または Essbase のアップグレードの後に、次のセキュリティ上の問題に対処する必要があります:

- ユーザーが Essbase にログオンして Essbase エージェントにアクセスする際、グローバル Essbase サーバー・アプリケーションにおけるサーバー・アクセスの役割がユーザーに割り当てられていることが不要になります。Essbase アプリケーションのどの役割を持つユーザーでも(直接的または間接的のどちらでプロビジョニングされている場合でも)、Essbase にログオンして Essbase エージェントにアクセスできます。

  グローバル Essbase サーバー・アプリケーションにおける役割を持たないユーザーは Essbase エージェントにアクセスできないようにするには、ユーザーの役割を手動で変更する必要があります。ユーザーがグループから継承する役割を必ず評価してください。
- ユーザーが Essbase にログオンする際、Essbase は Shared Services にそのユーザーの役割を問い合せます。ユーザーが持つ役割の数が多いほど、ログイン・プロセスの完了までの時間は長くなります。ログイン・パフォーマンスを向上するため、ユーザーに不必要な役割を割り当てないようにします。

  Essbase 管理者が Essbase アプリケーションを作成するときに、Essbase は、不要なアプリケーション・マネージャの役割をこの管理者に割り当てません。ご使用の Essbase 配置でアプリケーション・マネージャの役割が Essbase 管理者に割り当てられている場合は、Essbase 管理者からアプリケーション・マネージャの役割を手動で削除する必要があります。

  また、グローバル Essbase サーバー・アプリケーションにおける管理者の役割を持つ Essbase 管理者は、不要なアプリケーションの役割を割り当てられている可能性があります。Essbase には、Essbase 管理者のみに対するアプリケーションの役割を削除できる移行ユーティリティが備わっています。

移行ユーティリティ(32 ビットの Windows のみで稼働)は、両方のセキュリティ上の問題に影響されるユーザーを特定します。さらに、指定されている場合は、アプリケーションの役割を持つ Essbase 管理者の問題を修正することもできます。

➤ `migration.bat` ファイルを実行するには:

1. `ESSBASEPATH/utilities` ディレクトリに移動します。
2. `migration.zip` ファイルを解凍します。これによって `MigrationTools` サブディレクトリが作成されます。
3. `MigrationTools` サブディレクトリで、`migration.properities` ファイルを次の情報で編集します:
   - AdminUserName - 移行ユーティリティを実行する Essbase 管理者の名前
   - AdminUserPassword - Essbase 管理者のパスワード
   - HSSServer - Shared Services サーバーのホスト名
   - HSSPort - Shared Services サーバーのポート番号

- EssbaseProjectName - 移行ユーティリティを実行する対象の Essbase サーバーのプロジェクト名
- SavedToFile - TRUE に設定すると、移行スクリプトの結果を次のテキスト・ファイルに書き込むことが指定されます。このファイルは `MigrationTools` サブディレクトリ内にあります:
  - `EntitiesWithCubeRolesOnly.txt` - アプリケーションの役割は持っているが、グローバル Essbase サーバー・アプリケーションにおける役割は持っていないユーザーおよびグループをリストします
  - `EssbaseAdminsWithCubeRoles.txt` - アプリケーションの役割を持つ Essbase 管理者をリストします

    FALSE(デフォルト)に設定すると、結果は画面に書き込まれます。

    これらのファイルのサンプル・バージョンは `MigrationTools` サブディレクトリ内にあります。
- FixAdminUser - TRUE に設定すると、Essbase 管理者からアプリケーションの役割が削除されます。FALSE(デフォルト)に設定すると、Essbase 管理者の名前と、その管理者に割り当てられているアプリケーションの役割が画面に書き込まれます。

  例:

```
        AdminUserName = admin
AdminUserPassword = password
HSSServer = pant5
HSSPort = 58080
EssbaseProjectName = Analytic Servers:ALNG3:1
SavedToFile true
FixAdminUser false
```

注： Essbase サーバーに対して最初にユーティリティを実行する際は、結果を表示して確認できるように、SavedToFile を TRUE に、FixAdminUser を FALSE に設定することをお薦めします。それから、FixAdminUser を TRUE に設定してユーティリティを再度実行すれば、Essbase 管理者からアプリケーションの役割を削除できます。

4 `migration.bat` ファイルを実行します。

5 手動による必要な変更を行います。

Shared Services に対する Essbase ユーザーの役割

役割とは、ユーザーが実行できるタスクを指定するもので、次の方法でグループ化できます:

- 製品固有の役割

Essbase の役割の例は、管理者とデータベース・マネージャです。すべての Essbase の役割は、Shared Services アプリケーションに固有です(役割によってユーザーに付与される権限は、すべてのアプリケーションではなく、役割が割り当てられる特定のアプリケーションにのみ適用されます)。

- Shared Services の役割

  Shared Services の役割の例としては、プロジェクト・マネージャやプロビジョニング・マネージャをあげることができます。Shared Services の役割の多くはグローバルです(役割がすべての Shared Services アプリケーションに適用されます)。ただし、プロビジョニング・マネージャの役割は例外であり、アプリケーションに固有です。

  Oracle Enterprise Performance Management System User Security Administration Guide を参照してください。

次の Essbase の役割は、Essbase でタスクを実行するための様々なレベルの権限を提供します。

Essbase サーバーでは、次の役割をユーザーにプロビジョニングできます:

- 管理者
- アプリケーションの作成/削除
- サーバー・アクセス

各アプリケーションでは、次の役割をユーザーにプロビジョニングできます:

- アプリケーション・マネージャ
- データベース・マネージャ
- 計算
- 書込み
- 読取り
- フィルタ
- アプリケーションの起動と停止

Shared Services Console では、Essbase に属する役割は、Essbase ノードにグループ化されます。Essbase アプリケーションに属する役割は、アプリケーション・ノードにグループ化されます。

注： Administration Services の管理者ユーザーの役割を Shared Services を経由してプロビジョニングするという概念はありません。Shared Services では、Administration Services の管理者に役割は割り当てられません。

表 120 に、Essbase に固有のユーザーの役割、およびアプリケーションに固有のプロビジョニング・マネージャという Shared Services の役割を示します。表には、各ユーザーが実行できるタスクも記載します。

表 120　Essbase と Shared Services のユーザーの役割とタスク

ユーザーの役割	タスクの説明
プロジェクト・マネージャ	(Shared Services の役割)Shared Services 内でプロジェクトを作成および管理できます。
管理者(以前はスーパーバイザ)	サーバー、アプリケーションおよびデータベースへの完全な管理アクセス権を持っています。 **注：** プロビジョニング・マネージャの役割は、Shared Services アプリケーションに固有の役割で、Essbase 管理者(以前の呼び名はスーパーバイザ)の移行時に自動的に割り当てられます。ただし、Essbase 管理者を Shared Services Console で作成するときには、プロビジョニング・マネージャの役割を手動で割り当てる必要があります。プロビジョニング・マネージャの役割が割り当てられたユーザーは、アプリケーションに対する役割をユーザーとグループにプロビジョニングできます。
アプリケーションの作成/削除	アプリケーションおよびアプリケーション内のデータベースを作成および削除できます。このユーザーによって作成されたアプリケーションとデータベースに対するアプリケーション・マネージャ権限とデータベース・マネージャ権限を含みます。
サーバー・アクセス	最小限のアクセス権を与えられたアプリケーションまたはデータベースにアクセスできます。
アプリケーション・マネージャ(以前はアプリケーション・デザイナ)	特定のアプリケーション内でデータベースとアプリケーションの設定の作成、削除および変更を行うことができます。アプリケーション内のデータベースに対するデータベース・マネージャ権限を含みます。 アプリケーション・マネージャが削除できるのは、自分が作成したアプリケーションおよびデータベースのみです。 **注：** プロビジョニング・マネージャの役割は、Essbase アプリケーション・マネージャの移行時に自動的に割り当てられます。ただし、Essbase アプリケーション・マネージャを Shared Services Console で作成するときには、プロビジョニング・マネージャの役割を手動で割り当てる必要があります。
データベース・マネージャ(以前はデータベース・デザイナ)	割り当てられたアプリケーション内のデータベース(データベース・プロパティまたはキャッシュ設定を変更する場合など)、データベース・アーティファクト、ロックおよびセッションを管理できます。
計算	割り当てられた計算値とフィルタを使用して、指定範囲に基づくデータ値の計算、更新および読取りを行うことができます。
書込み	割り当てられたフィルタを使用して、指定範囲に基づくデータ値の更新および読取りを行うことができます。
読取り	データ値を読み取れます。
フィルタ	フィルタの制限に基づいて特定のデータとメタデータにアクセスできます。
アプリケーションの起動と停止	アプリケーションまたはデータベースを起動および停止できます。

Shared Services での Essbase プロジェクト、アプリケーションおよびデータベース

Shared Services アーティファクトには、プロジェクト、アプリケーション、ユーザーの役割、ユーザーおよびグループがあります。Shared Services でユーザーまたはグループにアクセス権を割り当てると、ユーザーまたはグループにアプリケー

ションに対する役割をプロビジョニングすることになります。Oracle Enterprise Performance Management System User Security Administration Guide を参照してください。

Shared Services と Essbase は両方とも、「アプリケーション」という用語を使用します。Essbase では、「アプリケーション」はデータベースのコンテナを指します。Shared Services では、「アプリケーション」はユーザーをプロビジョニングするアーティファクトを指します。この項では、特に Essbase アプリケーションであることが明記されていないかぎり、「アプリケーション」は Shared Services アプリケーションを指します。ほとんどの場合、Essbase アプリケーションは、Shared Services アプリケーションにマッピングされるので、区別は必要ありません。

Essbase の場合、移行は Essbase サーバーのレベルで実行されます。Essbase サーバーを Shared Services に移行すると、その Essbase サーバーに対して Shared Services プロジェクトが作成されます。プロジェクトの名前は `Essbase:machineName:EssbaseServer#`です。`machineName` は Essbase サーバー・コンピュータの名前で、`EssbaseServer#`はシーケンス番号です。最初に移行される Essbase サーバーのシーケンス番号は 1 です。同じコンピュータ上の複数の Essbase サーバーを移行する場合、シーケンス番号は 1 ずつ増加します。また、セキュリティ・ファイルを削除して Essbase サーバーを再び移行した場合は、正常な移行ごとに新しいシーケンス番号で新しいサーバー・プロジェクトが作成されます。Shared Services Console で不要なプロジェクトを削除できます。

Essbase では、プロジェクト内に次のアプリケーションを自動的に作成し、自動的にそのアプリケーションを Shared Services に登録します:

- `Essbase:machineName:EssbaseServer#`という名前のアプリケーション。これは、Shared Services プロジェクトと同じ名前です。このアプリケーションはグローバル Essbase サーバー・アプリケーションと呼ばれ、このアプリケーションを使用して Essbase サーバー・レベルでセキュリティを指定できます。移行後、Shared Services のグローバル・アプリケーションの名前とアプリケーション・プロジェクトの名前は、**rename global registration name** 文法で MaxL の **alter system** ステートメントを使用して変更できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。
- Essbase サーバー上の各 Essbase アプリケーションの Shared Services アプリケーション。Shared Services では、Essbase アプリケーションに複数のデータベースが含まれている場合、それらのデータベースのユーザー・セキュリティ・アクセス・レベルが同じである必要があります。(ただし、ユーザーは、同じアプリケーション内のデータベースに割り当てられた異なる計算スクリプトとデータベース・フィルタを持つことができます。660 ページの「Shared Services でのデータベース計算およびフィルタ・アクセスの割当て」を参照してください)。

  次の図では、`Essbase:JWARD:1` の第:1 インスタンスは、JWARD に移行された第 1Essbase サーバーに対する Shared Services プロジェクトです。
  `Essbase:JWARD:1` の第 2 インスタンスはグローバル Essbase サーバー・アプリケーションです。

```
    Projects
Essbase:JWARD:1
```

```
ASOsamp
DMDemo
Demo
Essbase Servers:JWARD:1
Sampeast
Sample
Sample_U
Samppart
```

Shared Services に移行した後で Essbase でアプリケーションとデータベースを作成すると、対応する Shared Services アプリケーションが Essbase サーバー・プロジェクト内に作成され、そのアプリケーションは自動的に Shared Services に登録されます。

Shared Services での Essbase ユーザーとグループ

Shared Services に移行すると、外部の認証ディレクトリにまだ存在しないすべてのネイティブな Essbase ユーザーとグループが、ネイティブな Shared Services ユーザー・ディレクトリ内のネイティブな Shared Services ユーザーとグループに変換され、同等の役割が与えられます。外部で認証されたユーザーは、Shared Services に登録されますが、元の認証ディレクトリに保管されたままです。663 ページの「ユーザーとグループの移行」を参照してください。

Shared Services に移行後、ユーザーとグループは、Shared Services Console または外部ユーザー・ディレクトリ経由で作成および管理する必要があります。Oracle Enterprise Performance Management System User Security Administration Guide を参照してください。

サポートされている認証プロバイダから外部認証ディレクトリにユーザーとグループを保管するとき、1 ユーザーと 1 グループを同じ名前にすることはできますが、同一プロバイダ上で、2 ユーザーまたは 2 グループを同じ名前にすることはできません。

Shared Services では、集約グループをサポートします。集約グループでは、親グループに 1 つ以上のサブグループが入っています。サブグループは、その親グループの役割を継承します。たとえば、親グループに Essbase 管理者の役割をプロビジョニングした場合、サブグループ(およびそのグループのユーザー)は、Essbase 管理者の役割を継承します。

注： Essbase では、アプリケーションまたはデータベースをコピーするときに、ターゲット Essbase サーバーが EPM System のセキュリティ・モードの場合、ユーザーとグループのセキュリティはアプリケーションと一緒にコピーされません。Shared Services Console のプロビジョニング・コピー機能を使用して、アプリケーションのセキュリティをコピーします。

Shared Services でのユーザーへのアクセス権の割当て

Shared Services Console には、EPM System 製品のユーザー管理タスクを実行できる集中管理 UI があります。Shared Services Console から Essbase 画面を起動して、セキュリティ・アクセスをデータベース・フィルタと計算スクリプトに割り当てることができます。EPM System のセキュリティ・モードでは、Shared Services Console、MaxL または API を使用してセキュリティを管理します(MaxL または API を使用してセキュリティを管理するときにはいくつかの制限があります。『Oracle Essbase テクニカル・リファレンス』および『Oracle Essbase API リファレンス』を参照してください)。管理サービス・コンソールでは、セキュリティ情報を表示できるのみです。

ユーザーとグループへのアクセス権の割当て、各アプリケーションのユーザー、グループおよびプロビジョニングされた役割に関するレポートの表示については、Oracle Enterprise Performance Management System User Security Administration Guide を参照してください。

Shared Services Console の起動とログイン

Essbase ユーザーを Shared Services Console で管理するには、次の Shared Services の役割がプロビジョニングされたユーザーとして、コンソールにログオンする必要があります:

- 該当する Essbase サーバーまたはアプリケーションのプロビジョニング・マネージャの役割
- 該当する認証ディレクトリのディレクトリ・マネージャの役割

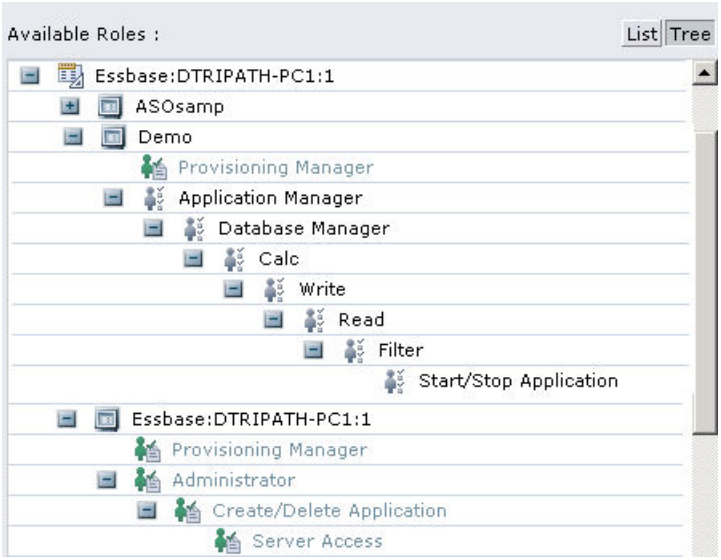
Shared Services Console を起動する場合、適切なユーザー権限でログオンする必要があります。たとえば、Essbase 管理者がユーザーを作成および削除できるように、その管理者にディレクトリ・マネージャの役割をプロビジョニングするには、Shared Services 管理者としてログオンする必要があります。

Shared Services でのサーバー・アクセスの割当て

EPM System セキュリティ・モードで、Essbase サーバー・レベルのセキュリティを指定するには(たとえば、Essbase サーバー上のすべての Essbase アプリケーションに対するプロビジョニング・マネージャの役割をユーザーにプロビジョニングする)、グローバル Essbase サーバー・アプリケーション、つまり、Essbase サーバーを表す Shared Services アプリケーションの該当する役割をユーザーにプロビジョニングします。655 ページの「Shared Services での Essbase プロジェクト、アプリケーションおよびデータベース」を参照してください。

図 143 は、DTRIPATH-PC1 という Essbase サーバーと Demo アプリケーションで使用可能な役割を示しています。

図 143　Shared Services Console のプロビジョニング・パネル



注： ユーザーに Essbase 管理者の役割をプロビジョニングするときには、手動でそのユーザーに Essbase サーバーおよびサーバー上の各 Essbase アプリケーションに対するプロビジョニング・マネージャの役割もプロビジョニングする必要があります(Essbase 管理者を移行した場合には、プロビジョニング・マネージャの役割は自動的に割り当てられます)。

Shared Services でのアプリケーション・アクセスの割当て

EPM System のセキュリティ・モードで、Essbase アプリケーション・レベルのセキュリティを指定するには(たとえば、Sample アプリケーションに対するデータベース・マネージャの役割をユーザーにプロビジョニングする)、アプリケーションの該当する役割をユーザーにプロビジョニングします。

注： ユーザーにアプリケーション・マネージャの役割をプロビジョニングするときには、手動でそのユーザーに該当するアプリケーションに対するプロビジョニング・マネージャの役割もプロビジョニングする必要があります。(Essbase アプリケーション・マネージャを移行する場合、プロビジョニング・マネージャの役割は自動的に割り当てられます)。

少なくともアプリケーション内のすべてのデータベースへの書込みアクセス権をすべてのユーザーに割り当てる場合などに、アプリケーションの最小権限を設定できます。デフォルトの設定は「なし」です。この場合、最小権限は設定されないので、すべてのユーザーは自分の役割に基づいてアプリケーションにアクセスできます。

➤ アプリケーションの最小権限を設定する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アプリケーションの最小権限の設定」を参照してください。

Shared Services でのデータベース計算およびフィルタ・アクセスの割当て

Shared Services Console で Essbase アプリケーションについてユーザーをプロビジョニングした後で、特定の Essbase アプリケーションとデータベースに対する詳細なアクセス権をユーザーとグループに割り当てることができます。たとえば、アプリケーションへのユーザー・アクセスを割り当て、そのアプリケーションに対する役割をユーザーに割り当てた後で、Essbase フィルタを割り当てるか、特定の計算スクリプトへのユーザー・アクセスを割り当てることができます。

フィルタと計算スクリプトは Essbase で作成する必要があります。第 41 章「セキュリティ・フィルタを使用したデータベース・セルへのアクセスの制御」を参照してください。

Shared Services Console から Essbase アプリケーションを選択すると、そのアプリケーションにプロビジョニングされているユーザーとグループが表示されます。この画面で、追加の権限を割り当てるユーザーおよびグループを選択できます。次に、ユーザーまたはグループがアクセスする必要があるデータベースを選択し、選択したユーザーとグループにフィルタと計算スクリプトへのアクセス権を割り当てます。Shared Services Console のオンライン・ドキュメントを参照してください。

➤ Shared Services でアクセス権を指定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**データベース計算およびフィルタ・アクセスの割当て**	Oracle Essbase Administration Services Online Help
MaxL	**grant**	Oracle Essbase Technical Reference

データベース計算とフィルタ・アクセスを割り当てるときには、Shared Services Console にログインしたユーザーとして自動的に Administration Services と Essbase とログオンします。このユーザーは、Essbase 管理者、アプリケーション・マネージャまたはデータベース・マネージャである必要があります。また、このユーザーには該当するアプリケーションに対するプロビジョニング・マネージャの役割が必要です。

データベース計算とフィルタ・アクセスを Essbase 管理者またはアプリケーション・マネージャに割り当てることはできません。

Shared Services でのアプリケーション・アクセス・タイプの割当て

Essbase と Planning には、Essbase ユーザーと Planning ユーザーに対するアプリケーション・アクセス・タイプという概念があります。たとえば、Essbase ユーザーをいずれかの Essbase 管理ツールを使用して作成すると、そのユーザーには自動的にアプリケーション・アクセス・タイプ「Essbase」が割り当てられます。Planning ユーザーを Planningインタフェースを使用して作成すると、そのユーザーには自動的にアプリケーション・アクセス・タイプ「Planning」が割り当てられます。ユーザーのアプリケーション・アクセス・タイプによって、そのユーザーが Essbase アプリケーションのみまたは Planning アプリケーションのみにアクセスできるのか、あるいは両方にアクセスできるのかが指定されます。

Shared Services Console で、グローバル Essbase サーバー・アプリケーションを選択し、そのアプリケーションにプロビジョニングされているユーザーとグループのリストを表示することができます。Oracle Enterprise Performance Management System User Security Administration Guide を参照してください。

データベース計算とフィルタ・アクセスを割り当てるときには、Shared Services Console にログインしたユーザーとして自動的に Administration Services と Essbase とログオンします。このユーザーは、有効な Essbase 管理者である必要があります。また、このユーザーには該当するアプリケーションに対するプロビジョニング・マネージャの役割が必要です。

➤ 新しくプロビジョニングしたユーザーのアプリケーション・アクセス・タイプ情報を反映するために Essbase をリフレッシュするには、「リフレッシュ」をクリックします。

Essbase の EPM System セキュリティへの移行

Essbase と Administration Services が Essbase ネイティブ・セキュリティ・モードである場合は、EPM System セキュリティ・モードに移行できます。Essbase の場合、移行は Essbase サーバーのレベルで実行されます。一度 EPM System セキュリティ・モードに変換すると、再び Essbase ネイティブ・セキュリティ・モードに戻すことはできません。

Essbase 管理サーバーは、Essbase サーバーが Essbase ネイティブ・セキュリティ・モードで動作している状態で、EPM System セキュリティ・モードで動作できます。ただし、管理サービス・コンソールから管理しているいずれかの Essbase サーバーが EPM System セキュリティ・モードで動作している場合は、Essbase Administration Server も EPM System セキュリティ・モードで動作する必要があります。

➤ Essbase サーバー、Essbase 管理サーバーおよびユーザーとグループを EPM System セキュリティに移行するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Shared Services への Essbase サーバーの変換とユーザーの移行	Oracle Essbase Administration Services Online Help
MaxL	**alter system**	『Oracle Essbase テクニカル・リファレンス』

移行を実行するには、Essbase 管理者である必要があります。

Essbase 管理サーバーの場合、インストールの後に Oracle Hyperion Enterprise Performance Management System コンフィグレータを実行し、Shared Services サーバーとログインを指定すると、その時点で Essbase 管理サーバーが EPM System セキュリティ・モードに変換されます。Shared Services の構成情報は、Essbase 管理サーバーのプロパティ・ウィンドウ(「構成」タブ)で確認できます。このとき、Administration Services ユーザーを選択して Shared Services に移行できます。

➤ Administration Services ユーザーを移行する場合、または移行に失敗した Essbase ユーザーとグループを移行しなおす場合には、次のツールを使用します:

ツール	トピック	場所
Administration Services	Shared Services へのユーザーの移行	Oracle Essbase Administration Services Online Help
MaxL	**display user** **display group** **alter user** **alter group**	Oracle Essbase Technical Reference

Administration Services ユーザーを移行するには、Administration Services 管理者である必要があります。

注： ユーザー、グループおよびアプリケーションを Shared Services に移行する前に、NETDELAY と NETRETRYCOUNT の構成設定が、この移行を完了できるだけの十分な大きさに設定されていることを確認してください。NETDELAY は、セキュリティ・ファイルのサイズに応じて、少なくとも 3 時間(場合によってはそれ以上)に設定します。移行が完了した後、これらの設定を元の値に戻してください。これらの設定は、クライアントの `essbase.cfg` ファイルで指定します。このファイルは、移行を起動するクライアント・コンピュータの `ARBORPATH/bin` フォルダに配置されます。たとえば、管理サービス・コンソールを使用して移行を起動する場合、クライアントの `essbase.cfg` ファイルは、Essbase 管理サーバーがインストールされているコンピュータの `ARBORPATH/bin` フォルダに存在する必要があります。

移行の前後に、Essbase によってセキュリティ・ファイルのバックアップが自動的に作成されます(`essbase.bak_preUPM` と `essbase.bak_postUPM`)。これらのファイルを安全な場所に手動でバックアップすることをお薦めします。

Administration Services の Essbase サーバー・プロパティ・ウィンドウには、サーバーが EPM System セキュリティ・モードにあるかどうかに関する情報が表示されます。

アプリケーションとデータベースの移行

Shared Services に移行した後、移行した Essbase サーバーごとに 1 つのプロジェクトが作成されます。このプロジェクトには、移行したサーバー上の各 Essbase アプリケーションに対する Shared Services アプリケーションが含まれています。655 ページの「Shared Services での Essbase プロジェクト、アプリケーションおよびデータベース」を参照してください。

ユーザーとグループの移行

Shared Services に移行すると、外部の認証ディレクトリにまだ存在しないすべてのネイティブな Essbase ユーザーとグループが、ネイティブな Shared Services ユーザー・ディレクトリ内のネイティブな Shared Services ユーザーとグループに変換され、同等の役割が与えられます。たとえば、ネイティブな Essbase 管理者(以前のスーパーバイザ)は、Essbase 管理者とプロビジョニング・マネージャの役割が割り当てられた Shared Services ユーザーになります。また、データベース上の計算権限を持つネイティブな Essbase ユーザーは、そのデータベースを含むアプリケーションでの計算の役割が割り当てられた Shared Services ユーザーになります。移行中、管理者とアプリケーション・マネージャには、該当するアプリケーションに対するプロビジョニング・マネージャの役割が自動的に与えられます。

注： Essbase が EPM System セキュリティ・モードで実行された場合、Essbase のユーザーの作成/削除権限は廃止されます。Essbase ユーザーを作成または削除するには、Essbase 管理者である必要があります。さらに、Shared Services でユーザーを作成または削除するには、Shared Services 管理者である必要があります。

外部認証ユーザーはすべて Shared Services に登録されますが、元の認証ディレクトリに保管されたままになります。ユーザー・ディレクトリが実行されていない場合は、移行全体が失敗します。

使用不可になった Essbase ユーザーまたはグループは移行されません。

Essbase のユーザー名は、Shared Services のグループ名として存在できません。存在する場合、その Essbase ユーザーは移行されません。

Shared Services では、Essbase アプリケーションに複数のデータベースが含まれている場合、それらのデータベースのユーザー・セキュリティ・アクセス・レベルが同じである必要があります。移行中、ユーザーが同じアプリケーション内の 2 つのデータベースに対して異なるアクセス・レベルを持っている場合は、両方のデータベースに対するより制限的なアクセス・レベルがそのユーザーに与えられます。このような場合には、移行を実行した管理者に警告が送信され、その情報が `AccessModifiedUsers_n.txt` ファイルに記録されます。このファイルは、`ARBORPATH/bin` ディレクトリにあります。

ユーザーとグループの移行は次の順序で実行されます:

1. Shared Services に登録されているアプリケーション
2. グループ
3. ユーザー

移行が失敗した場合の移行のステータスは、その失敗が発生した場所によって異なります。たとえば、移行がステップ 1 で失敗した場合は、移行全体が失敗します。移行がステップ 2 で失敗した場合の結果は、その失敗の理由によって異なります。移行がステップ 3 で失敗し、1 人以上のユーザーが移行に失敗した場合は、アプリケーションやグループが移行されていない可能性があります。

移行に失敗したユーザーとグループは、`essbase.sec` ファイルにリストされます。このファイルは、`ARBORPATH/bin` ディレクトリにあります。

Administration Services のユーザーの外部化ウィザードを使用して Administration Services ユーザーを移行するか、以前に移行に失敗した Essbase ユーザーを移行しなおす場合は、ウィザードで指定したファイルと、Essbase サーバー・ログ (`EPM_ORACLE_HOME/logs/essbase/essbase.log`)に移行エラーが記録されます。

グループを移行できなかった場合は、このグループのユーザーもすべて移行されません。ユーザーを正常に移行するには、グループを修復して移行する必要があります。

グループが Essbase と Shared Services の両方に存在する場合は、次の条件を満たす必要があります:

- Essbase に 2 つのグループが存在する場合、Shared Services に、それらのグループが同じ階層内の異なるレベル(祖先と子の関係)として含まれていてはなりません(例 2 を参照)。含まれている場合は、移行プロセス全体が失敗します。
- Shared Services グループに、同じ名前の Essbase グループに存在しないユーザーが含まれていてはなりません。含まれている場合は、その Essbase グループと、そのグループ内のすべてのユーザーが移行されません。
- Essbase グループに、Shared Services に存在するユーザーが含まれていてはなりません。ただし、その Shared Services ユーザーが同じ名前の Shared Services グループに属している場合を除きます。含まれている場合は、その Essbase グループと、そのグループ内のすべてのユーザーが移行されません。

次の例では、グループの移行における考慮事項に着目します:

例 1: この例にあるグループは、Essbase から Shared Services に正常に移行されます。

Essbase にグループ 1 およびグループ 2 という名前のグループが存在します:

```
group 1, group 2
```

Shared Services にはこれと同一の 2 つのグループが存在し、それ以外にグループ 1 とグループ 2 を含むグループ 3 も存在します:

```
group 3
|
```

```
group 1, group 2
```

グループ 1 とグループ 2 が互いに Shared Services と同じレベルにあり、Essbase にグループ 3 が存在しないため、これらのグループは正常に移行されます。

注： グループ 3 に Essbase サーバー・インスタンスへの管理者(以前のスーパーバイザ)アクセス権があり、Essbase のグループ 1 とグループ 2 にユーザー・アクセス権がある場合、結果として得られる Shared Services のグループ 1 とグループ 2 には管理者アクセス権が与えられます。

例 2: Shared Services に、異なるレベルにあるグループ 1 とグループ 2 が含まれているため、この例にある移行は失敗します。

Essbase にグループ 1 およびグループ 2 という名前のグループが存在します:

```
group 1, group 2
```

Shared Services にはグループ 1 とグループ 2 が存在しますが、グループ 1 にグループ 2 が含まれています:

```
group 1
  |
group 2
```

例 3: Essbase にグループ 1、グループ 2 およびグループ 3 が含まれ、Shared Services にグループ 1 やグループ 2 とは異なるレベルにあるグループ 3 が含まれまているため、この例にある移行は失敗します。

Essbase に、グループ 1、グループ 2 およびグループ 3 という名前のグループが存在します:

```
group 1, group 2, group 3
```

Shared Services にはグループ 1 とグループ 2 が存在しますが、グループ 1 とグループ 2 を含むグループ 3 が存在します:

```
  group 3
    |
group 1, group 2
```

ユーザー ID およびグループ ID

Essbase を EPM System セキュリティ・モードで実行している場合、ユーザーまたはグループのプロバイダ・ディレクトリまたは一意の ID 属性を指定すると、ユーザー名とグループ名は一意である必要がなくなりました。

MaxL では、ユーザー名とグループ名は、`name@provider` または一意の ID 属性として指定できます。

プロバイダは、LDAP または Active Directory などのユーザー・ディレクトリの名前で、外部ユーザーまたはグループをホストします。一意の ID 属性、あるいは単に「ID」とは、各ユーザーおよびグループに割り当てられた一意の文字列です。ID を使用すると、Essbase はプロバイダ間で同じ名前のユーザーやグループを識別できます。

詳細は、『Oracle Essbase テクニカル・リファレンス』の MaxL の USER-NAME および GROUP-NAME に関する項を参照してください

注： ユーザー名またはグループ名に@文字が含まれている場合、プロバイダも指定する必要があります。プロバイダを指定しないと、Shared Services では、@文字はプロバイダ名を示す区切り記号であるとみなされます。たとえば、Native Directory プロバイダを使用しているユーザー admin@msad をログインさせるには、admin@msad@Native Directory と指定する必要があります。

EPM System セキュリティ・モードでのセキュリティ情報のリカバリ

Shared Services のセキュリティ情報と Essbase セキュリティ・ファイル内のセキュリティ情報の間に不一致が発生した場合は、不一致の種類によって、Shared Services の情報と Essbase セキュリティ・ファイルの情報のどちらが優先されるかが決定されます。

参照:

- 666 ページの「ユーザーおよびグループ情報」
- 667 ページの「アプリケーション情報」

ユーザーおよびグループ情報

ユーザーおよびグループ情報については、Shared Services が優先されます。ユーザーおよびグループ情報は、Shared Services および外部ユーザー・ディレクトリのバックアップとリカバリの手順を使用して復元できます。『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

注： ユーザーおよびグループ情報をリカバリする場合、フィルタ、計算スクリプトおよびアプリケーション・アクセス・タイプへの関連付けは、Shared Services のバックアップにこれらの情報が存在しなければすべて失われます。Essbase セキュリティ・バックアップ・ファイルにこれらの情報が存在しない場合は、ユーザーまたはグループへの関連付けのみでなく、フィルタおよび計算スクリプト自体も失われます。

アプリケーション情報

アプリケーションおよびデータベース情報については、Essbase が優先されます。アプリケーションが Shared Services から削除されても、Essbase では引き続きそのアプリケーションが使用可能です。そのアプリケーションを Shared Services に再登録する必要があります。アプリケーションが Essbase で削除された場合、そのアプリケーションは Shared Services から自動的に削除されます。

➤ アプリケーションを Shared Services に再登録するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Shared Services へのアプリケーションの再登録	Oracle Essbase Administration Services Online Help
MaxL	**alter application**	『Oracle Essbase テクニカル・リファレンス』

39 Essbaseネイティブ・セキュリティ・モードでのユーザーの管理およびセキュリティ

この章の内容

関連項目:

- 第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」
- 付録 G「異なるセキュリティ・モデルにおけるセキュリティ関連操作の相違」
- Oracle Enterprise Performance Management System User Security Administration Guide
- Oracle Enterprise Performance Management System User Security Administration Guide

Essbase ネイティブ・セキュリティ・モードについて

Essbase は、Essbase 内のアプリケーション、データベース、その他のアーティファクトへのアクセスを管理するためのシステムを提供します。Essbase ネイティブ・セキュリティ・システムを使用すると、ローカル・エリア・ネットワークで使用可能な保護およびセキュリティが提供されます。

Essbase ネイティブ・セキュリティは、環境のための最適なプランの開発を可能にするマルチレイヤー・アプローチにより、様々なデータベース・セキュリティの

ニーズに対応しています。ユーザーやグループに様々なレベルの権限を付与したり、システム、アプリケーションまたはデータベース・レベルで定義できます。セキュリティは、次の方法で適用できます:

- ユーザーおよびグループ。

  個々のユーザーおよびユーザーのグループに権限を付与します。レベルが高い場合、これらの権限は、アプリケーションやデータベースに対して定義された最低限の権限に優先します。通常ユーザーには、固有の権限はありません。ユーザーやグループに権限を付与するには、ユーザーやグループを編集するか、**grant** MaxL ステートメントを使用します。674 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループへの権限の付与」を参照してください。

  Essbase パスワードのかわりに、外部の認証リポジトリのパラメータを使用してログオンするユーザーを作成できます。ユーザーが、LDAP などの外部の認証リポジトリを使用できるようにする場合は、EPM System セキュリティを実装し、そのセキュリティ・モードへの参照を使用して Essbase ユーザーを作成する必要があります。681 ページの「Essbase ネイティブ・セキュリティ・モードにおける外部認証での EPM System セキュリティの使用」を参照してください。

- アプリケーション設定およびデータベース設定。

  アプリケーションまたはデータベースのすべてのユーザーに共通の権限を設定するために、すべてのユーザーが各アプリケーションまたはデータベース・スコープで持つことのできる最小限の権限を設定できます。この最小限より低い権限を持つユーザーやグループは、アクセスできるようになります。これより高い権限が付与されているユーザーやグループは影響を受けません。また、アプリケーション設定を使用して、異なる種類のアクセスも一時的に使用不可にできます。681 ページの「ネイティブ・セキュリティ・モードでのアプリケーションおよびデータベースに関するグローバル・セキュリティの管理」を参照してください。

- サーバー全体に対する設定。

  Essbase サーバー全体のログイン制限を作成および管理します。Essbase サーバー全体、または特定のアプリケーションやデータベースのみで実行されている現在のセッションおよび要求を表示および終了します。687 ページの「Essbase ネイティブ・セキュリティ・モードでの Essbase サーバー上のユーザー・アクティビティの管理」を参照してください。

- データベース・フィルタ。

  ユーザーおよびグループが特定のメンバーに対して所有できるデータベース権限を定義します。この権限は、メンバーからメンバーの個々のデータ値(セル)に至るまでの詳細なレベルで定義可能です。第 41 章「セキュリティ・フィルタを使用したデータベース・セルへのアクセスの制御」を参照してください。

表 121 は、セキュリティ権限と、それらの権限で実行可能なタスクについて説明したものです。

表 121　Essbase の権限

権限	有効範囲	説明
アクセスなし(またはなし)	システム、アプリケーションまたはデータベース全体	アクセス権がグローバルに、またはフィルタによって付与されていないかぎり、どのユーザー、グループ、データ値に対しても固有のアクセス権はありません。アクセスなしは、通常ユーザーを作成したときのデフォルトです。アクセスなしの権限を持つユーザーは、自分のパスワードを変更できます。
読取り	データベース	データ値を読み取れます。
書込み	データベース	データ値の読取りと更新を行う権限。
メタ読取り	データベース	メタデータ(次元名およびメンバー名)の読取り、対応するメンバー指定データの更新を行う権限。
実行(または計算)	システム、アプリケーション、データベース、または単一の計算全体	割り当てられた計算を使用して、割り当てられたスコープに対するデータ値の計算、読取りおよび更新を行う権限。 管理者、そのアプリケーションのアプリケーション・マネージャおよびそのデータベースのデータベース・マネージャは、実行アクセス権を割り当てなくても、計算を実行できます。
データベース・マネージャ	データベース	アウトラインの変更、フィルタの作成および割当て、データベース設定の変更、データベース上のロックの削除、データベース上のセッションおよび要求の終了を行う権限。 あるデータベースでデータベース・マネージャ権限を持つユーザーが、別のデータベースでも、常に同じ権限を持つわけではありません。
アプリケーション・マネージャ	アプリケーション	割り当てられたアプリケーション内のデータベースの作成、削除および変更を行う権限。アプリケーション設定(最小限の権限を含む)の変更、アプリケーション上のロックの削除、アプリケーション上のセッションおよび要求の終了、アプリケーション内の任意のアーティファクトの変更を行う権限。同時にシステムレベルのアプリケーションの作成/削除の権限も付与されていないかぎり、アプリケーションを作成または削除できません。 あるアプリケーションでアプリケーション・マネージャ権限を持つユーザーが、別のアプリケーションでも、同じ権限を持つわけではありません。
フィルタ・アクセス	データベース	ユーザーまたはグループに割り当てられたフィルタの制限に従って、特定のデータおよびメタデータにアクセスする権限です。フィルタ定義では、データベースのサブセットについて、各サブセットに読取り、書込み、アクセスなし、メタ読取りが許可されるかどうかが指定されます。1 人のユーザーまたは 1 つのグループに付与できるフィルタは、1 つのデータベースにつき 1 つのみです。フィルタは他の権限と組み合せて使用できます。第 41 章「セキュリティ・フィルタを使用したデータベース・セルへのアクセスの制御」を参照してください。
アプリケーションの作成/削除	システム全体	アプリケーションやこれらのアプリケーション内のデータベースの作成および削除、作成されたアプリケーションの権限、ロックおよびリソースの制御を行う権限。このユーザーによって作成されたアプリケーションやデータベースのデザイナ権限を含みます。
ユーザー、グループの作成/削除	システム全体	自分と同じ、またはより低い権限を持つユーザーまたはグループの、作成、削除、編集または名前変更を行う権限。
管理者	システム全体	システム全体およびすべてのユーザーとグループに対するフル・アクセス権。

Essbase を引き続きネイティブ・セキュリティ・モードで使用する

Essbase サーバーで、ユーザーとグループを以前のリリースと同じように管理する場合は、Essbase ネイティブ認証を引き続き使用できます。Essbase ネイティブ・セキュリティ・モードでは、引き続き管理サービス・コンソールによってユーザーを管理します。以前と同様に、ネイティブ・ユーザーや外部ユーザーを引き続き作成できます。

外部認証を Essbase ネイティブ・セキュリティ・モードで使用する予定がある場合は、Shared Services で外部認証を構成する必要があります。681 ページの「Essbase ネイティブ・セキュリティ・モードにおける外部認証での EPM System セキュリティの使用」を参照してください。

次のオプションがあります:

- Essbase サーバーと Essbase 管理サーバーはどちらも、Essbase ネイティブ・セキュリティ・モードで動作できます。次の両方が当てはまる場合は、Shared Services のインストールや構成は必要ありません:
  - Essbase と Administration Services が、インストールされている唯一の EPM System 製品であること。
  - Essbase サーバーと Essbase 管理サーバーの両方を今後も Essbase ネイティブ・セキュリティ・モードで実行する予定であり、外部認証をネイティブ・セキュリティ・モードで使用する予定がないこと。
- Essbase 管理サーバーは、Essbase サーバーが Essbase ネイティブ・セキュリティ・モードで動作している状態で、EPM System セキュリティ・モードで動作できます。Provider Services サーバーにも同じルールが適用されます。

注: 管理サービス・コンソールから管理している Essbase サーバーを EPM System セキュリティ・モードで実行している場合、Essbase 管理サーバーも EPM System セキュリティ・モードで実行する必要があります。Essbase サーバーのユーザーの管理に、EPM System セキュリティ・モードと Essbase ネイティブ・セキュリティ・モードの組合せを使用することはできません。1 つのモードを選択して、そのモードで Essbase サーバーのすべての Essbase ユーザーを管理する必要があります。

Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループの作成

Essbase でユーザーまたはグループを作成する場合は、セキュリティ・プロファイルを定義します。このセキュリティ・プロファイルは、ユーザーおよびグループが互いに操作したり、アプリケーションやデータベースにアクセスしたりするときに持つ権限の範囲を定義するための場所です。

Administration Services を使用している場合は、Essbase 管理サーバーでもユーザーを作成する必要があります。107 ページの「Administration Services ユーザーについて」を参照してください。

Essbase ネイティブ・セキュリティ・モードでのユーザーの作成

ユーザーの作成とは、ユーザー名、パスワードおよび権限を定義することを意味します。また、ユーザーのグループ・メンバーシップを指定したり、ユーザーが次のログイン試行でパスワードを変更する必要があることや、ユーザー名を使用不可にしてユーザーがログオンできないように指定することもできます。

ユーザー名には、ESSLANG 変数で参照されるコード・ページ内で定義されている文字のみ使用できます。

➤ ユーザーを作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Essbase サーバーでのユーザーの作成	Oracle Essbase Administration Services Online Help
MaxL	**create user**	『Oracle Essbase テクニカル・リファレンス』

たとえば、admin という名前のユーザーを作成して、管理者権限を付与するには、次の MaxL ステートメントを実行します:

```
create user admin identified by 'password';

grant administrator to admin;
```

Essbase ネイティブ・セキュリティ・モードでのアプリケーション・アクセス・タイプの割当て

Essbase と Planning には、Essbase ユーザーと Planning ユーザーに対するアプリケーション・アクセス・タイプという概念があります。たとえば、Essbase ユーザーをいずれかの Essbase 管理ツールを使用して作成すると、そのユーザーには自動的にアプリケーション・アクセス・タイプ「Essbase」が割り当てられます。Planning ユーザーを Planning インタフェースを使用して作成すると、そのユーザーには自動的にアプリケーション・アクセス・タイプ「Planning」が割り当てられます。ユーザーのアプリケーション・アクセス・タイプによって、そのユーザーが Essbase アプリケーションのみまたは Planning アプリケーションのみにアクセスできるのか、あるいは両方にアクセスできるのかが指定されます。

➤ ユーザーのアプリケーション・アクセス・タイプを指定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ユーザーのアプリケーション・アクセス・タイプの設定	Oracle Essbase Administration Services Online Help

ツール	トピック	場所
MaxL	**create user**	『Oracle Essbase テクニカル・リファレンス』

Planning ユーザーのアプリケーション・アクセス・タイプの指定方法は、Oracle Hyperion Planning のドキュメンテーションを参照してください。

Essbase ネイティブ・セキュリティ・モードでのグループの作成

グループは、最小限のアクセス権を共有するユーザーで構成されています。ユーザーをグループに配置することにより、各ユーザーに同一の権限を繰り返し割り当てる時間が節約されます。

注： グループ内のユーザーに対して個別に権限が割り当てられている場合、グループに割り当てられた権限を超える権限をグループのメンバーが持つ場合があります。

グループを作成、編集またはコピーするプロセスは、グループのパスワードが存在しない点を除き、ユーザーに対するプロセスと同じです。グループ名と権限を、ユーザーの場合と同様に定義します。

ユーザーを作成する場合は、そのユーザーをグループに割り当てられます。同様に、グループを作成する場合も、ユーザーをそのグループに割り当てられます。グループのパスワードは存在しないため、ユーザーごとにパスワードを定義する必要があります。

➤ グループを作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Essbase サーバーでのグループの作成	Oracle Essbase Administration Services Online Help
MaxL	**create group**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループへの権限の付与

個々のユーザーやグループのセキュリティ権限を定義できます。グループは、最小限の権限を共有するユーザーで構成されています。ユーザーは、グループの権限を継承するだけでなく、グループの権限を超えるアクセス権を持てます。

ユーザーおよびグループに権限を付与する方法は、次のとおりです:

- ユーザーまたはグループの作成時、または編集によってユーザー・タイプまたはグループ・タイプを指定します。この方法では、すべてのアプリケーションおよびデータベースにまたがるシステムレベルの権限を指定します。

675 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザー・タイプおよびグループ・タイプの割当て」を参照してください。

- アプリケーションまたはデータベースにアクセスする権限を付与します。

  676 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーやグループへのアプリケーション・アクセス権およびデータベース・アクセス権の付与」を参照してください。

- ユーザーおよびグループへのデザイナ権限の付与。

  677 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーおよびグループへのデザイナ権限の付与」を参照してください。

Essbase ネイティブ・セキュリティ・モードでのユーザー・タイプおよびグループ・タイプの割当て

ユーザーおよびグループに権限を割り当てる方法の 1 つに、ユーザーおよびグループの作成または編集(権限の変更)時に、ユーザーおよびグループのタイプを定義する方法があります。

Administration Services では、システムレベルの権限を指定するために、ユーザーやグループを異なる方法で作成できます。これらの方法は、管理サービス・コンソールではユーザー・タイプとして表されます。MaxL にはユーザー・タイプは存在せず、かわりに、ユーザーを作成した後に権限を付与します。

Administration Services では、次のタイプを使用してユーザーを作成できます:

- 管理者

  管理者権限を持つユーザーまたはグループは、システム全体およびすべてのユーザーとグループに対するフル・アクセス権を持ちます。サーバーに Essbase をインストールするユーザーは、そのサーバーのシステム管理者に指定されます。Essbase では、各サーバー上の少なくとも 1 人のユーザーが管理者権限を持っている必要があります。そのため、サーバー上の最後の管理者の権限は削除またはダウングレードできません。

- User

  通常の権限を持つユーザーまたはグループには、どのユーザー、グループ、リソースに対しても固有のアクセス権はありません。このタイプのユーザーは、デフォルトのユーザーです。

- ユーザー、グループの作成/削除の権限を持つユーザー

  このタイプのユーザーまたはグループは、自分と同じか、またはより低い権限のユーザーおよびグループの作成、削除、編集および名前変更を行えます。

- アプリケーションの作成/削除の権限を持つユーザー

  このタイプのユーザーまたはグループは、アプリケーションの作成および削除と、作成したアプリケーションおよびデータベースに適用される権限およびリソースの管理を行えます。

  アプリケーションの作成/削除の権限を持つユーザーは、ユーザーを作成および削除できませんが、作成したアプリケーションに対するアプリケーション

レベルの権限を管理できます。681 ページの「ネイティブ・セキュリティ・モードでのアプリケーションおよびデータベースに関するグローバル・セキュリティの管理」を参照してください。

673 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの作成」および 674 ページの「Essbase ネイティブ・セキュリティ・モードでのグループの作成」を参照してください。

Essbase ネイティブ・セキュリティ・モードにおけるユーザーやグループへのアプリケーション・アクセス権およびデータベース・アクセス権の付与

ユーザーやグループに対して、どのユーザー・タイプにも暗黙的には含まれていないリソース固有の権限を付与する必要がある場合は、Administration Services でユーザーを作成または編集するときに、特定のアプリケーション権限またはデータベース権限を付与できます。MaxL を使用する場合は、**grant** ステートメントを使用してユーザーを作成した後に、これらの権限を付与します。

ユーザーおよびグループに対して、アプリケーションおよびデータベースに関する権限を付与したり変更したりする操作は、ユーザー編集の観点から行うことも、アプリケーションまたはデータベース・セキュリティの観点から行うこともできます。これらの結果は同じです。

注： データベースのデータにアクセスするための十分な権限を、ユーザーが持っていない場合には、データ値がクエリーに表示されないか、または #NOACCESS と表示されます。

すでに管理者であるユーザーまたはグループに権限を付与する必要はありません。これらのユーザーまたはグループは、Essbase サーバー上のすべてのリソースに対する完全な権限を持っています。特定のデータベースについて、ユーザーまたはグループに、次のいずれかの権限も付与できます:

表 122　データベース権限

データベース権限	説明
なし	データベース内のどのアーティファクトまたはデータ値へのアクセス権もないことを示します。
フィルタ・アクセス	データおよびメタデータへのアクセスが、ユーザーに割り当てられたフィルタで制限されていることを示します。(第 41 章「セキュリティ・フィルタを使用したデータベース・セルへのアクセスの制御」を参照してください。) 「フィルタ」チェック・ボックスによって、ユーザーまたはグループにフィルタ・アーティファクトを付与します。1 人のユーザーまたは 1 つのグループに付与できるフィルタは、1 つのデータベースにつき 1 つのみです。このオプションまたは「なし」を除く他の任意のオプションを選択すると、リスト・ボックスからフィルタ・アーティファクトを選択できるようになります。
読取り専用	読取り権限、つまり、すべてのデータ値を取得する権限を示します。また、レポート・スクリプトも実行できます。

データベース権限	説明
読取りと書込み	すべてのデータ値を取得および更新できることを示します(ただし、計算はできません)。ユーザーは Essbase アーティファクトを実行できますが、変更できません。
メタ読取り	対応するメンバー指定に対して、メタデータ(次元名およびメンバー名)を取得および更新できることを示します。
計算	デフォルト計算、またはユーザーが実行権限を付与された任意の計算を使用して、すべてのデータ値を取得、更新および計算できることを示します。
データベース・マネージャ	すべてのデータ値を取得、更新および計算できることを示します。さらに、すべてのデータベース関連ファイルを変更できます。

➤ ユーザーまたはグループのアプリケーション権限またはデータベース権限を付与または変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アプリケーションおよびデータベースに対するユーザー/グループ権限の管理	Oracle Essbase Administration Services Online Help
MaxL	権限を付与する場合: **grant** ユーザー・タイプまたはグループを変更する場合: **alter user**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードにおけるユーザーおよびグループへのデザイナ権限の付与

ユーザーやグループに、特定のアプリケーションまたはデータベースのためのアプリケーション・マネージャまたはデータベース・マネージャの権限を付与できます。これらの権限は、特定のアプリケーションまたはデータベースを担当する必要があるが、他の目的には通常ユーザーの権限のみが必要なユーザーに管理権限を割り当てる場合に有効です。

次のいずれかの条件を満たす場合は、他のユーザーにデータベース・アクセス権を付与する必要があります:

- ユーザーがデータベース・アクセスに必要な十分なユーザー権限を付与されていない場合。
- 対象のデータベースに設定された最小限の権限では、ユーザーに十分なアクセス権が付与されない場合。
- フィルタによって、ユーザーに十分なアクセス権が付与されていない場合。

ユーザーまたはグループにデザイナ権限を付与するために使用できる方法については、676 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーやグループへのアプリケーション・アクセス権およびデータベース・アクセス権の付与」を参照してください。

Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループの管理

複数のユーザーおよびグループのセキュリティを管理するため、ユーザーが持つ権限のレベルに応じて、次のユーザー管理タスクを使用できます。参照:

- 678 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループの表示」
- 678 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの編集」
- 679 ページの「Essbase ネイティブ・セキュリティ・モードでのグループの編集」
- 679 ページの「Essbase ネイティブ・セキュリティ・モードにおける既存のセキュリティ・プロファイルのコピー」
- 680 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループの削除」
- 680 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザー名およびグループ名の変更」

Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループの表示

➤ ユーザーおよびグループを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Essbase サーバー・ユーザーおよびグループの表示	Oracle Essbase Administration Services Online Help
MaxL	**display user** または **display group**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードでのユーザーの編集

ユーザーの編集とは、ユーザーを作成したときに確立されたセキュリティ・プロファイルを変更することを意味します。ユーザー・パスワードの変更方法は、689 ページの「Essbase ネイティブ・セキュリティ・モードでのパスワード変更の適用」を参照してください。

➤ パスワードまたはその他のユーザー・プロパティを変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Essbase サーバー・ユーザーのプロパティの編集	Oracle Essbase Administration Services Online Help
MaxL	**alter user**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードでのグループの編集

グループの編集とは、グループを作成したときに確立したセキュリティ・プロファイルを変更することを意味します。

➤ グループ・メンバーシップを表示または変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	グループ・プロパティの編集	Oracle Essbase Administration Services Online Help
MaxL	**display user in group** **alter group**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードにおける既存のセキィリティ・プロファイルのコピー

同じ権限を持つユーザーを別のユーザーとして作成するための簡単な方法に、既存のユーザーのセキュリティ・プロファイルのコピーがあります。この新しいユーザーには、元のユーザーと同じユーザー・タイプ、グループ・メンバーシップおよびアプリケーション/データベース・アクセス権が割り当てられます。

また、既存のグループのセキュリティ・プロファイルをコピーすることによって新しいグループも作成できます。この新しいグループには、元のグループと同じグループ・タイプ、ユーザー・メンバーシップおよびアプリケーション・アクセス権が割り当てられます。

権限に応じて、同じ Essbase サーバー上のユーザーやグループをコピーしたり、ある Essbase サーバーから別の Essbase サーバーにコピーしたりできます。また、ユーザーやグループを、アプリケーションとともにサーバー間で移行もできます。Oracle Essbase Administration Services Online Help の「ユーザーのコピー」を参照してください。

ユーザーまたはグループをコピーするには、既存のユーザーまたはグループのセキュリティ・プロファイルを複製し、それに新しい名前を付けます。これにより、同一にしようとしている権限を再割り当てる時間が節約されます。

注： コピーすると、作成者が持っていないセキュリティ権限はすべてコピーから削除されます。たとえば、ユーザーの作成/削除の権限を持つユーザーが、既存の管理者のプロファイルをコピーすることによって管理者を作成できません。

➤ 既存のユーザーまたはグループのセキュリティ・プロファイルをコピーすることによってユーザーまたはグループを作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Essbase サーバー・ユーザーのコピー グループのコピー	Oracle Essbase Administration Services Online Help
MaxL	**create user** **create group**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループの削除

➤ ユーザーおよびグループを削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Essbase サーバー・ユーザーの削除 グループの削除	Oracle Essbase Administration Services Online Help
MaxL	**drop user** **drop group**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードでのユーザー名およびグループ名の変更

➤ ユーザー名およびグループ名を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Essbase サーバー・ユーザーの名前変更 グループの名前変更	Oracle Essbase Administration Services Online Help
MaxL	**alter user** **alter group**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードにおける外部認証での EPM System セキュリティの使用

外部認証とは、Essbase で必要となるユーザー・ログイン情報が、Lightweight Directory Access Protocol (LDAP)ディレクトリなどの中央の認証ディレクトリで管理されていることを意味します。

認証ディレクトリとは、ログイン名、パスワード、その他の企業情報などの、ユーザー情報の集中管理されたストアのことです。このリポジトリは電話帳のような働きをします。認証ディレクトリには、ユーザー名やパスワードよりはるかに多い情報、たとえば、電子メール・アドレス、従業員 ID、役職、アクセス権、電話番号などが含まれている可能性があります。また、ユーザー以外のアーティファクト、たとえば、企業の場所やその他のエンティティに関する情報が含まれている可能性もあります。

Essbase ネイティブ・セキュリティ・モードで外部認証用に EPM System セキュリティを使用するには、次のように、Shared Services をインストールして構成する必要があります:

- Essbase を Shared Services に登録します。

  Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

- Essbase のユーザー・ディレクトリを構成します。

  Oracle Enterprise Performance Management System User Security Administration Guide を参照してください。

➤ Administration Services を使用してユーザーの外部認証を管理するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の外部認証の管理に関する項を参照してください。

ネイティブ・セキュリティ・モードでのアプリケーションおよびデータベースに関するグローバル・セキュリティの管理

ユーザーやグループへの権限の付与に加えて、アプリケーションおよびデータベース全体とそれに関連するファイルやリソースのセキュリティ設定を変更できます。アプリケーションおよびデータベースのセキュリティ設定を使用すると、アプリケーションやデータベースの接続の管理や、最も低い共通のセキュリティ・プロファイルの作成できます。

Essbase ネイティブ・セキュリティ・モードでのアプリケーション設定の定義

アプリケーション設定を変更することによって、アプリケーションに適用される権限やその他のセキュリティ設定を定義できます。アプリケーションに対して定義するこれらの設定は、ユーザー・レベルでさらに高い権限が付与されたユーザーを除き、すべてのユーザーに影響を与えます。

アプリケーション設定を変更できるのは、管理者権限(またはそのアプリケーションに関するアプリケーション・マネージャ権限)を持つユーザーのみです。

参照:

- 682 ページの「Essbase ネイティブ・セキュリティ・モードでの一般アプリケーション接続オプションの設定」
- 685 ページの「Essbase ネイティブ・セキュリティ・モードにおけるアプリケーションおよびデータベースの最小権限の設定」

Essbase ネイティブ・セキュリティ・モードでの一般アプリケーション接続オプションの設定

次のアプリケーション設定が可能です:

- アプリケーションの簡単な記述
- ロックのためのタイムアウト設定
- アプリケーションのロード、コマンド・プロセス、接続、更新およびセキュリティを制御するオプション
- アプリケーションの最小限の権限を定義する設定(685 ページの「Essbase ネイティブ・セキュリティ・モードにおけるアプリケーションおよびデータベースの最小権限の設定」を参照)

様々なレベルのアプリケーション・セキュリティに、次の設定が使用できます:

- アプリケーションの開始を許可。

  これが使用不可になっていると、すべてのユーザーがアプリケーションを直接開始したり、または結果としてアプリケーションを開始させるような操作(アプリケーション設定の変更やデータベースの作成など)を行えなくなります。デフォルトでは、アプリケーションは開始を妨げられていません。

- Essbase サーバーの開始時に開始。

  これが使用可能になっていると、アプリケーションは、Essbase サーバーが開始すると常に自動的に開始します。デフォルトでは、Essbase サーバーが開始してもアプリケーションは開始しません。

- コマンドを許可。

  このチェックが解除されていると、ユーザーは、データベース情報の表示やデータベース設定の変更などのデータに固有でない要求を含め、アプリケーション内のデータベースに要求を発行できなくなります。管理者は、メンテ

ナンス操作中のデータベースへの偶発的な変更を防ぐための安全機構として、この設定の影響を受けます。デフォルトでは、コマンドは使用可能です。

- 接続を許可。

  このチェックが解除されていると、そのアプリケーションのアプリケーション・マネージャより権限が低いユーザーは、開始対象のデータベースを必要とする、アプリケーション内のデータベースへの接続を実行できなくなります。デフォルトでは、データベースへの接続は許可されています。

- 更新を許可。

  このチェックが解除されていると、ディスク上のデータベース構造への変更、たとえば、データに影響を与える可能性のあるすべての操作ができなくなります。この制限には、アウトライン操作は含まれません。メタデータの更新をブロックするには、データベースを読取り専用モードに設定するか、「コマンドを許可」または「接続を許可」(あるいはその両方)のチェックを解除します。デフォルトでは、更新は使用可能です。

- セキュリティを使用可能にする。

  このチェックが解除されていると、Essbase はアプリケーション内のすべてのセキュリティ設定を無視し、すべてのユーザーをアプリケーション・マネージャとして処理します。デフォルトでは、セキュリティは使用可能です。

表 123 は、アプリケーション保護設定の導入が有効になるとき、その期間、および影響を与えるユーザーに関する説明です。

表 123　アプリケーション保護設定の適用範囲および持続期間

使用不可になるアプリケーション設定	使用不可になった設定の効果が現れるタイミング	使用不可になった設定の影響を受けるユーザー	使用不可になった設定の持続する期間
アプリケーションの開始をユーザーに許可	即時	管理者を含む全ユーザー。 現在ログインしているユーザーおよびその後ログインするユーザー。	管理者が起動設定を再度使用可能にするまで、アプリケーションを開始できません。
Essbase サーバーの開始時にアプリケーションを開始	即時	全ユーザー。	管理者がこの設定を使用可能にしないかぎり、アプリケーションは Essbase サーバーとともに開始しません。
コマンドを許可	即時	管理者を含む全ユーザー。 現在ログインしているユーザーおよびその後ログインするユーザー。	次のいずれかのアクションが行われるまで、コマンドを使用できません: ● コマンドを使用不可に設定した管理者がログアウトする。 ● アプリケーションが終了され再起動される。 ● 管理者がコマンドを再度使用可能に設定する。

使用不可になるアプリケーション設定	使用不可になった設定の効果が現れるタイミング	使用不可になった設定の影響を受けるユーザー	使用不可になった設定の持続する期間
接続を許可	即時。ただし、すでにデータベースをロードしているユーザーには影響しない。	アプリケーション・マネージャより権限の低いユーザー。 現在ログインしているユーザーおよびその後ログインするユーザー。 データベースにすでに接続しているユーザーは影響を受けません。	次のいずれかのアクションが行われるまで、接続を使用できません: ● アプリケーションが終了され再起動される。 ● 管理者が接続を再度使用可能にする。
更新を許可	即時	管理者を含む全ユーザー。 現在ログインしているユーザーおよびその後ログインするユーザー。	次のいずれかのアクションが行われるまで、更新を使用できません: ● 更新を不可にした管理者がログアウトする。 ● アプリケーションが終了され再起動される。 ● 管理者が更新を再度使用可能にする。
セキュリティを使用可能にする	即時	管理者を含む全ユーザー。 現在ログインしているユーザーおよびその後ログインするユーザー。	ユーザーがセキュリティを再度使用可能に設定するまで、セキュリティ設定が使用不可になります。

➤ アプリケーション設定を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アプリケーション・プロパティの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter application**	『Oracle Essbase テクニカル・リファレンス』

注： コマンドまたは更新を使用不可にする必要のあるメンテナンス操作を実行している場合は、これらの設定を使用不可にしたセッションと同一のセッション内でメンテナンス操作を実行してください。

MaxL スクリプトでコマンドまたは更新を使用不可にする場合は、スクリプトの最後がセッションの終わりになることに注意してください。現在の MaxL スクリプトからネストした MaxL または ESSCMD スクリプトを呼び出すと、そのセッションも終了します。

ESSCMD スクリプトでコマンドまたは更新を使用不可にする場合は、スクリプトの最後がセッションの終わりになりますが、現在の ESSCMD スクリプトからネストした ESSCMD スクリプトを呼び出しても、セッションは終了しません。

注意 いずれかの許可の設定を解除している場合は、クライアント・コンピュータの電源断または再起動を実行しないでください。常に、サーバーから正しくログオフしてください。不適切なシャットダウンによって、アプリケーションにアクセスできなくなり、アプリケーションの完全なシャットダウンと再起動が必要になる場合があります。

停電またはシステムの問題が発生し、Essbase サーバーが不適切な方法で Essbase クライアントから切断され、アプリケーションにアクセスできなくなった場合は、アプリケーションをシャットダウンして再起動する必要があります。763 ページの「アプリケーションの開始と停止」を参照してください。

Essbase ネイティブ・セキュリティ・モードにおけるアプリケーションおよびデータベースの最小権限の設定

アプリケーションまたはデータベース・レベルで、最小限のデータベース・アクセス権を指定できます。最小限のデータベース・アクセス権がアプリケーションに対して指定されている場合は、そのアプリケーション内のすべてのデータベースに適用されます。最小限の権限が、アプリケーションまたはデータベースに対して「なし」(または「アクセス権なし」)より高いレベルに設定されている場合は、すべてのユーザーがそれらのデータベースへのアクセスのためにその権限を継承します。

たとえば、あるアプリケーションに最小限のデータベース・アクセス・レベルとして読取り権限が割り当てられている場合は、個々のユーザーの権限に読取りアクセス権が含まれていなくても、すべてのユーザーがそのアプリケーション内の任意のデータベースを読み取れます。同様に、あるデータベースに最小限の権限として「なし」が設定されている場合は、十分な権限が(直接の付与によって、あるいはフィルタまたはグループ・メンバーシップにより暗黙的に)付与されているユーザーのみがそのデータベースにアクセスできます。

管理者、アプリケーション・マネージャまたはデータベース・マネージャの権限を持つユーザーは、自分が所有しているアプリケーションまたはデータベースに適用される最小権限の設定には影響されません。管理者はすべてのリソースに対するフル・アクセス権を、アプリケーション・マネージャとデータベース・マネージャは自分のアプリケーションまたはデータベースに対するフル・アクセス権を持っています。

最小限の権限より低い権限を持つユーザーまたはグループは、どのアプリケーションまたはデータベースに関しても、少なくともその最小限の権限は継承します。

アプリケーションの最小権限の設定への変更は、この設定より権限が低いデータベースにのみ影響を与えます。つまり、より低いレベルで定義された設定の方が、よりグローバルな設定より優先されます。

表 124 に表示されている権限は、アプリケーションおよびデータベースの最小限の設定として使用できます。アプリケーションのデータベースは、アプリケーション権限がデータベースの権限より高く設定されている場合は常に、そのアプリケーションの権限を継承します。

表 124　アプリケーションおよびデータベースで使用可能な最小権限の設定

権限	説明
なし	アプリケーションまたはデータベースに対して最小限の権限が定義されていないことを指定します。「なし」は、新しく作成されたアプリケーションおよびデータベースに対するデフォルトのグローバル権限です。
読取り	アプリケーションまたはデータベース内のすべてのアーティファクトまたはデータ値に対して読取り専用アクセスを指定します。ユーザーは、ファイルの表示、データ値の取得、レポート・スクリプトの実行を行えます。読取りアクセス権では、データ値の更新、計算、アウトラインの変更は許可されません。
書込み	アプリケーションの各データベース、または 1 つのデータベース内のすべてのデータ値に対して更新アクセス権を指定します。ユーザーは、Essbase ファイルの表示、データ値の取得および更新、レポート・スクリプトの実行を行えます。書込みアクセスでは、計算やアウトラインの変更は許可されません。
メタ読取り	指定されたメンバーに読取りアクセス権を付与しますが、祖先のデータ、兄弟のデータおよびメタデータは表示されません。
計算	アプリケーションの各データベース、または 1 つのデータベース内のすべてのデータ値に対して計算および更新アクセス権を指定します。ユーザーは、ファイルの表示、データ値に基づいた計算の取得、更新および実行、レポートや計算スクリプトの実行を行えます。計算アクセス権では、アウトラインの変更は許可されません。
デザイナ (アプリケーションまたはデータベース)	アプリケーションの各データベース、または 1 つのデータベース内のすべてのデータ値に対して計算および更新アクセス権を指定します。さらに、デザイナ権限では、ユーザーが、アウトラインやファイルの表示および変更、データ値に基づいた計算の取得、更新および実行、レポートや計算スクリプトの実行を行えます。

注： データベースについて読取り以上のアクセス権を持つユーザーであればそのデータベースを起動できますが、データベースを停止できるのは、管理者、そのアプリケーションのアプリケーション・マネージャ権限を持つユーザーおよびそのデータベースのデータベース・マネージャ権限を持つユーザーのみです。

➤ アプリケーションの最小限の権限を設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アプリケーションの最小権限の設定	Oracle Essbase Administration Services Online Help
MaxL	**alter application**	『Oracle Essbase テクニカル・リファレンス』

➤ データベースの最小限の権限を設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データベースの最小権限の設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードでの Essbase サーバー上のユーザー・アクティビティの管理

このトピックでは、Essbase サーバーに接続しているユーザーのアクティビティを管理する方法について説明します。参照:

- 687 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの切断および要求の終了」
- 688 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザー・ロックの管理」
- 688 ページの「Essbase ネイティブ・セキュリティ・モードでのパスワードとユーザー名の管理」
- 689 ページの「Essbase ネイティブ・セキュリティ・モードでのパスワード変更の適用」
- 689 ページの「Essbase ネイティブ・セキュリティ・モードにおける無効にしたユーザーの名前の表示とアクティブ化」

パーティション・データベースのセキュリティ管理の詳細は、第 15 章「パーティション・アプリケーションの設計」を参照してください。

Essbase ネイティブ・セキュリティ・モードでのユーザーの切断および要求の終了

セキュリティ・システムにより、メンテナンス作業を実行するために、ユーザーを Essbase サーバーから切断できます。

セッションを表示したり、セッションを切断したり、要求を終了したりするには、管理者権限または指定されたアプリケーションのアプリケーション・マネージャの権限が必要です。表示したり終了したりできるのは、自分の権限以下の権限を持つユーザーのセッションまたは要求のみです。

セッションとは、システム、アプリケーション、またはデータベース・スコープで Essbase サーバーに接続されているユーザーのログインからログアウトまでの時間を指します。ユーザーは、いつでも複数のセッションを開けます。たとえば、ユーザーは別のデータベースにログオンできます。適切な権限を持っている場合は、選択した任意の基準に基づいてセッションをログオフできます;たとえば、管理者は、すべてのデータベースまたは 1 つのデータベースからユーザーをログオフできます。

要求とは、ユーザーまたは別のプロセスによって Essbase サーバーに送信されたクエリーのことです。たとえば、データベースのデフォルト計算や、データベース・アウトラインの再構築などがあります。各セッションは、一度に 1 つの要求のみを処理できます。

注： 再構築プロセスを終了できません。終了を試みた場合、「command not accepted」エラーが戻され、再構築プロセスは終了されません。

- Administration Services を使用してセッションまたは要求を切断するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「ユーザー・セッションおよび要求の切断」を参照してください。

- MaxL を使用してセッションまたは要求を切断するには、**alter system kill request** または **alter system logout session** を使用します。『Oracle Essbase テクニカル・リファレンス』を参照してください。

- MaxL シェルまたは ESSCMD を使用して自分のアクティブ要求を切断するには、[Esc]キーを押します。

- カスタム API プログラムを使用して自分のアクティブ要求を切断するには、カスタム・コールバック関数を設定します。『Oracle Essbase API リファレンス』の ESS_INIT_T に関する項を参照してください。

Essbase ネイティブ・セキュリティ・モードでのユーザー・ロックの管理

Smart View のユーザーは、データをスプレッドシートからサーバーに対話的に送信できます。複数ユーザーの同時アクセスを可能にしながらデータ整合性を維持するために、Essbase では、ユーザーが更新の目的でデータをロックできます。データを更新しようとするユーザーは、他のユーザーが同じデータの変更を試みられないようにするために、まずレコードをロックする必要があります。

場合によっては、ロック解除操作の強制が必要になることがあります。たとえば、アクティブなロックを含むデータベースを計算しようとした場合、ロックが検出されると計算は待機しなければならなくなります。そのロックを解除することにより、計算を再開できるようになります。

ロックしているユーザーの表示と、そのロックの除去を実行できるのは、管理者のみです。

- ロックを表示または削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	「データのロックの表示」および「データのロック解除」	Oracle Essbase Administration Services Online Help
MaxL	**drop lock**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードでのパスワードとユーザー名の管理

ユーザーに許可されるログイン試行の回数、サーバーから使用不可にされるまでユーザー・アカウントが非アクティブ状態になる日数、およびユーザーが同じパスワードを使用できる日数に制限を加えられます。これらの設定にアクセスできるのは、システム管理者(管理者権限を持つユーザー)のみです。これらの制限は、

サーバー上のすべてのユーザーに適用され、「OK」をクリックすると有効になります。

注： 後から失敗したログイン試行の許可される回数を変更すると、Essbase はすべてのユーザーのカウントをリセットします。たとえば、15 であった設定を 20 に変更した場合、ユーザーには 20 回の新しい試行が許可されます。この設定を 2 に変更した場合、設定が 15 であったときにその回数を超えていたユーザーはロックアウトされません。設定を変更するたびに、カウントは 0 に戻ります。

➤ ユーザーに制限を加えるには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**パスワードの有効期限の管理** **ユーザーの自動切断**	Oracle Essbase Administration Services Online Help
MaxL	**alter system** **alter user**	『Oracle Essbase テクニカル・リファレンス』

Essbase ネイティブ・セキュリティ・モードでのパスワード変更の適用

ユーザーのパスワードを変更し、新パスワードを他の Essbase サーバーに適用できます。ソース・サーバーとターゲット・サーバーの両方に、ユーザーおよびグループの作成/削除の権限が必要です。パスワードを変更している対象のユーザーがターゲット・サーバー上に存在している必要があり、かつターゲット・サーバーが実行されている必要があります。

Administration Services を使用してユーザーの Essbase サーバー・パスワードを変更する場合や、ユーザーが Administration Services ユーザーでもある場合は、そのユーザーの Administration Services ユーザー・プロパティが自動的に更新されます。そのユーザーの Administration Services パスワードは影響されません。Oracle Essbase Administration Services Online Help の「Essbase サーバー・ユーザーのパスワードの変更」を参照してください。

➤ ユーザーの Essbase サーバー・パスワードを変更し、新パスワードを他の Essbase サーバーに適用するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「サーバー間でのパスワードの適用」を参照してください。

Essbase ネイティブ・セキュリティ・モードにおける無効にしたユーザーの名前の表示とアクティブ化

Essbase サーバーのレベルでユーザー名を使用不可にすることによって、ユーザーが Essbase サーバーにログインできないようにできます。指定された制限をユー

ザーが超えると、そのユーザー名は自動的に使用不可になります。あるいは、個々のユーザーのユーザー名を手動で使用不可にもできます。688 ページの「Essbase ネイティブ・セキュリティ・モードでのパスワードとユーザー名の管理」を参照してください。

管理サービス・コンソールでは、Essbase サーバーで使用不可になっているすべてのユーザー名を表示し、アクティブにすることができます。使用不可になっているユーザー名を表示したり、再びアクティブにできるのは、少なくともユーザーの作成/削除の権限を持つユーザーのみです。

➤ ユーザー名を手動で使用不可にするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ユーザー名の無効化	Oracle Essbase Administration Services Online Help
MaxL	**alter user**	『Oracle Essbase テクニカル・リファレンス』

➤ 現在使用不可になっているユーザー名を表示したり、アクティブにしたりするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	無効化されたユーザー名の表示またはアクティブ化	Oracle Essbase Administration Services Online Help
MaxL	**alter user**	『Oracle Essbase テクニカル・リファレンス』

40 Essbaseセキュリティ・ファイル(essbase.sec)の管理

この章の内容

Essbase セキュリティ・ファイルについて

Essbase が EPM System セキュリティ・モードにある場合は、一部のセキュリティ情報が Shared Services および外部ユーザー・ディレクトリによって保管され、一部のセキュリティ情報が Essbase セキュリティ・ファイル(essbase.sec)に保管されます。

Shared Services または外部ユーザー・ディレクトリによって、次の情報が保管されます:

- ユーザー
- グループ
- パスワード
- アプリケーションに関するユーザーとグループの役割の情報

essbase.sec には、次の情報が保管されます:

- 計算スクリプトのアクセス権
- フィルタ・アクセス
- アプリケーション・アクセス・タイプ
- アプリケーション・プロパティおよびデータベース・プロパティ、代替変数および(ブロック・ストレージ・データベースの場合の)DISKVOLUME の設定など

Essbase がネイティブ・セキュリティ・モードにある場合は、ユーザー、グループ、パスワード、権限、フィルタ、アプリケーション、データベースおよびそれ

らに対応するディレクトリに関するすべての情報が、essbase.sec に保管されます。

essbase.sec ファイルは、ARBORPATH/bin ディレクトリにあります。

essbase.sec の内容は暗号化されます。

Essbase セキュリティ・バックアップ・ファイルの管理

Essbase サーバーを起動するたびに、Essbase によって tempessbase.sec という名前の一時ファイルが作成されます。このファイルは、Essbase セキュリティ・ファイル(essbase.sec)を検証するのに使用されます。essbase.sec を使用して Essbase サーバーが正常に起動すると、tempessbase.sec は削除され、Essbase によって essbase_timestamp.bak という名前のセキュリティ・バックアップ・ファイルが作成されます。

Essbase が維持するセキュリティ・バックアップ・ファイルの数、セキュリティ・バックアップ・ファイルを作成する間隔、essbase.sec ファイルが無効の場合、起動時に最新で有効なセキュリティ・バックアップ・ファイルに切り替えるかどうかを管理できます。

- NUMBEROFSECFILEBACKUPS 構成設定: Essbase が作成および管理するセキュリティ・バックアップ・ファイルの最大数(2 から 10 まで)を指定します。制限を超えると、Essbase によって最も古いタイムスタンプのセキュリティ・バックアップ・ファイルが削除され、最新のバックアップ・ファイルが作成されます。

  デフォルトでは、Essbase は最低でも 2 バージョンの essbase_timestamp.bak を維持します。

- SECFILEBACKUPINTERVAL 構成設定: セキュリティ・バックアップ・ファイルの作成前に Essbase が待機する時間を指定します。

  デフォルトでは、Essbase は 300 秒(5 分)ごとに新しいセキュリティ・バックアップ・ファイルを作成します。

- ENABLESWITCHTOBACKUPFILE 構成設定: Essbase が起動時に有効なセキュリティ・バックアップ・ファイルを自動的にロードするかどうか(essbase.sec ファイルが無効な場合)を指定します。

  デフォルトでは、セキュリティ・ファイルが無効な場合、Essbase は起動時にセキュリティ・バックアップ・ファイルをロードしません。

  696 ページの「Essbase セキュリティ・ファイルの外部ディスク上の Essbase の状態への調整」も参照してください。

essbase_timestamp.bak の内容は暗号化されます。

➤ essbase_timestamp.bak ファイルを更新するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	セキュリティ・バックアップ・ファイルの更新	Oracle Essbase Administration Services Online Help
MaxL	**alter system sync security backup**	『Oracle Essbase テクニカル・リファレンス』
essbase.cfg	NUMBEROFSECFILEBACKUPS SECFILEBACKUPINTERVAL ENABLESWITCHTOBACKUPFILE	『Oracle Essbase テクニカル・リファレンス』

注： 移行の前後に、Essbase によってセキュリティ・ファイルのバックアップが自動的に作成されます(essbase.bak_preUPM と essbase.bak_postUPM)。661 ページの「Essbase の EPM System セキュリティへの移行」を参照してください。

Essbase セキュリティ・ファイルの復元

ファイルが欠落しているか、または無効(または破損している)であり、かつすべてのバックアップ・ファイルが無効な場合、Essbase セキュリティ・ファイル(essbase.sec)を復元する必要があります。

Essbase サーバーを開始しようとしてもパスワードのプロンプトが表示されない場合や、パスワードが拒否された場合は、バックアップ・ファイルが作成されません。essbase.sec を最新の有効なセキュリティ・バックアップ・ファイルから復元するには、essbase_timestamp.bak を essbase.sec にコピーします。これらのファイルはどちらも、ARBORPATH/bin ディレクトリにあります。

Essbase を EPM System セキュリティ・モードで使用している場合は、Shared Services および使用している任意の外部ユーザー・ディレクトリから最新のバックアップを復元することも必要です。

essbase.sec ファイルが欠落している場合(Essbase Server の bin ディレクトリが移動された後にパッチが適用された場合など)、Essbase では起動時に新しいセキュリティ・ファイルが作成されますが、Essbase は EPM System セキュリティ・モードのかわりに Essbase ネイティブ・セキュリティ・モードで起動します。次のメッセージに類似したメッセージが essbase.log ファイルに記録されます:

```
   [Mon Oct 08 16:10:30 2012]Local/ESSBASE0///2636/Error(1051430)
Failed to create Essbase project [EssbaseCluster-1] at Shared Services with Error
[Application group already exist.]

[Mon Oct 08 16:10:30 2012]Local/ESSBASE0///2636/Warning(1056822)
Failed to migrate Essbase to HSS security mode during startup. Check
SharedServices_Security_client.log. May need to remove essbase.sec and any of the
newly migrated Essbase applications from HSS, then rerun configuration tool.]
```

Essbase を EPM System セキュリティ・モードで起動するには、Shared Services から Essbase プロジェクト(EssbaseCluster-1 など)を削除した後、Essbase を再起動します。

essbase.sec ファイルが無効な場合は、次のメッセージに類似したメッセージが essbase.log ファイルに記録されます:

```
   [Mon Oct 08 15:35:19 2012]Local/ESSBASE0///2600/Warning(1056801)
Validation of security file [C:\Oracle\Middleware\user_projects
\epmsystem1\EssbaseServer\essbaseserver1\BIN\ESSBASE.SEC] failed

[Mon Oct 08 15:35:19 2012]Local/ESSBASE0///2600/Warning(1056801)
Validation of security file [C:\Oracle\Middleware\user_projects
\epmsystem1\EssbaseServer\essbaseserver1\bin\ESSBASE_1349731009.BAK] failed
```

essbase.cfg ファイルで ENABLESWITCHTOBACKUPFILE 構成設定が TRUE に設定されている場合、Essbase では起動時に有効なセキュリティ・バックアップ・ファイルが自動的にロードされます。次のメッセージに類似したメッセージが essbase.log ファイルに記録されます:

```
   [Mon Oct 08 15:35:19 2012]Local/ESSBASE0///2600/Warning(1056802)
ESSBASE will use the file [C:\Oracle\Middleware\user_projects
\epmsystem1\EssbaseServer\essbaseserver1\bin\ESSBASE_1349720690.BAK] as the security
file

[Mon Oct 08 15:35:19 2012]Local/ESSBASE0///2600/Info(1056797)
Incremental security backup started by SYSTEM. The file created is
[C:\Oracle\Middleware\user_projects\epmsystem1\EssbaseServer\essbaseserver1\bin
\ESSBASETS_1349735719.BAK]

[Mon Oct 08 15:35:20 2012]Local/ESSBASE0///2600/Info(1051134)
External Authentication Module: [Single Sign-On] enabled

[Mon Oct 08 15:35:20 2012]Local/ESSBASE0///2600/Info(1051051)
Essbase Server - started
```

すべてのセキュリティ・バックアップ・ファイルが無効な場合は、次のメッセージに類似したメッセージが essbase.log ファイルに記録されます:

```
   ...
rtype = 50630 rlen = 50116 rindex = 49602
Read Error:Part of security file is missing
Read Error:Invalid security file
rtype = 12849 rlen = 13363 rindex = 32513
Read Error:Part of security file is missing
Read Error:Invalid security file

Fatal Error: The security file is not valid and there are no valid back up files to
use. Restart Essbase after deleting the security file, a new security file will be
created.
```

ここでも、Shared Services から Essbase プロジェクト(EssbaseCluster-1 など)を削除した後、Essbase を再起動します。

Shared Services および Essbase のバックアップ手順については、『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』および該当する外部ユーザー・ディレクトリのドキュメンテーションを参照してください。

注意 Essbase セキュリティ・ファイルと Shared Services は同時にバックアップしてください。

Essbase セキュリティ・バックアップ・ファイル比較の頻度の変更

Essbase が最新の有効なセキュリティ・バックアップ・ファイル(`essbase_timestamp.bak`)を更新するのは、Essbase よって、セキュリティ・ファイル(`essbase.sec`)は最新のセキュリティ・バックアップ・ファイル(`essbase_timestamp.bak`)の作成後に変更されたと判別された場合です。デフォルトでは、Essbase は、Essbase サーバーの起動時のみではなく、指定された間隔でこの確認を実行します。

間隔値を変更する前に、次の点を考慮してください:

- Administration Services では、セキュリティ・バックアップ・ファイルのセキュリティ・ファイルとの確認頻度、およびユーザーの非アクティビティの確認頻度が、同じチェック・ボックスによって管理されています。
- デフォルト値は 5 分です。これは、セキュリティ・バックアップ・ファイルがセキュリティの変更を取り込めるだけの十分な頻度で確認されるようにするための推奨設定値です。また、5 分は、非アクティビティの確認にも推奨される値です。
- この値にゼロを設定すると、非アクティブの確認は使用不可になり、`essbase_timestamp.bak` ファイルは 5 分ごとに `essbase.sec` と比較されます(必要に応じて更新されます)。
- セキュリティ・ファイルを頻繁に更新する必要がない場合は、大きい値を入力します。パフォーマンスが問題にならない場合は、小さい値を入力します。

指定された間隔のみではなく、いつでもセキュリティ・バックアップ・ファイルを更新できます。

➤ セキュリティ・バックアップ・ファイルを手動で更新したり、バックアップ・ファイル比較の頻度を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	セキュリティ・バックアップ・ファイルの更新	Oracle Essbase Administration Services Online Help
MaxL	**alter system sync security_backup**	『Oracle Essbase テクニカル・リファレンス』

Essbase セキュリティ・ファイルの外部ディスク上の Essbase の状態への調整

Essbase の起動時に使用するのが最新で有効なセキュリティ・バックアップ・ファイル(essbase_timestamp.bak)であり、essbase.sec ではない場合、**alter system** MaxL ステートメントを使用して、外部ディスク上の Essbase アプリケーションおよびデータベースの状態に一致するよう、セキュリティ・ファイルを調整できます。

alter system reconcile 文法によって essbase.log にメッセージが記録されるのは、次の場合です:

- アプリケーションまたはデータベース・フォルダがディスク上にあるが、セキュリティ・ファイルにない場合
- アプリケーションまたはデータベースがセキュリティ・ファイルにあるが、ディスク上にない場合。このシナリオでは、**alter system reconcile force** 文法を使用して、セキュリティ・ファイルからアプリケーションまたはデータベースを削除します。

Essbase セキュリティ・ファイルの断片化の管理

Essbase のセキュリティ・エンティティ(フィルタ、ユーザー、グループ、アプリケーション、データベース、代替変数、ディスク・ボリューム、パスワード、その他の Essbase アーチファクト)を変更または削除すると、セキュリティ・ファイル(essbase.sec)の断片化が発生する場合があります。ファイルで発生する断片化が多すぎると、セキュリティ関連のパフォーマンスが低下することがあります。

Essbase では、エージェントが停止するたびに、セキュリティ・ファイルが自動的にコンパクト化されます。セキュリティ・ファイルの断片化ステータスを確認し、エージェントを停止することなくセキュリティ・ファイルをコンパクト化できます。

- 696 ページの「Essbase セキュリティ・ファイルの断片化ステータスの表示」
- 697 ページの「エージェント実行中の Essbase セキュリティ・ファイルのコンパクト化」

Essbase セキュリティ・ファイルの断片化ステータスの表示

セキュリティ・ファイルの断片化ステータスは、パーセントで表示されます。

➤ セキュリティ・ファイルの断片化ステータスを表示するには、**security file fragmentation_percent** 文法で MaxL の **display system** ステートメントを使用します。

エージェント実行中の Essbase セキュリティ・ファイルのコンパクト化

セキュリティ・ファイルのコンパクト化は、手動で実行する他に、最適化処理を自動的に開始する断片化比率を SECURITYFILECOMPACTIONPERCENT 構成設定に定義して、自動実行することもできます。

➤ エージェントを停止することなくセキュリティ・ファイルをコンパクト化するには、次のツールを使用します:

ツール	トピック	場所
エージェント	COMPACT	Essbase サーバー・ウィンドウにあるコマンド・プロンプトでエージェント・コマンドを入力します
MaxL	**alter system compact security file**	『Oracle Essbase テクニカル・リファレンス』
`essbase.cfg`	SECURITYFILECOMPACTIONPERCENT	『Oracle Essbase テクニカル・リファレンス』

注： エージェントの実行中にセキュリティ・ファイルの断片化を解消すると、その操作が完了するまでエージェントの動作が遅くなります。コンパクト化操作は、数分かかる場合もあります。

Essbase セキュリティ・ファイルの読取り可能フォーマットへのエクスポート

Essbase 管理者は、Essbase サーバー・インスタンスの `essbase.sec` ファイルのコンテンツを、読取り可能なテキスト・ファイル・フォーマットにエクスポートできます。これは、確認するのに役立ちます。

注意 `essbase.sec` をエクスポートする場合は、データの整合性を保証するために、会社のセキュリティ手順に従ってください。

管理サービス・コンソールから、または MaxL ステートメントとして実行可能な export security file コマンドは、現在ログインしている Essbase サーバー・セッションに対して実行されます。Essbase サーバー・セッションは、サービスとして実行できます。

➤ essbase.sec をエクスポートするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	セキュリティ・ファイルのエクスポート	Oracle Essbase Administration Services Online Help
MaxL	**export security_file**	『Oracle Essbase テクニカル・リファレンス』

注： DUMP エージェント・コマンドは、DUMP コマンドがサービスとして Essbase サーバーに対して実行できないのを除き、export security file コマンドと同じです。表 133 を参照してください。

41 セキュリティ・フィルタを使用したデータベース・セルへのアクセスの制御

この章の内容

セキュリティ・フィルタについて

アプリケーション、データベース、ユーザーおよびグループに対して定義されたセキュリティ・レベルが不足している場合は、Essbase セキュリティ・フィルタによって、より具体的な制御が提供されます。フィルタを使用すると、どういう種類のアクセスが、データベースのどの部分に、これらの設定が適用されるどのユーザーに許可されているかを定義することによって、データベース内の個々のデータへのアクセスを制御できます。

管理者権限がある場合は、任意のユーザーやグループに対して、任意のフィルタを定義して、割り当てられます。これらのフィルタは、自分自身には影響しません。

アプリケーションの作成/削除の権限がある場合は、自分が作成したアプリケーションに対するフィルタの割当ておよび定義が可能です。

アプリケーション・マネージャまたはデータベース・マネージャの権限がある場合は、自分のアプリケーションまたはデータベース内のフィルタを定義および割り当てできます。

セキュリティ・フィルタを使用した権限の定義

フィルタは、データ値(つまりセル)へのセキュリティ・アクセスを制御します。データベースの特定の部分に対するセキュリティのニーズに対応するフィルタを作成します。フィルタを定義する場合は、特定のデータベース・セルに対する制限を指定します。フィルタは保存されるときに、他のフィルタと区別するための一意の名前が付けられ、サーバーによってセキュリティ・ファイル `essbase.sec` に保管されます。その後、Essbase サーバー上の任意のユーザーまたはグループにそのフィルタを割り当てることができます。

たとえば、利益情報を含むセルへのアクセスを制限するために、マネージャが RED という名前のフィルタを設計し、それをデータベースに関連付けるとします。このフィルタは REVIEWERS という名前の訪問者グループに割り当てられ、これらのユーザーがデータベースの大部分を読み取れるが変更はできず、「利益」データ値にはアクセスできないようにします。

フィルタは、データベース・メンバーに対する 1 つ以上のアクセス設定で構成されます。次のアクセス・レベルを指定し、それをメンバーのリストから 1 つのセルまでの範囲のデータに適用できます:

- なし: 指定したメンバー・リストに対しては、データの取得も更新もできません。
- 読取り: 指定したメンバー・リストに対しては、データの取得はできますが、更新はできません。
- 書込み: 指定したメンバー・リストに対しては、データの取得および更新ができます。
- メタ読取り: 指定のメンバーに対して、メタデータ(次元名およびメンバー名)の取得および更新ができます。

注: メタ読取りアクセス・レベルによって、他のすべてのアクセス・レベルが上書きされます。データに対して追加のフィルタが定義されている場合、それらのフィルタは、定義されているすべてのメタ読取りフィルタ内で適用されます。代替変数に対してメタ読取りフィルタを割り当て、その代替変数を取得しようとすると、未知のメンバーのエラーが発生しますが、代替変数の値は表示されます。これは予期される動作です。パーティション内でメタデータ・セキュリティを完全にオフにはできません。そのため、ソース・データベースでメタデータ・セキュリティを設定しないでください。設定すると、ターゲット・パーティションで正しくないデータが生成される可能性があります。メタデータ・セキュリティがオンになっており、子に共有メンバーを含むメンバーに対してドリル・アップまたは取得を実行すると、共有メンバーの元のメンバーがフィルタされるため、未知のメンバーのエラーが発生します。このエラーを回避するには、共有メンバーの元のメンバーにメタデータ・セキュリティ・アクセス権を付与します。

フィルタ定義で指定されていないセルはすべて、データベース・アクセス・レベルを継承します。ただし、よりデータ固有のフィルタ定義は一般的なデータベース・アクセス・レベルより高い詳細レベルを示すため、データベース・レベルで割り当てられたアクセス権をフィルタで追加または削除できます。

フィルタ定義の対象にならないデータ値には、まずユーザーについて定義したアクセス・レベルが適用され、Essbase がネイティブ・セキュリティ・モードにある場合は、次にグローバルなデータベース・アクセス・レベルが適用されます。

計算アクセス権は、ユーザーおよびグループに付与された権限によって制御されます。データベースに対する計算アクセス権を持つユーザーは、フィルタによってブロックされず、その計算の実行によって更新されるすべてのデータ要素に影響を与えられます。Essbase がネイティブ・セキュリティ・モードにある場合は、計算アクセス権もまた、アプリケーションまたはデータベースに対する最小限のグローバル権限によって制御されます。

注： MDX 計算済メンバーの計算中に、ユーザーにアクセス権のないセルは #MISSING セルとして扱われます。

フィルタの作成

データベース値に対して加える必要のある一連のアクセス制限ごとにフィルタを作成できます。同じアクセス・ニーズを持つユーザーに対して、個別のフィルタを作成する必要はありません。フィルタを作成した後、それを複数のユーザーまたはユーザーのグループに割り当てられます。ただし、1 人のユーザーまたは 1 つのグループに割り当てられるフィルタは、1 つのデータベースにつき 1 つのみです。

注： 1 組のメンバー(子や子孫など)を戻す計算関数を使用していて、その結果が空集合になる場合は、セキュリティ・フィルタは作成されません。領域定義の結果が空集合になったことを示すエラーがアプリケーション・ログに書き込まれます。

フィルタを作成する前に、次のアクションを実行してください:

- サーバーに接続し、フィルタと関連付けるデータベースを選択します。
- 付録 A「制限」にあるフィルタの命名ルールを確認してください。

➤ フィルタを作成するには、次のツールを使用します:

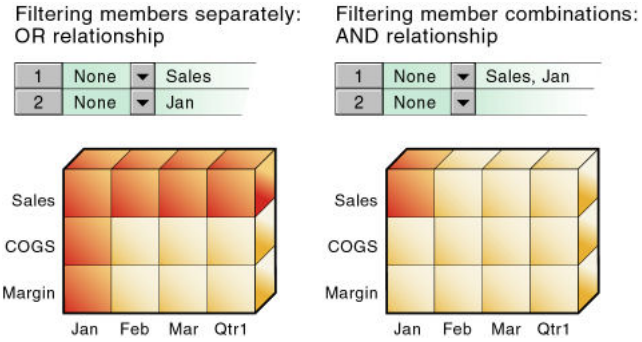
ツール	トピック	場所
Administration Services	フィルタの作成または編集	Oracle Essbase Administration Services Online Help
MaxL	**create filter**	『Oracle Essbase テクニカル・リファレンス』

メンバーへのフィルタとメンバーの組合せへのフィルタ

図 144 は、データベース・セルへのアクセスを制御するための異なる方法を示しています。データは、メンバー全体にフィルタをかけるか、メンバーの組合せにフィルタをかけることによって保護できます。

- 各メンバーに個別にフィルタをかけた場合には、各メンバーのすべてのデータにフィルタがかかります。
- メンバーの組合せにフィルタをかけた場合には、メンバーの交差点にあるデータにフィルタがかかります。

図 144　データの AND/OR 関係に対するフィルタの影響



注： メンバーの組合せに対するフィルタリング(AND 関係)は、メタ読取りには適用されません。メタ読取りでは、各メンバーに個別にフィルタをかけます(OR 関係)。

個々のメンバーに対するフィルタ

1 つ以上のメンバーのすべてのデータにフィルタをかけるには、フィルタ・エディタで、各メンバーの独自の行に対するアクセスを定義します。フィルタの別々の行に対するフィルタ定義は、OR 関係で処理されます。

たとえば、売上または 1 月へのアクセスをブロックするために、ユーザー KSmith に次のフィルタが割り当てられているとします:

- アクセス: なし。メンバー指定: 売上。
- アクセス: なし。メンバー指定: 1 月。

次回ユーザー KSmith が Sample.Basic にアクセスすると、Qtr1 の収益マージンのスプレッドシート表示(図 145)では、このユーザーがメンバー Sales、メンバー Jan のデータ値にアクセスできない、つまり#NOACCESS が付けられていることを示しています。メンバー Sales の内外で、ビューの売上および 1 月のすべてのデータはブロックされています。売上の兄弟でマージンの子である COGS(売上原価)のデータは、1 月の COGS を例外として、アクセスできます。

図 145　売上または 1 月へのアクセスをブロックするフィルタの適用結果

	A	B	C	D	E
1				Product	Market
2	Sales	Jan	Scenario	#NoAccess	
3		Feb	Scenario	#NoAccess	
4		Mar	Scenario	#NoAccess	
5		Qtr1	Scenario	#NoAccess	
6	COGS	Jan	Scenario	#NoAccess	
7		Feb	Scenario	14307	
8		Mar	Scenario	14410	
9		Qtr1	Scenario	42877	
10	Margin	Jan	Scenario	#NoAccess	
11		Feb	Scenario	17762	
12		Mar	Scenario	17803	
13		Qtr1	Scenario	52943	

メンバーの組合せに対するフィルタ

メンバーの組合せのデータにフィルタをかけるには、フィルタ・エディタで、行を使用してメンバーの各組合せのアクセスを定義します。フィルタ定義で、カンマで区切られた 2 つのメンバー・セットは、これらの 2 つのメンバー・セットの結合として処理されます(AND 関係)。

たとえば、ユーザー RChinn が次のフィルターに割り当てられているとします: アクセス: なし。メンバー指定: 売上、1 月。

次回ユーザー RChinn が Sample.Basic にアクセスすると、Qtr1 の収益マージンのスプレッドシート表示(図 146)では、このユーザーがメンバー Sales と Jan の交差でデータ値にアクセスできない、つまり#NoAccess が付けられていることを示しています。ビューの 1 月の売上データはブロックされています。ただし、他の月の売上データ、および 1 月の売上以外のデータはアクセス可能です。

図 146　売上、1 月へのアクセスをブロックするフィルタの適用結果

	A	B	C	D	E
1			Product	Market	Scenario
2	Sales	Jan	#NoAccess		
3		Feb	32069		
4		Mar	32213		
5		Qtr1	95820		
6	COGS	Jan	14160		
7		Feb	14307		
8		Mar	14410		
9		Qtr1	42877		
10	Margin	Jan	17378		
11		Feb	17762		
12		Mar	17803		
13		Qtr1	52943		

フィルタでの代替変数の使用

代替変数を使用すると、定期的に変更される情報をより容易に管理できます。各代替変数には、割り当てられた名前と値があります。データベース・マネージャは、この値をいつでも変更できます。フィルタで代替変数が指定された場合は、その時点の代替変数の値が使用されます。

たとえば、ユーザーのグループに現在の月のデータのみを表示したい場合は、CurMonth という名前の代替変数を設定し、フィルタ(MonthlyAccess)を定義できます。ここで、メンバー名に`&CurMonth` を使用してアクセスを指定します。指定の先頭にアンパサンド(&)を使用すると、Essbase では、メンバー名ではなく代替変数として識別されます。MonthlyAccess フィルタを該当するユーザーに割り当てます。

毎月、CurMonth 代替変数の値を現在の月のメンバー名(1 月や 2 月など)に変更するだけで済みます。この新しい値は、割り当てられているすべてのユーザーに適用されます。

118 ページの「代替変数の使用」を参照してください。

属性関数によるフィルタ

フィルタを使用して、特定の属性を共有している基本メンバーのデータへのアクセスを制限できます。属性次元で定義された特定の属性を持つメンバーのデータにフィルタをかけるには、属性メンバーを@ATTRIBUTE 関数または@WITHATTR 関数と組み合せて使用します。

注： @ATTRIBUTE と@WITHATTR はメンバー・セット関数です。フィルタ定義では、ほとんどのメンバー・セット関数を使用できます。

たとえば、ユーザー PJones に次のフィルタが割り当てられているとします。アクセス: なし。メンバー指定: @ATTRIBUTE(Caffeinated_False)。

次回ユーザー PJones が Sample.Basic にアクセスすると、カリフォルニアでの第 1 四半期のコーラの売上のスプレッドシート表示(図 147)では、Caffeinated_False に関連付けられた基本次元メンバーのデータ値にアクセスできないことを示しています。ビューのカフェインなしのコーラの売上データはブロックされています。カフェインなしのコーラは基本メンバーで、Caffeinated_False は属性次元 Caffeinated の関連付けられたメンバー(前述のスプレッドシート表示には表示されない)ことに注意してください。

図 147　カフェインなしの製品へのアクセスをブロックするフィルタの適用結果

	Sales	California	Qtr1	Actual
Cola	1998			
Diet Cola	367			
Caffeine Free Cola	#Miss			
Colas	2767			

メタデータのフィルタリング

メタデータのフィルタリングにより、データのフィルタリングおよび追加のセキュリティ層が提供されます。メタデータのフィルタリングを使用すると、管理者はユーザーのビューからアウトライン・メンバーを削除して、ユーザーに関与するメンバーのみへのアクセスを提供できます。

メタ読取り権限をメンバーに適用するときにフィルタを使用すると、次のような効果があります:

1. そのメンバーのすべての祖先のデータがフィルタ・ユーザーのビューで非表示になります。
2. そのメンバーのすべての兄弟のデータおよびメタデータ(メンバー名)がフィルタ・ユーザーのビューで非表示になります。

フィルタの管理

参照:

- 705 ページの「フィルタの表示」

フィルタの表示

➤ フィルタのリストを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	フィルタの作成または編集	Oracle Essbase Administration Services Online Help
MaxL	**display filter**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	LISTFILTERS	『Oracle Essbase テクニカル・リファレンス』

フィルタの編集

➤ フィルタを編集するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	フィルタの作成または編集	Oracle Essbase Administration Services Online Help
MaxL	**create filter**	『Oracle Essbase テクニカル・リファレンス』

フィルタのコピー

自分の権限に従って、フィルタを任意の Essbase サーバー上のアプリケーションおよびデータベースにコピーできます。また、アプリケーション移行の一部として、フィルタもサーバー間でコピーできます。

➤ フィルタをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	フィルタのコピー	Oracle Essbase Administration Services Online Help
MaxL	**create filter**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYFILTER	『Oracle Essbase テクニカル・リファレンス』

フィルタの名前変更

➤ フィルタの名前を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	フィルタの名前変更	Oracle Essbase Administration Services Online Help
MaxL	**create filter**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	RENAMEFILTER	『Oracle Essbase テクニカル・リファレンス』

フィルタの削除

➤ フィルタを削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	フィルタの削除	Oracle Essbase Administration Services Online Help
MaxL	**drop filter**	『Oracle Essbase テクニカル・リファレンス』

フィルタの割当て

フィルタを定義した後、それらのフィルタをユーザーまたはグループに割り当てられます。それにより、同じフィルタ設定を必要とする複数のユーザーを管理できます。フィルタの定義への変更は、そのフィルタのユーザーによって自動的に継承されます。

フィルタは、管理者の役割を持つユーザーには影響を与えません。1 人のユーザーまたは 1 つのグループに割り当てられるフィルタは、1 つのデータベースにつき 1 つのみです。

EPM System セキュリティ・モードでのファイルの割当て

EPM System セキュリティ・モードでは、Oracle Hyperion Shared Services Console を使用してフィルタを割り当てます。

➤ ユーザーまたはグループにフィルタを割り当てるには、660 ページの「Shared Services でのデータベース計算およびフィルタ・アクセスの割当て」を参照してください。

Essbase ネイティブ・セキュリティ・モードでのフィルタの割当て

➤ ユーザーまたはグループにフィルタを割り当てるには、『Oracle Essbase Administration Services オンライン・ヘルプ』のフィルタの割当てに関する項を参照してください。

フィルタ定義のオーバーラップ

フィルタ内に同一のメンバー指定を持つ行が複数含まれている場合には、次のルールが記載順に適用されて、継承されるアクセス権が決定されます:

1. より詳細な次元の組合せリストを定義するフィルタは、より一般的なフィルタに優先します。
2. 前述のルールによってオーバーラップの競合が解決しない場合は、オーバーラップするフィルタ行の中で、最も高いアクセス・レベルが適用されます。

たとえば、次のフィルタには、オーバーラップの競合が含まれます:

- アクセス: 書込み。メンバー指定: 実績。
- アクセス: なし。メンバー指定: 実績。
- アクセス: 読取り。メンバー指定: 実績、@IDESCENDANTS(New York)。

3 番目の指定は、他の 2 つより高い詳細レベルでセキュリティを定義します。そのため、ニューヨーク支店内のメンバーのすべての実際データに読取りアクセス権が付与されます。

書込みアクセス権は、なしよりアクセス・レベルが高いため、実際内の残りのデータ値には書込みアクセス権が付与されます。

その他のすべてのセル(予算など)には、最小限のデータベース権限に従ってアクセスできます。

書込みアクセス権を持っているユーザーは、読取りアクセス権も持っています。

注: データベース・アウトライン内のメンバーへの変更は、フィルタに自動的には反映されません。変更されたメンバー参照を手動で更新する必要があります。

メタデータのフィルタ定義の重複

任意の行内の影響を受けるメンバー・セット(メタ読取りメンバーとその祖先)が、他の行のメタ読取りメンバーと重複していない場合のみ、複数行を使用したメタ読取りフィルタを定義する必要があります。複数行に対するメタ読取りを含むフィ

ルタでは、行ごとに 1 つの次元を指定することをお薦めします。ただし、祖先とメタ読取りメンバーが重複していないかぎり、1 つの次元の異なるメンバーを複数のメタ読取り行の中に指定することは有効です。

たとえば、Sample Basic では、次のフィルタ定義に重複があります:

- アクセス: メタ読取り。メンバー指定: California。
- アクセス: メタ読取り。メンバー指定: West。

最初の行では、メタ読取りを California に適用すると、California にアクセスできるようになりますが、その祖先へのアクセスはブロックされます。そのため、West に対するメタ読取りは無視されます。このフィルタが割り当てられているユーザーは、West へアクセスできません。

California のみでなく West にもメタ読取りアクセスを割り当てるには、これらの行を次の 1 行にまとめます。アクセス: メタ読取り。メンバー指定: California、West。

アクセス定義のオーバーラップ

ユーザーおよびグループのアクセス権の定義がオーバーラップする場合は、次のルールが、記載順に適用されます:

1. より詳細な次元の組合せリストを定義するアクセス・レベルは、より一般的なアクセス・レベルに優先します。
2. 前述のルールによってオーバーラップの競合が解決しない場合は、最も高いアクセス・レベルが適用されます。

例 1:

ユーザーフレッドには、次のデータベース・アクセスが定義されています:

```
   FINPLAN   R
CAPPLAN   W
PRODPLAN   N
```

彼は、グループ・マーケティングに割り当てられており、グループ・マーケティングには、次のデータベース・アクセスが付与されています:

```
   FINPLAN   N
CAPPLAN   N
PRODPLAN   W
```

彼の有効な権限は、次のように設定されています:

```
   FINPLAN   R
CAPPLAN   W
PRODPLAN   W
```

例 2:

ユーザー・メアリーには、次のデータベース・アクセスが定義されています:

```
   FINPLAN   R
PRODPLAN   N
```

彼女は、グループ・マーケティングに割り当てられており、グループ・マーケティングには、次のデータベース・アクセスが付与されています:

```
   FINPLAN   N
PRODPLAN   W
```

彼女の有効な権限は、次のように設定されています:

```
   FINPLAN   R
PRODPLAN   W
```

さらに、メアリーは(データベース FINPLAN のための)フィルタ・アーティファクト RED も使用しています。このフィルタには、次の 2 つのフィルタ行があります:

- アクセス: 読取り。メンバー指定: 実績。
- アクセス: 書込み。メンバー指定: 予算、@IDESCENDANTS(New York)。

また、グループ・マーケティングは(データベース FINPLAN のための)フィルタ・アーティファクト BLUE も使用しています。このフィルタには、次の 2 つのフィルタ行があります:

- アクセス: 読取り。メンバー指定: 実際、売上。
- アクセス: 書込み。メンバー指定: 予算、売上。

フィルタのオーバラップから、メアリーの有効な権限と、メアリーおよびメアリーのグループに割り当てられる権限は次のようになります:

- R: FINPLAN データベース全体。
- W: New York ブランチ内のすべての予算データに対して。
- W: 予算および売上に関連するデータ値に対して。

42 ネイティブ・セキュリティ・モードでのセキュリティの例

この章の内容

この章で説明するセキュリティに関するサンプルの問題とソリューションは、サンプル・アプリケーションに基づいており、第 38 章「EPM System セキュリティ・モードでのユーザーの管理およびセキュリティ」で説明するセキュリティ手順を使用しています。

例 1: ユーザーにはデータベースに対する同じアクセス権が必要

スー・スミス、ビル・ブラウン、ジェイン・ジョーンズの 3 名の従業員が Essbase を使用する必要があります。各従業員には、サンプル・アプリケーション内のすべてのデータベースに対する更新アクセス権が必要です。

Essbase をネイティブ・セキュリティ・モードで使用する場合のソリューション:

ユーザーは、1 つのアプリケーションのみに対する更新アクセス権を必要としているため、管理者権限は必要ありません。ユーザーは、アプリケーション、ユーザーまたはグループを作成または削除する必要がないため、作成/削除の権限を持つ特殊なタイプのユーザーとして定義されている必要はありません。これらのユーザーには、サンプル・アプリケーションに対するアプリケーション・マネージャ権限のみが必要です。

管理者は、次の操作を行う必要があります:

1. Administration Services を使用してユーザーを設定します。

   Oracle Essbase Administration Services Online Help を参照してください。

2. スー、ビルおよびジェインを、アプリケーション・マネージャ権限を持つ通常のユーザーとして作成します。

スー、ビルおよびジェインをアプリケーション・マネージャ権限なしに作成してある場合は、これら 3 人のユーザーに、アプリケーション・マネージャ権限を割り当てます。

673 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの作成」または 677 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーおよびグループへのデザイナ権限の付与」を参照してください。

例 2: ユーザーにはデータベースに対する異なるアクセス権が必要

スー・スミス、ビル・ブラウン、ジェイン・ジョーンズの 3 名の従業員が Essbase を使用する必要があります。スーおよびビルには、サンプル・アプリケーション内のすべてのデータベースに対するフル・アクセス権が必要です。ジェインには、サンプル・アプリケーション内のすべてのデータベースに対する完全な計算アクセス権が必要ですが、ジェインがデータベース定義を定義または維持する必要はありません。

Essbase をネイティブ・セキュリティ・モードで使用する場合のソリューション:

管理者は、次の操作を行う必要があります:

1. Administration Services を使用してユーザーを設定します。

   Oracle Essbase Administration Services Online Help を参照してください。

2. アプリケーション・マネージャ権限を持つ通常のユーザーとして、Sue および Bill を作成します。

   スーおよびビルをアプリケーション・マネージャ権限なしに作成してある場合は、これら 2 人のユーザーにアプリケーション・マネージャ権限を割り当てます。

   673 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの作成」または 677 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーおよびグループへのデザイナ権限の付与」を参照してください。

3. サンプル・アプリケーションに対する最小限のデータベース・アクセス設定として、グローバルな計算アクセス権を定義し、すべての追加ユーザーに、サンプル・アプリケーションのすべてのデータベースに対する計算アクセス権を付与します。

   Oracle Essbase Administration Services Online Help の「アプリケーションの最小権限の設定」を参照してください。

4. ジェインを、追加の権限を持たない通常のユーザーとして作成します。ジェインは、アプリケーションのグローバル設定から計算アクセス権を継承します。

   673 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの作成」を参照してください。

例 3: ユーザーにはデータベースに対する異なるアクセス権が必要であり、ユーザーが追加される

スー・スミス、ビル・ブラウン、ジェイン・ジョーンズの 3 名の従業員が Essbase を使用する必要があります。スーおよびビルには、サンプル・アプリケーション内のすべてのデータベースに対するフル・アクセス権が必要です。ジェインには、サンプル・アプリケーション内のすべてのデータベースに対する完全な更新および計算アクセス権が必要ですが、ジェインがデータベース定義を定義または維持することはありません。ユーザーが追加されます。これらのすべてのユーザーに、すべてのデータベースに対する読取りアクセス権が必要です。

Essbase をネイティブ・セキュリティ・モードで使用する場合のソリューション:

現在のユーザーが必要としているアプリケーションおよびデータベースに対するアクセス権はそれぞれ異なるため、それらの権限を個別に定義します。次に、将来のユーザーに個々の読取り権限を割り当てる時間を節約するために、読取りをアプリケーションのグローバル設定にします(ユーザー権限やグローバル・アクセス権を割り当てる順序は問題になりません)。

管理者は、次の操作を行う必要があります:

1. Administration Services を使用してユーザーを設定します。

   Oracle Essbase Administration Services Online Help を参照してください。

2. アプリケーション・マネージャ権限を持つ通常のユーザーとして、スーおよびビルを作成または編集します。

   673 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの作成」および 677 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーおよびグループへのデザイナ権限の付与」を参照してください。

3. ジェインを通常のユーザーとして作成し、ジェインにサンプル・アプリケーションの計算権限を付与します。

   673 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの作成」および 676 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーやグループへのアプリケーション・アクセス権およびデータベース・アクセス権の付与」を参照してください。

4. サンプル・アプリケーションに対する最小限のデータベース・アクセス設定として、グローバルな読取りのアクセス権を定義し、すべての追加ユーザーに、サンプル・アプリケーションのすべてのデータベースに対する読取りのアクセス権を付与します。

   Oracle Essbase Administration Services Online Help の「データベースの最小権限の設定」を参照してください。

例 4: ユーザーにはアプリケーションおよびデータベースに対する異なるアクセス権が必要

スー・スミス、ビル・ブラウン、ジェイン・ジョーンズの 3 名の従業員が Essbase を使用する必要があります。スーには、サンプル・アプリケーションに対するフル・アクセス権のみが必要です;ジェインには Basic データベースのすべてのメンバーに対する計算アクセス権が必要です。ビルには、すべてのメンバーに対する読取りアクセス権が必要です。その他のユーザーにはデータベースに対するアクセス権を付与しないようにします。

またジェインとビルは、スーが定義するレポート・スクリプトを実行する必要があります。

Essbase をネイティブ・セキュリティ・モードで使用する場合のソリューション:

ユーザーが必要としているアプリケーションおよびデータベースへのアクセス権はそれぞれ異なるので、グローバル・アクセス設定をなしに定義し、ユーザー権限を個別に割り当てます。

管理者は、次の操作を行う必要があります:

1. Administration Services を使用してユーザーを設定します。(ジェインおよびビルはレポート・スクリプトを実行する必要があるため、これら 2 人のユーザーは Administration Services を使用する必要があります。)

   Oracle Essbase Administration Services Online Help を参照してください。

2. スーを通常のユーザーとして作成し、サンプル・アプリケーションのアプリケーション・マネージャ権限を付与します。

   673 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの作成」および 677 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーおよびグループへのデザイナ権限の付与」を参照してください。

3. ジェインを通常のユーザーとして作成し、ジェインにサンプル・アプリケーションの計算権限を付与します。

   673 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの作成」および 676 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーやグループへのアプリケーション・アクセス権およびデータベース・アクセス権の付与」を参照してください。

4. ビルを通常のユーザーとして作成し、ビルにサンプル・アプリケーションの読取りの権限を付与します。

   673 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの作成」および 676 ページの「Essbase ネイティブ・セキュリティ・モードにおけるユーザーやグループへのアプリケーション・アクセス権およびデータベース・アクセス権の付与」を参照してください。

例 5: 管理者がメンテナンスを実行する必要がある

管理者スー・スミスは、サンプル・アプリケーションに関するメンテナンスを実行する必要があります。スーは、データベース・アウトラインを変更し、データを再ロードする必要があります。アプリケーションを変更している間は、他のユーザーがアプリケーションに接続できないようにする必要があります。

Essbase をネイティブ・セキュリティ・モードで使用する場合のソリューション:

スーは、次のタスクを実行する必要があります:

1. 「コマンドを許可」設定を使用不可にして、他のユーザーがアプリケーションに接続できないようにするとともに、接続済ユーザーがその後の操作を実行できなくなるようにします。

   Oracle Essbase Administration Services Online Help の「アプリケーションに対するユーザー・アクティビティの遮断」を参照してください。

2. アクティブなロックを持つユーザーがいるかどうかを確認します。

   アクティブなロックを持つユーザーがいる場合は、スーの計算またはデータ・ロード・コマンドが、ロックされたレコードへのアクセスを待機して停止する可能性があります。スーは、そのユーザーに更新を完了させるか、またはそのロックを解除できます。

   Oracle Essbase Administration Services Online Help の「データのロックの表示」を参照してください。

3. アクティブなロックを持つユーザーがいないことを確認した後、アプリケーションのメンテナンス作業を開始します。

第 VII 部

Unicodeによる複数言語アプリケーションの使用可能化

Unicode による複数言語アプリケーションの使用可能化の内容：

43 EssbaseのUnicode実装の理解

この章の内容

この章の情報は、ブロック・ストレージと集約ストレージ・データベースに適用されます。

Unicode について

国や言語の境界を超えてデータを共有することは、多国籍企業にとっての課題です。従来より、各コンピュータは、ロケール指定に基づいてテキストを保管したり、表示したりしてきました。ロケールは、通貨や日付のフォーマット、データのソート順、文字セットのエンコード方式などの、ローカル言語および文化的な規則を識別します。エンコード方式とは、文字テキストを、コード・ページまたはエンコード方式フォーマットで定義されたデータとして保管するために使用されるビットの組合せを指します。Essbase では、コード・ページによって、文字が非 Unicode エンコード方式のビットの組合せにマッピングされます。

異なるエンコード方式を使用すれば、異なる文字に同じビットの組合せをマッピングする可能性があるため、あるコンピュータで作成されたファイルは、異なるロケールの別のコンピュータ上では、誤って解釈される可能性があります。

Unicode 標準を使用すると、ロケールの異なるコンピュータが文字データを共有できるようになります。Unicode によって、複数の言語の文字セットを同時にサ

ポートするために十分な、数千のビットの組合せを持つエンコード形式が提供されます。Unicode では、すべての文字マッピングを 1 つのエンコード形式に結合することにより、ロケール設定の異なるコンピュータ上で作成された文字データをユーザーが正しく表示できます。

Essbase は、Unicode 標準のバージョン 2.1 に準拠し、UTF-8 エンコード方式を使用しています。www.unicode.org を参照してください。

Essbase では、Unicode の実装により、グローバル企業の従業員が、Essbase データベースに保管されている企業情報を自国の言語で表示できるようにしています。たとえば、それぞれの言語の別名テーブルを使用して、台湾のユーザーはデータベース・レポートを漢字で表示でき、フランスのユーザーは同じレポートをフランス語の文字で表示できます。

以降のトピックでは、Essbase での Unicode 実装の特徴について説明します。

注： Essbase では、ユーザー定義の文字セット(UDC)はサポートされていません。

Unicode モードのアプリケーションを使用する場合

Unicode モードのアプリケーションは、次の状況でのみ使用することを検討してください:

- 異なる言語を使用するユーザーが、自国の言語および文字セットを使用して、同じデータベースの情報を表示できるようにする必要がある場合。

  たとえば、日本語とドイツ語の別名テーブルを使用して、日本とドイツのユーザーが共通の製品セットに関する情報を自国の言語で表示できるようにする場合です。

- 非 Unicode モードのアプリケーションでサポートされている長さを超えるアーティファクト名を扱う必要がある場合。

  たとえば、アプリケーション名やデータベース名を、8 文字よりも長くする必要がある場合や、マルチバイトの文字セットを使用して、さらに長い文字数のアーティファクト名を扱う必要がある場合です。

  付録 A「制限」を参照してください。

- 変換されたマルチバイトの Essbase 実装で、「ラウンドトリップ」の問題が発生しました。2 つの異なるビット値が同じ文字にマッピングされる可能性があります。この問題は、マルチバイトのオペレーティング・システムやアプリケーション・プログラム間の通信で発生することがあります。

  Java アプリケーションとして、Administration Services および Provider Services は常に Unicode で動作するため、ラウンドトリップの変換エラーは発生しません。

Unicode モードのアプリケーションを使用するかどうかを決定する場合は、次の点に注意してください:

- Unicode モードのアプリケーションで非 Unicode テキスト・ファイルを使用するには、ロケールとその管理方法を理解していることが必要です。データベースの破損を引き起こす可能性のあるエラーを防止するために、UTF-8 でエンコードされたファイルを使用することをお薦めします。

  732 ページの「ファイルのエンコード方式の管理」を参照してください。

- Unicode モードのアプリケーションを操作するには、Unicode モードのアプリケーションで使用される長い文字列を使用するために、非 Unicode モードのアプリケーションをサポートするために記述されたカスタム・クライアント・プログラムを作成する必要があります。この作業は、アプリケーションの設計に応じて、簡単な再構築で済む場合と再プログラミングが必要になる場合があります。変更されるプログラムのコーディング方法によっては、必要なメモリーが増えることがあります。

  『Oracle Essbase API リファレンス』を参照してください。

ロケールおよび ESSLANG 変数

Essbase では、コンピュータのロケールを定義するための ESSLANG 変数を使用しています。たとえば、アメリカ英語をサポートするには、ESSLANG を `English_UnitedStates.Latin1@Binary` に設定できます。

Essbase サーバー・インストールごとに、ESSLANG 変数を指定する必要があります。この変数は、コンピュータのオペレーティング・システムで定義されたロケールに設定されている必要があります。

クライアント・コンピュータでは、ESSLANG 変数の指定はオプションです。ESSLANG が定義されている場合、Essbase は、この ESSLANG 値をコンピュータ・ロケールとして使用します。ESSLANG が指定されていな場合は、オペレーティング・システムのロケールが使用されます。

ESSLANG 変数は、Essbase のインストール時に設定されます。Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

非 Unicode モードのクライアントおよびアプリケーションの場合は、クライアントと Essbase サーバーのロケール値が同じである必要があります。Unicode モードのアプリケーションの場合は、クライアントと Essbase サーバーのロケール値が異なっていてもかまいません。

Unicode モードのアプリケーションおよび非 Unicode モードのアプリケーション

アプリケーションは、Unicode モードのアプリケーションまたは非 Unicode モードのアプリケーションのいずれかに指定されます。

Unicode モードのアプリケーションは、複数の文字セットをサポートしています。Unicode モードのアプリケーションを使用している場合、Essbase は UTF-8 エンコード形式を使用して、文字テキストを解釈および保管します。Unicode モードのア

プリケーションでの文字ベースのアーティファクト(メンバー名や別名の名前など)には、異なる言語の文字を含められます。

Unicode モードのアプリケーションは、非 Unicode エンコード方式および UTF-8 の入力ファイルを受け入れるため、Essbase はロケール・インディケータとユーザー・プロンプト表示を使用して、非 Unicode でエンコードされたファイルの読取りまたは書込みを行います。

Unicode モードのアプリケーションで動作しているクライアントは、Essbase サーバーとは異なるロケールにできます。たとえば、ロケールが日本語のクライアント・コンピュータと、ロケールがドイツ語のクライアント・コンピュータは、ロケールがスペイン語の Essbase サーバーのインスタンス上の同じ Unicode モードのアプリケーションで動作できます。

Unicode モードのアプリケーションの場合、ほとんどのアーチファクトには、非 Unicode モードのアプリケーションの場合よりも、長い名前を付けられます。アーチファクト名の長さは、バイト数ではなく、文字数に基づいて計算されます。723 ページの「名前の長さの拡張」を参照してください。

非 Unicode モードのアプリケーションは、ロケール値によって定義された 1 つの文字セットをサポートしています。このロケール値は、Essbase サーバーと、非 Unicode モードのアプリケーションで動作しているすべての非 Unicode クライアントで同じである必要があります。デフォルトでは、Essbase はアプリケーションを非 Unicode モードで作成します。

アプリケーションの作成時にそのアプリケーションを Unicode モードとして定義するか(729 ページの「Unicode モードのアプリケーションの作成」を参照)、非 Unicode モードのアプリケーションを別の手順で Unicode モードに移行できます(729 ページの「アプリケーションの Unicode モードへの移行」を参照)。

注： Unicode モードのアプリケーションを非 Unicode モードに変換できません。

Unicode モードのアプリケーションと非 Unicode モードのアプリケーションは同じ Essbase サーバー上に存在できます。

Essbase サーバーの Unicode モードと非 Unicode モード

Essbase サーバーは、Unicode モードのアプリケーションを作成する権限と、既存のアプリケーションを Unicode モードに移行する権限を持っているときに Unicode モードになります。728 ページの「Essbase サーバーの Unicode モードへの設定」を参照してください。

Essbase サーバーが Unicode モードにあるかどうかは、アプリケーションの作成および移行にのみ影響します。Essbase サーバーの Unicode の設定には関係なく、Unicode モードと非 Unicode モードの両方のアプリケーションを使用できます。

名前の長さの拡張

Unicode モードのアプリケーションでは、アプリケーション、データベース、メンバーの名前などの文字列で使用可能な最大文字数は、非 Unicode モードのアプリケーションの場合よりも多くなります。

Unicode のアーティファクト名の最大長は、各文字に必要なバイト数には関係なく、文字数に基づいています。非 Unicode のアーティファクト名の最大長は、バイト数で計算されます。

バイト数によって制限されないことは、中国語や日本語などのマルチバイトの文字セットを使用するアプリケーションにとって有利です。たとえば、Unicode モードのアプリケーションでのメンバー名の制限は、たとえマルチバイト文字であっても 80 文字です。これに対して、非 Unicode アプリケーションでのメンバー名の制限は 80 バイトです。付録 A「制限」を参照してください。

注： この拡張されたサイズ制限は、アウトラインやユーザーが記述するクライアント・プログラムのサイズに影響を与える可能性があります。

拡張された名前サイズを活用するために、すべてのユーザーが同じロケールで作業する場合でも、Unicode モードのアプリケーションを使用することがあります(720 ページの「Unicode モードのアプリケーションを使用する場合」を参照)。

異なるバージョンのクライアント・プログラムと Essbase アプリケーションの間の互換性

Essbase は、Unicode モードと非 Unicode モードの両方のクライアント・プログラムをサポートしています。

非 Unicode モードのクライアント・プログラム(たとえば、MaxL シェル):

- 短い文字列の Essbase API で通信します
- Unicode モードのアウトラインを変更できません
- Unicode モードのアウトラインのアウトライン同期を実行できません

Unicode モードのクライアント・プログラム(たとえば、管理サービス・コンソールや Smart View):

- 長い文字列を使用して、Unicode エンコード方式の Essbase API で通信します
- 非 Unicode モードのアウトラインをマルチバイト文字と同期できません

注： 管理サービス・コンソールまたは別のツールを使用して MaxL スクリプトを作成し、それを UTF-8 で保存してからそのスクリプトを MaxL シェルで実行すると、MaxL シェルは Unicode モードのクライアントの役割を前提にします。たとえば、この方法を使用すると、次元構築によってアウトラインを更新できます。スクリプトを作成する場合は、UTF-8 シグネチャを含めることを忘れないでください。734 ページの「エンコード方式のインディケータ」を参照してください。

Unicode 対応の C API

7.0 より前の Essbase リリースで使用されていたカスタム作成されたクライアント・プログラムは、これらのカスタム・プログラムが短い文字列および短いバッファを使用しているため、Unicode モードのアプリケーションでは使用できません。

Unicode モードのアプリケーションへの制限されたアクセスを提供するために、これらのプログラムの記述方法によっては、Essbase の Unicode 対応リリースで古いカスタム・クライアント・プログラムを再コンパイルできます。再コンパイルされると、これらのプログラムは、長いバッファおよび短い文字列で動作します。

Unicode モードのアプリケーションと非 Unicode モードのアプリケーションへの完全なアクセスのためには、Essbase の Unicode 対応の API 関数を使用して既存のカスタム・アプリケーションを変更する必要があります。書き換えられ、コンパイルされたクライアントは、Unicode で完全にサポートされるため、長いバッファおよび長い文字列で動作します。『Oracle Essbase API リファレンス』を参照してください。

テキストのエンコード方式とロケールの特定

Essbase は、Unicode モードのアプリケーションでの、非 Unicode でエンコードされたファイル(レポートや計算スクリプトなど)の使用をサポートしています。ファイルのエンコード方式の種類を識別するために、Essbase ではエンコード方式のインディケータ(UTF-8 シグネチャまたはロケール・インディケータ)を探します。ファイルにどちらのエンコード方式のインディケータも含まれておらず、そのファイルが Essbase サーバー上にない場合、Administration Services はユーザーにファイルのロケールを入力するよう求めます。734 ページの「エンコード方式のインディケータ」を参照してください。

Essbase Unicode ファイル・ユーティリティには、テキスト・ファイルに UTF-8 シグネチャまたはロケール・インディケータを挿入するためのオプションが含まれています。または、テキスト・エディタやその他の手段を使用して挿入もできます。739 ページの「Essbase Unicode ファイル・ユーティリティ」を参照してください。

Unicode 対応の管理ツール

Unicode モードのアプリケーションを管理するために、Administration Services および MaxL シェルを使用できます。Unicode モードのクライアントである管理サービ

ス・コンソールでは、Unicode モードのアプリケーションと非 Unicode モードのアプリケーションを管理できます。

Unicode に関連した管理操作には、Essbase サーバーの Unicode 関連モードを、次の機能を使用可能または使用不可にするように変更することが含まれます:

- Unicode モードのアプリケーションの作成
- 非 Unicode モードのアプリケーションの Unicode モードへの移行
- Essbase サーバーおよびアプリケーションの Unicode 関連ステータスの表示

第 44 章「Unicode モードのアプリケーションの管理」を参照してください。

取得ツール

Essbase には、Unicode モードのアプリケーションおよび非 Unicode モードのアプリケーションからデータを取得するためのいくつかの方法が用意されています。

レポート・スクリプト

レポート・スクリプトの出力ファイルは、アプリケーションのエンコード方式でエンコードされます。たとえば、レポート・スクリプトが Unicode モードのアプリケーション内のデータベースに対して実行された場合、そのレポート・スクリプトの出力は UTF-8 でエンコードされます。非 Unicode モードのアプリケーション内のデータベースに対して実行された場合、その出力はアプリケーションのエンコード方式でエンコードされます。

スプレッドシート

Smart View では、Unicode モードのアプリケーションおよび非 Unicode モードのアプリケーションでデータを表示できます。

Smart View を実行するには、Provider Services に接続します。これらのプログラムをインストールするには、Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

Smart View の操作方法については、Oracle Hyperion Smart View for Office User's Guide を参照してください。

SQL インタフェース

SQL インタフェースを使用すると、Unicode モードのリレーショナル・データベースから Unicode モードの Essbase アプリケーションにデータをロードできます。表 125 は、Unicode モードのアプリケーションおよび非 Unicode モードのアプリケーションでサポートされているエンコード方式を示しています:

表 125　SQL インタフェースでサポートされている Unicode モードのアプリケーションおよび非 Unicode モードのアプリケーションのエンコード方式

アプリケーションのモード	リレーショナル・データベースのエンコード方式
非 Unicode	サポートされる非 Unicode エンコード方式
Unicode	サポートされる非 Unicode エンコード方式
Unicode	UTF-8

SQL インタフェースでは、UTF-8 エンコード方式のユーザー認証(ユーザー名とパスワード)およびアプリケーション情報が受け入れられます。

737 ページの「サポートされているロケール」を参照してください。

Sample_U.Basic

Unicode モードのアプリケーションの詳細を学習するには、Sample_U.Basic アプリケーションを使用します。Sample_U.Basic のメンバー名は英語で表示されます。

表 126 は、Sample_U.Basic データベースに含まれている英語以外の別名テーブルとそのインポート・ファイルを示しています:

表 126　Sample_U.Basic データベース内の英語以外の別名テーブル

言語	英語以外の別名テーブル
中国語	nameschn.alt
ドイツ語	namesger.alt
日本語	namesjpn.alt
ロシア語	namesrsn.alt

44 Unicodeモードのアプリケーションの管理

この章の内容

この章の情報は、ブロック・ストレージと集約ストレージ・データベースに適用されます。

Unicode をサポートするためのコンピュータの設定

Unicode モードのアプリケーションを管理するコンピュータ上で、UTF-8 でエンコードされたテキストを処理するには、次のツール(サードパーティ・ベンダーから提供)をインストールします:

- UTF-8 フォント: 非 ASCII 文字を含む、UTF-8 でエンコードされたテキストを表示するため
- (オプション) Unicode エディタ(UTF-8 シグネチャを含む): データ・ソースやその他のテキスト・ファイルを手動で編集するため

Essbase では、テキスト・ファイルを UTF-8 エンコード方式に変換する Essbase Unicode ファイル・ユーティリティを提供しています。739 ページの「Essbase Unicode ファイル・ユーティリティ」を参照してください。

パスワード・セキュリティの定義

Essbase のセキュリティは、Essbase サーバーのレベルで定義されます。パスワードを作成するには、Essbase サーバー・コンピュータ上の ESSLANG 変数に指定されているコード・ページに従ってエンコードされた文字を使用します。

マルチバイト文字セットの環境でも、パスワードにマルチバイト文字を含められません。標準の ASCII 文字セットにある文字(0 から 127 までの範囲)を使用することを検討してください。この文字セットには、標準の英語のアルファベットにある大文字および小文字と数字の 0 から 9 までが含まれています。これらの文字は、すべてのコード・ページおよび UTF-8 に共通です。

アプリケーションとデータベースの名前の指定

アプリケーションとデータベースの命名時には、ESSLANG 変数で指定された文字セットによってサポートされている文字を使用する必要があります。Essbase サーバーがインストールされているコンピュータ上において、ESSLANG 変数はそのコンピュータのオペレーティング・システムに定義されているロケールに設定されている必要があります。

Unicode モードの Essbase サーバーの管理

このセクションのトピックでは、Unicode モードの Essbase サーバーを管理するためのプロセスについて説明します。

Essbase サーバーの Unicode モードへの設定

Essbase サーバーを Unicode モードに設定することにより、Unicode モードのアプリケーションを作成するための権限と、アプリケーションを Unicode モードに移行するための権限が付与されます。

➤ Essbase サーバーを Unicode モードに設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Unicode モードのアプリケーションを作成するための Essbase サーバーの権限の管理	Oracle Essbase Administration Services Online Help
MaxL	**alter system**	『Oracle Essbase テクニカル・リファレンス』

注： UNICODEENABLE 設定を `essbase.cfg` に追加すると、Essbase Server を Unicode モードに設定しなくても Unicode モード・アプリケーションを作成できます。Unicode モードのアプリケーションは、Essbase サーバーが Unicode モードに設定されていない場合でも使用できます。

Essbase サーバーの Unicode 関連モードの表示

Essbase サーバーの Unicode 関連モードは、Essbase サーバーに Unicode モードのアプリケーションを作成するための権限と、アプリケーションを Unicode モードに移行するための権限があるかどうかを示します。

管理サービス・コンソールでは、Unicode 関連モードはサーバー・プロパティとして表示されます。

MaxL シェルでは、Unicode 関連モードは、次の値を持つ構成として表示されます:

- 2 (非 Unicode モードのアプリケーションの場合)
- 3 (Unicode モードのアプリケーションの場合)

➤ Essbase サーバーが Unicode モードに設定されているかどうかを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Unicode モードのアプリケーションを作成するための Essbase サーバーの権限の管理	Oracle Essbase Administration Services Online Help
MaxL	**display system**	『Oracle Essbase テクニカル・リファレンス』

Unicode モードのアプリケーションの管理

このセクションのトピックでは、Unicode モードのアプリケーションを管理するためのプロセスについて説明します。

Unicode モードのアプリケーションの作成

アプリケーションを作成する場合は、アプリケーションを Unicode モード対応にするよう指定できます。

➤ Unicode モード対応のアプリケーションを作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Unicode モードのアプリケーションの作成	Oracle Essbase Administration Services Online Help
MaxL	**create application**	『Oracle Essbase テクニカル・リファレンス』

アプリケーションの Unicode モードへの移行

Essbase サーバーで非 Unicode モードのアプリケーションを Unicode モードに移行する場合は、アプリケーション・ファイル内の文字のエンコード方式が UTF-8 エンコード方式に変換されます。

アプリケーション・ファイルを Unicode モードに移行する場合は、次のタスクを実行します:

- アプリケーション内のすべてのアプリケーションおよびデータベース・ファイルをバックアップします。

『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

- 必要に応じて、アウトライン変更ファイルを適用してから、それらのファイルを削除します。

  アウトライン変更ファイルは、存在する場合は各データベース・ディレクトリ内にあり、ファイル名はデータベース名と同じで、拡張子は.chg です。たとえば、/sample/basic/basic.chg のようになります。

- ログ・ファイル内で Unicode エンコード方式と非 Unicode エンコード方式が混在しないように、ログ・ファイルをバックアップしてから、ログ・ファイルを消去または移動します。

  ログ・ファイルは、.log、.xcp、.olg のいずれかの拡張子で終わります。

- アプリケーションを Unicode モードに移行するための Essbase サーバーの権限を付与します。728 ページの「Essbase サーバーの Unicode モードへの設定」を参照してください。

➤ アプリケーションを非 Unicode モードから Unicode モードに移行するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アプリケーションの Unicode モードへの移行	Oracle Essbase Administration Services Online Help
MaxL	**alter application**	『Oracle Essbase テクニカル・リファレンス』

計算スクリプト、レポート・スクリプト、データ・ソースなどのテキスト・ファイルは、UTF-8 エンコード方式に変換されません。Unicode モードのアプリケーションの場合、テキスト・ファイルは、UTF-8 または非 Unicode ロケールでエンコードできます。Essbase Unicode ファイル・ユーティリティを使用すると、非 Unicode でエンコードされたファイルを UTF-8 に変換できます。739 ページの「Essbase Unicode ファイル・ユーティリティ」を参照してください。

注： Unicode モードのアプリケーションで非 Unicode テキスト・ファイルを処理することを選択する場合は、非 Unicode テキスト・ファイルのロケールが Essbase によってどのように決定されるかを理解しているようにしてください。732 ページの「ファイルのエンコード方式の管理」を参照してください。

Unicode アプリケーションと非 Unicode アプリケーションの間のデータベースのバックアップおよび復元

Unicode アプリケーションと非 Unicode アプリケーションを使用している場合、Essbase では、Unicode アプリケーションからバックアップされたデータベースの非 Unicode アプリケーションへの復元は許可されていません。

Unicode アプリケーションと非 Unicode アプリケーションの間での、バックアップされたデータベースの復元がサポートされている組合せのリストについては、『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

アプリケーションの Unicode 関連モードの表示

Administration Services では、アプリケーションの Unicode 関連モードはアプリケーション・プロパティとして表示されます。MaxL では、次の値を持つアプリケーション・タイプとして表示されます:

- 2 (非 Unicode モードのアプリケーションの場合)
- 3 (Unicode モードのアプリケーションの場合)

➤ アプリケーションが Unicode モードのアプリケーションであるかどうかを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**アプリケーションの監視**	Oracle Essbase Administration Services Online Help
MaxL	**display application**	『Oracle Essbase テクニカル・リファレンス』

レポート・スクリプトでの制御文字の使用

非 Unicode モードのアプリケーションでレポートを処理する場合、レポート・ライターは、縦欄式データを正確に整列できるようにするための言語固有のコードを使用します。Unicode のエンコード方式には言語固有のフォーマット・コードは含まれていないため、レポートの列が正確に整列されない可能性があります。

1 バイトのコード・ページを使用する非 Unicode モードのアプリケーションでは、レポート・スクリプトで、次の 16 進制御文字がサポートされています:

```
x20, x09, x0A, x0D, xB3, xC3, xC4, xC2, xC0, x0C, xB1
```

マルチバイト・コード・ページを使用する Unicode モードのアプリケーションおよび非 Unicode モードのアプリケーションでは、レポート・スクリプトで、次の 16 進制御文字がサポートされています:

```
x20, x09, x0A, x0C, x0D
```

パーティションでの処理

パーティションを設計し処理する場合は、次の状況について検討します:

- パーティションでは、非 Unicode モードのデータベースを Unicode モードのデータベースに接続できません。
- リンク・パーティションではない透過パーティションおよび複製パーティションの場合は、パーティションのソースとターゲットとで、アプリケーションのモード(Unicode モードまたは非 Unicode モード)を揃えておく必要があります。
- 同じパーティションのすべてのデータベースは、同じエンコード方式である必要があります。
- Unicode モードのデータベースにまたがるパーティションは、Unicode モードのクライアント、たとえば、Administration Services や、管理サービス・コンソールで実行された MaxL スクリプトで管理できます。
- マルチバイト文字を含む非 Unicode モードのアウトラインは、Administration Services や MaxL スクリプトなどの Unicode モードのクライアントでは同期できません。ただし、MaxL シェル(essmsh)などの、Unicode に対応していないクライアントまたは非 Unicode モードのクライアントを使用できます。

ログの処理

デフォルトでは、Essbase のエージェント・ログ・ファイルは、Essbase サーバーで定義された ESSLANG 変数に指定されているロケールでエンコードされます。UNICODEAGENTLOG 構成設定を使用すると、エージェント・ログ・ファイルを UTF-8 エンコード方式で書き込めます。UTF-8 フォントを使用すると、ログ・ファイル内のすべての文字が読取り可能になります。『Oracle Essbase テクニカル・リファレンス』を参照してください。

Unicode モードのアプリケーションの場合、データベースのアプリケーション・ログ・ファイルおよびアウトライン変更ログ・ファイルは、UTF-8 でエンコードされています。表示するには:

- アプリケーション・ログの場合は、Administration Services のログ・ビューアまたはログ・アナライザを使用するか、または UTF-8 エディタを使用します
- アウトライン変更ログの場合は、UTF-8 エディタを使用します

Unicode モードのアプリケーションに関連するログ・ファイルのエンコード方式は、非 Unicode エンコード方式に変更できません。

ファイルのエンコード方式の管理

Essbase は、多くの非 Unicode エンコード方式をサポートしています(737 ページの「サポートされているロケール」を参照)。さらに、Unicode モードのアプリケーションでは UTF-8 エンコード方式がサポートされています。

ビットの組合せが同じでもエンコード方式が異なれば異なる文字にマッピングされることがあるため、特に非 Unicode でエンコードされたファイルを Unicode モードのアプリケーションで使用する場合など、Essbase でファイルのエンコードをどのように扱うのかを理解しておく必要があります。

注： 複数のロケールにわたって共有される Unicode モードのアプリケーションには、UTF-8 でエンコードされたファイルを使用することをお薦めします。ロケールやエンコード方式を追跡する必要がないため、UTF-8 エンコード方式を使用する方が簡単です。また、Essbase 管理サーバーが内部的に Unicode エンコード方式を使用しており、フォーマット間の内部の変換が必要なくなるため効率も向上します。

以降のトピックでは、Essbase によるファイルのエンコード方式の決定方法、およびエンコード方式が異なるファイルの管理方法について説明します。

ファイルのエンコード方式の決定方法

非 Unicode モードのアプリケーションの場合、Essbase および Administration Services は、文字テキストが Essbase サーバーで定義された ESSLANG 値に指定されているロケールでエンコードされていることを前提にします。

Essbase は、Unicode モードのアプリケーションを処理する場合、文字テキストを内部的に UTF-8 エンコード方式でエンコードし、UTF-8 でその文字テキストを保管します。また、エクスポート・ファイルも UTF-8 でエンコードされます。Essbase はまた、Unicode モードのアプリケーションを処理する場合、非 Unicode でエンコードされた入力ファイル(スクリプト・ファイル、ルール・ファイル、データ・ソースなど)も扱い、内部的に UTF-8 に変換します。

注： Unicode モードのアプリケーションの場合、Essbase では、`dbname.cfg` クエリー・ログ設定ファイルが UTF-8 でエンコードされていることが必要です。

Essbase および Administration Services では、ファイルのエンコード方式のインディケータ(UTF-8 シグネチャまたはロケール・インディケータ)を使用して、ファイルが UTF-8 またはサポートされている非 Unicode エンコード方式のどちらでエンコードされているかを認識します。

ロケール・インディケータは、エンコード方式が Essbase サーバーのロケールに一致する非 Unicode 入力ファイルではオプションです。ただし、エンコード方式が一致しない場合は、ロケール・インディケータを指定する必要があります。734 ページの「エンコード方式のインディケータ」を参照してください。

UTF-8 でエンコードされているテキスト・ファイルには、UTF-8 シグネチャが必要です。

Administration Services では、次のプロセスを使用して、非 Unicode でエンコードされたファイルのエンコード方式を決定します:

1. ファイル内にロケール・インディケータが存在する場合、Administration Services では指定されたエンコード方式を使用します。
2. ロケール・インディケータが存在せず、ファイルがアプリケーション内に配置されている場合、Administration Services では Essbase サーバーのロケールに指定されているエンコード方式を使用します。

3. ロケール・インディケータが存在せず、ファイルがアプリケーション内に配置されていない場合、Administration Services ではファイルのタイプに基づいてエンコード方式を決定します:
    - テキスト・ファイル: Administration Services では、エンコード方式を入力するよう求めます。
    - アウトライン・ファイルおよびルール・ファイル: Administration Services では、Essbase サーバーのロケールに指定されているエンコード方式を使用します。

Essbase で次元構築またはデータ・ロードを実行する場合は、ルール・ファイルとデータ・ファイルのエンコード方式が異なっていても問題はありません。たとえば、ルール・ファイル内のテキストが UTF-8 でエンコードされており、データ・ソースが非 Unicode のコンピュータ・ロケールでエンコードされていても問題はありません。

注： 管理サービス・コンソールを使用してスクリプト・ファイルまたはデータ・ソースを作成する場合は、適切なエンコード方式のインディケータがファイル内に自動的に含まれます。他のいずれかのツールを使用して、Unicode でエンコードされたテキスト・ファイルを作成する場合は、UTF-8 シグネチャが含まれていることを確認する必要があります。非 Unicode でエンコードされたテキスト・ファイルで、そのエンコード方式が Essbase サーバーのロケールと異なる場合はロケール・インディケータが必要です。

次の Essbase のシステム・テキスト・ファイルは、Essbase サーバーで定義された ESSLANG 値に指定されているロケールでエンコードされます:

- `essbase.cfg`
- ESSCMD スクリプト

エンコード方式のインディケータ

テキスト(メンバー名など)を正しく解釈するには、Essbase で、そのテキストのエンコード方式を認識している必要があります。多くのファイルにはエンコード方式のインディケータが含まれていますが、場合によっては、エンコード方式を指定するよう求められることがあります。たとえば、次のような場合があります:

- Administration Services がファイルを作成し、それを Essbase サーバーとは別の場所に保管する場合
- Administration Services が、非 Unicode であった、7.0 より前のリリースの Essbase で作成されたファイルを読み取る場合

エンコード方式のインディケータのタイプは、ファイルのタイプによって異なります:

- アプリケーションおよびデータベースで内部的に使用され、ユーザーが直接編集できないファイルは、一般にバイナリ・ファイルであり、エンコード方式のインディケータは含まれていません。

これらのファイル内の文字テキストは、アプリケーションのモードに基づいてエンコードされています:

- Unicode モードのアプリケーション・ファイルに含まれているテキストは、UTF-8 でエンコードされています。
- 非 Unicode モードのアプリケーション・ファイルに含まれているテキストは、そのアプリケーションが作成された Essbase サーバーの ESSLANG 変数に指定されているロケールでエンコードされています。

- ユーザーが編集できるバイナリ・ファイル(アウトライン・ファイルやルール・ファイルなど)

  必要に応じて、Essbase では、文字テキストが UTF-8 でエンコードされているかどうかを内部的に追跡します。UTF-8 エンコード方式でない場合、Essbase は内部のロケール・インディケータを使用して、文字テキストのエンコード方式に使用されているロケールを識別します。

- 次の編集可能なテキスト・ファイルでは、UTF-8 シグネチャまたはロケール・インディケータを使用してエンコード方式が示されます:
  - 計算スクリプト
  - レポート・スクリプト
  - MaxL スクリプト
  - 次元構築およびデータ・ロードに使用するデータ・ソース
  - 別名テーブルのインポート・ファイル(Administration Services では、別名テーブルのインポート・ファイルが UTF-8 でエンコードされていることが必要です)

注意 7.0 より前のリリースの Essbase サーバー・インストール(Unicode に対応していません)で作成されたロケール・インディケータを含む、非 Unicode でエンコードされたファイルを使用しないでください。これらのロケール・インディケータを削除するには、Essbase Unicode ファイル・ユーティリティを使用します。739 ページの「Essbase Unicode ファイル・ユーティリティ」を参照してください。

UTF-8 シグネチャ

UTF-8 でエンコードされたテキスト・ファイルには、テキスト・ファイルの先頭のマークである UTF-8 シグネチャが含まれている必要があります。このシグネチャは、一部のサードパーティ製 UTF-8 テキスト・エディタで表示されます。

UTF-8 シグネチャは、多くの UTF-8 テキスト・エディタで作成できます。また、Essbase Unicode ファイル・ユーティリティを使用して、ファイルに UTF-8 シグネチャを挿入もできます(739 ページの「Essbase Unicode ファイル・ユーティリティ」を参照)。

管理サービス・コンソールを使用してファイルを作成した場合は、UTF-8 シグネチャがファイルに自動的に挿入されます。

ロケール・インディケータ(ロケール・ヘッダー・レコード)

ロケール・インディケータは、非 Unicode テキスト・ファイルのエンコード方式を識別するための追加のテキスト・レコードであるロケール・ヘッダー・レコードです。非 Unicode でエンコードされたテキスト・ファイルの作成時に、ファイル内の最初のレコードとしてロケール・ヘッダー・レコードを挿入するか、Essbase Unicode ファイル・ユーティリティを使用してヘッダーを追加できます。

注意 UTF-8 でエンコードされたファイル内にロケール・ヘッダー・レコードを挿入しないでください。テキスト・ファイルに両方のタイプのエンコード方式のインディケータが含まれていると、そのロケール・ヘッダーは最初のデータ・レコードとして読み取られます。

ロケール・ヘッダー・レコードのフォーマットは次のとおりです:

```
//ESS_LOCALE
locale-name
```

`locale-name` は、サポートされている Global C ロケールで、そのフォーマットは ESSLANG 変数と同じです:

```
language
_
territory
.
code_page_name
@
sortsequence
```

次の例は、あるロシア語コード・ページのロケール・ヘッダー・レコードを示しています:

```
//ESS_LOCALE Russian_Russia.ISO-8859-5@Default
```

注: Essbase は、レコードの `code_page_name` の部分のみを調べます。`sortsequence` の指定は、レポート・スクリプト内のソート順には影響を与えません。

また、次の規則も適用されます:

- `locale-name` の後には、空白、タブまたは<end of line>を使用してヘッダーを終了します。
- 空白またはタブは、次の位置に使用できます:
  - キーワード"`//ESS_LOCALE`"の前
  - "`//ESS_LOCALE`"と `locale-name` の間

- ロケール指定の後

互換性のために、管理サービス・コンソールは、7.0 より前のリリースの Essbase サーバー・インストール上の計算スクリプトを異なるフォーマットで保存します。ロケールの前に//を付けるのではなく、管理サービス・コンソールでは、ロケール・ヘッダーを計算スクリプトのコメント・マークの間に挿入します:

```
/* */
```

サポートされているロケール

次のサポートされているロケールを、ロケール・ヘッダー・レコードおよび Essbase Unicode ファイル・ユーティリティで、または ESSLANG 変数の値として使用できます:

```
   Arabic_SaudiArabia.ISO-8859-6@Default
Arabic_SaudiArabia.MS1256@Default
Croatian_Croatia.ISO-8859-2@Croatian
Croatian_Croatia.MS1250@Croatian
CyrillicSerbian_Yugoslavia.ISO-8859-5@Default
CyrillicSerbian_Yugoslavia.MS1251@Default
Czech_CzechRepublic.ISO-8859-2@Czech
Czech_CzechRepublic.MS1250@Czech
Danish_Denmark.ISO-8859-15@Danish
Danish_Denmark.IBM500@Danish
Danish_Denmark.Latin1@Danish
Danish_Denmark.MS1252@Danish
Dutch_Netherlands.IBM037@Default
Dutch_Netherlands.IBM500@Default
Dutch_Netherlands.ISO-8859-15@Default
Dutch_Netherlands.Latin1@Default
Dutch_Netherlands.MS1252@Default
English_UnitedStates.IBM037@Binary
English_UnitedStates.IBM285@Binary
English_UnitedStates.IBM500@Binary
English_UnitedStates.MS1252@Binary
English_UnitedStates.Latin1@Binary
English_UnitedStates.US-ASCII@Binary
Finnish_Finland.IBM500@Finnish
Finnish_Finland.ISO-8859-15@Finnish
Finnish_Finland.Latin1@Finnish
Finnish_Finland.MS1252@Finnish
French_France.IBM297@Default
French_France.IBM500@Default
French_France.ISO-8859-15@Default
French_France.Latin1@Default
French_France.MS1252@Default
German_Germany.IBM273@Default
German_Germany.IBM500@Default
German_Germany.ISO-8859-15@Default
German_Germany.Latin1@Default
German_Germany.MS1252@Default
Greek_Greece.ISO-8859-7@Default
```

```
Greek_Greece.MS1253@Default
Hebrew_Israel.ISO-8859-8@Default
Hebrew_Israel.MS1255@Default
Hungarian_Hungary.ISO-8859-2@Hungarian
Hungarian_Hungary.MS1250@Hungarian
Italian_Italy.IBM280@Default
Italian_Italy.IBM500@Default
Italian_Italy.ISO-8859-15@Default
Italian_Italy.Latin1@Default
Italian_Italy.MS1252@Default
Japanese_Japan.IBM930@Binary
Japanese_Japan.JapanEUC@Binary
Japanese_Japan.MS932@Binary
Korean_Korea.MS1361@Binary
Korean_Korea.MS949@Binary
Norwegian_Norway.IBM500@Danish
Norwegian_Norway.ISO-8859-10@Danish
Norwegian_Norway.ISO-8859-15@Danish
Norwegian_Norway.ISO-8859-4@Danish
Norwegian_Norway.Latin1@Danish
Norwegian_Norway.MS1252.Default
Portuguese_Portugal.IBM037@Default
Portuguese_Portugal.IBM500@Default
Portuguese_Portugal.ISO-8859-15@Default
Portuguese_Portugal.Latin1@Default
Portuguese_Portugal.MS1252@Default
Romanian_Romania.ISO-8859-2@Romanian
Romanian_Romania.MS1250@Romanian
Russian_Russia.ISO-8859-5@Default
Russian_Russia.MS1251@Default
Serbian_Yugoslavia.ISO-8859-2@Default
Serbian_Yugoslavia.MS1250@Default
SimplifiedChinese_China.IBM935@Binary
SimplifiedChinese_China.MS936@Binary
SimplifiedChinese_China.UTF-8@Binary
Slovak_Slovakia.ISO-8859-2@Slovak
Slovak_Slovakia.MS1250@Slovak
Slovenian_Slovenia.ISO-8859-10@Slovenian
Slovenian_Slovenia.ISO-8859-2@Slovenian
Slovenian_Slovenia.ISO-8859-4@Slovenian
Slovenian_Slovenia.MS1250@Slovenian
Spanish_Spain.IBM500@Spanish
Spanish_Spain.ISO-8859-15@Spanish
Spanish_Spain.Latin1@Spanish
Spanish_Spain.MS1252@Spanish
Swedish_Sweden.IBM500@Swedish
Swedish_Sweden.ISO-8859-15@Swedish
Swedish_Sweden.Latin1@Swedish
Swedish_Sweden.MS1252@Swedish
Thai_Thailand.MS874@Thai
TraditionalChinese_Taiwan.EUC-TW@Binary
TraditionalChinese_Taiwan.IBM937@Binary
TraditionalChinese_Taiwan.MS950@Binary
Turkish_Turkey.ISO-8859-3@Turkish
Turkish_Turkey.ISO-8859-9@Turkish
Turkish_Turkey.MS1254@Turkish
Ukrainian_Ukraine.ISO-8859-5@Ukrainian
```

Ukrainian_Ukraine.MS1251@Ukrainian

Essbase Unicode ファイル・ユーティリティ

Essbase Unicode ファイル・ユーティリティを使用すると、非 Unicode でエンコードされたテキスト・ファイルを UTF-8 エンコード方式に変換したり、エンコード方式のインディケータを追加または削除したりできます。このプログラムは、次の操作をサポートしています:

- 非 Unicode でエンコードされたテキスト・ファイルを UTF-8 エンコード方式 (UTF-8 シグネチャを含む)に変換します。
- UTF-8 でエンコードされたテキスト・ファイルに UTF-8 シグネチャを追加します。
- 非 Unicode でエンコードされたテキスト・ファイルの先頭にロケール・ヘッダー・レコードを挿入します。
- アウトライン・ファイル(.otl)またはルール・ファイル(.rul)にロケール・インディケータを追加します。
- ロケール・インディケータをファイルから削除します(このユーティリティでは UTF-8 シグネチャを削除できません)。

Essbase Unicode ファイル・ユーティリティは、ESSBASEPATH/bin ディレクトリに保管され、essutf8.exe (Windows)または ESSUTF8 (UNIX)という名前が付いています。このユーティリティ・プログラムは、次のファイルとともに使用できます:

- 計算スクリプト
- レポート・スクリプト
- MaxL スクリプト
- 次元構築およびデータ・ロードに使用するテキストのデータ・ソース
- 別名テーブルのインポート・ファイル
- アウトライン・ファイル
- ルール・ファイル

このユーティリティおよびコマンド構文の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

第 VIII 部
Essbaseの管理

Essbase の管理の内容：

45 Essbaseの管理(OPMNを使用)

この章の内容

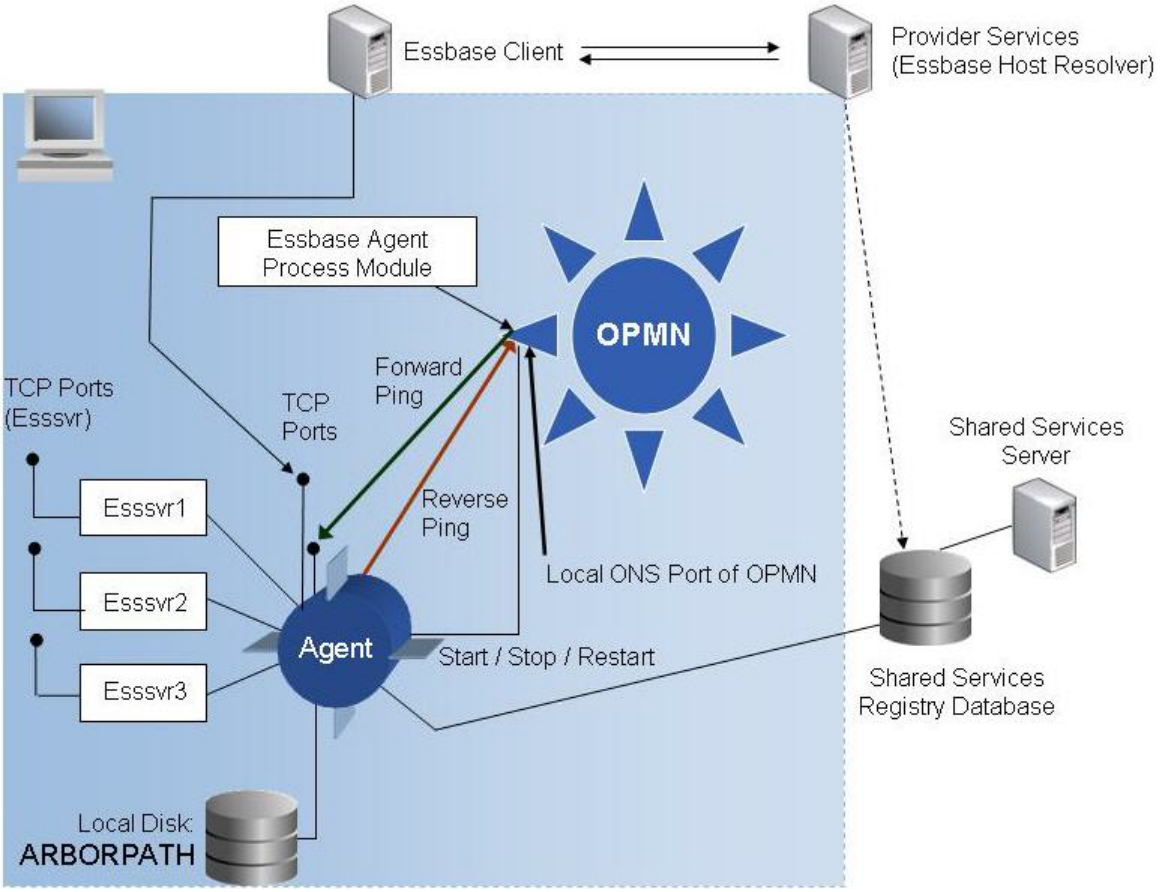
OPMN について

Oracle Process Manager and Notification (OPMN)サーバーを使用して、Essbase エージェント・プロセスを監視および制御できます。Essbase エージェント情報を opmn.xml ファイルに追加して、OPMN コマンドライン・インタフェースを使用することにより、Oracle Process Manager and Notification Server で Essbase エージェントを起動、停止および再起動できるようにします。OPMN は、Essbase エージェントが応答しなくなった場合、予期せずに終了した場合、または使用不可になったことが ping および通知操作によって確認された場合、Essbase エージェントを自動的に再起動できます。OPMN は EPM_HOME にインストールされます。

図 148 には、Essbase を OPMN と統合する方法が記載されています。

また、OPMN に用意されているフェイルオーバー機能を使用して、Essbase クラスタの高可用性を実現できます(747 ページの「Essbase フェイルオーバー・クラスタの理解」を参照)。

図 148　Essbase と OPMN の統合



Oracle Process Manager Modules (PM Modules)は、プロセス管理および状態監視を目的として Essbase エージェントと通信します。Essbase エージェントと OPMN は、管理対象コンポーネントとして、コンポーネント間の通知用転送メカニズムである Oracle Notification Server (ONS)を使用して通信します。

Essbase を起動(OPMN コマンドライン・インタフェースを使用)すると(745 ページの「Essbase の起動および停止(OPMN を使用)」を参照)、OPMN は、エージェント・プロセスを起動し、ステータスを初期化済に設定します(`opmn.log` ファイルにメッセージが記録されます)。この状態が活動状態に切り替わるかどうかは、OPMN がエージェントから応答(reverse ping)を受信する(つまり、エージェントがアクティブである)かどうかによって決まります。

Essbase が起動して稼働すると、OPMN は TCP ベースの forward ping をエージェントに送信します。ping 試行が失敗すると、OPMN はエージェントとのネゴシエーションを最大で 3 回試行します。すべての ping 試行が失敗すると、OPMN はエージェントを停止します。OPMN は、次のシナリオに対してエージェントの再起動を試行します:

- `restart-on-death` (`opmn.xml`)が TRUE に設定されている場合。

  本番環境では、`restart-on-death` は常に TRUE に設定しておく必要があります。フェイルオーバーの試行前に、まずローカル・ノードでプロセスを再起動することをお薦めします。`restart-on-death` が TRUE に設定されていると、OPMN はまず、`opmn.xml` に構成されている再起動の回数(最初の起動と再起動の回数)だけ、ローカル・ノードの Essbase の再起動を試行します。すべ

ての再起動に失敗すると、スタンバイ・ノードへのフェイルオーバーが行われます。

開発およびテスト環境では、restart-on-death を FALSE に設定してフェイルオーバーをテストできます。

デフォルト・モードである非フェイルオーバー・モードでは、restart-on-death のデフォルト設定は FALSE です。

- フェイルオーバー・モードがオンの場合。これは restart-on-death 値より優先されます。restart-on-death が FALSE で、フェイルオーバー・モードがオンである場合、OPMN は Essbase をアクティブ・ノードまたはパッシブ・ノードで起動する可能性があります。

Essbase を停止(OPMN コマンドライン・インタフェースを使用)すると、OPMN はエージェントをシャットダウンすることにより、エージェントが保持しているすべてのリソースを解放します。Essbase サーバーは、親エージェント・プロセスがシャットダウンされていることを検出すると、自動的にシャットダウンされます。

Essbase の起動および停止(OPMN を使用)

OPMN によって管理されるコンポーネントは、手動で起動または停止しないようにする必要があります。システム・コンポーネントを起動および停止するには、opmnctl コマンドライン・ユーティリティを使用してください。

注： Essbase インスタンス(OPMN を使用して起動された)を MaxL を使用してシャットダウンしようとすると、OPMN を使用して Essbase をシャットダウンするよう警告されます。

Windows 環境では、Essbase を起動する前に、指定ユーザーで起動されるように OPMN サービス(Oracle Process Manager (instanceName))を変更し、共有ネットワーク・ファイルにアクセスできるようにします。

Essbase を OPMN から起動、停止および監視するには、次のコマンドを使用してください:

- `opmnctl status`

  システム・コンポーネント・プロセスのステータスを確認できます。

- `opmnctl startproc ias-component= EssbaseInstanceName`

  EssbaseInstanceName という名前のシステム・コンポーネントを起動します。

- `opmnctl restartproc ias-component= EssbaseInstanceName`

  EssbaseInstanceName という名前のシステム・コンポーネントを再起動します。

- `opmnctl stopproc ias-component= EssbaseInstanceName`

  EssbaseInstanceName という名前のシステム・コンポーネントを停止します。

ここで、EssbaseInstanceName は、次のいずれかです:

- フェイルオーバー・クラスタを実装しなかった場合は、`EssbaseInstanceName` は、Essbase サーバーの構成時に入力した Essbase インスタンスの名前です。
- フェイルオーバー・クラスタを実装した場合は、`EssbaseInstanceName` は、Essbase クラスタの設定時に入力した Essbase クラスタの名前です。

Essbase サーバーの開始および停止スクリプトは OPMN にリダイレクトされます。Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

OPMN コマンドの詳細は、『Oracle Process Manager and Notification Server 管理者ガイド』を参照してください。

トラブルシューティングおよびロギング情報は、Oracle Enterprise Performance Management System Installation and Configuration Troubleshooting Guide を参照してください。

論理名を使用した Essbase へのログイン

Essbase クライアントは、URL を使用して、次の形式で Essbase クラスタに接続できます:

```
http
(s)
://host:port/aps/Essbase?ClusterName=
clusterName
&SecureMode=yesORno
```

ログインを簡略化するために、Essbase クライアントは、URL ではなく論理クラスタ名を`<name>:<secure>`という形式で直接使用する必要があります。クラスタ名を使用したクライアント・ログインを可能にする場合は、まず最初に Provider Services を構成するためのプロパティを指定してください。クラスタ名は、構成ファイルに指定されている Provider Services サーバーによって解決されます。

- `essbase.cfg` の APSRESOLVER – パーティション、XRef、および XWrite に対して Essbase サーバーを定義する場合などのサーバー/サーバー間通信

  Oracle Essbase Technical Reference を参照してください。
- `essbase.properties` の `aps.resolver.urls` – クライアント/サーバー間通信。Smart View から Essbase に接続する場合など

  Oracle Hyperion Provider Services Administration Guide を参照してください。

サーバー名を解決する際に、Login API は次の優先順位に従います:

1. 内部キャッシュ内のサーバー名(API レイヤーに保持)を検索します。
2. キャッシュ内にサーバー名が見つからない場合は、`essbase.cfg` にリストされている個々の Provider Services サーバーにコンタクトして、サーバー名の解決を試行します。

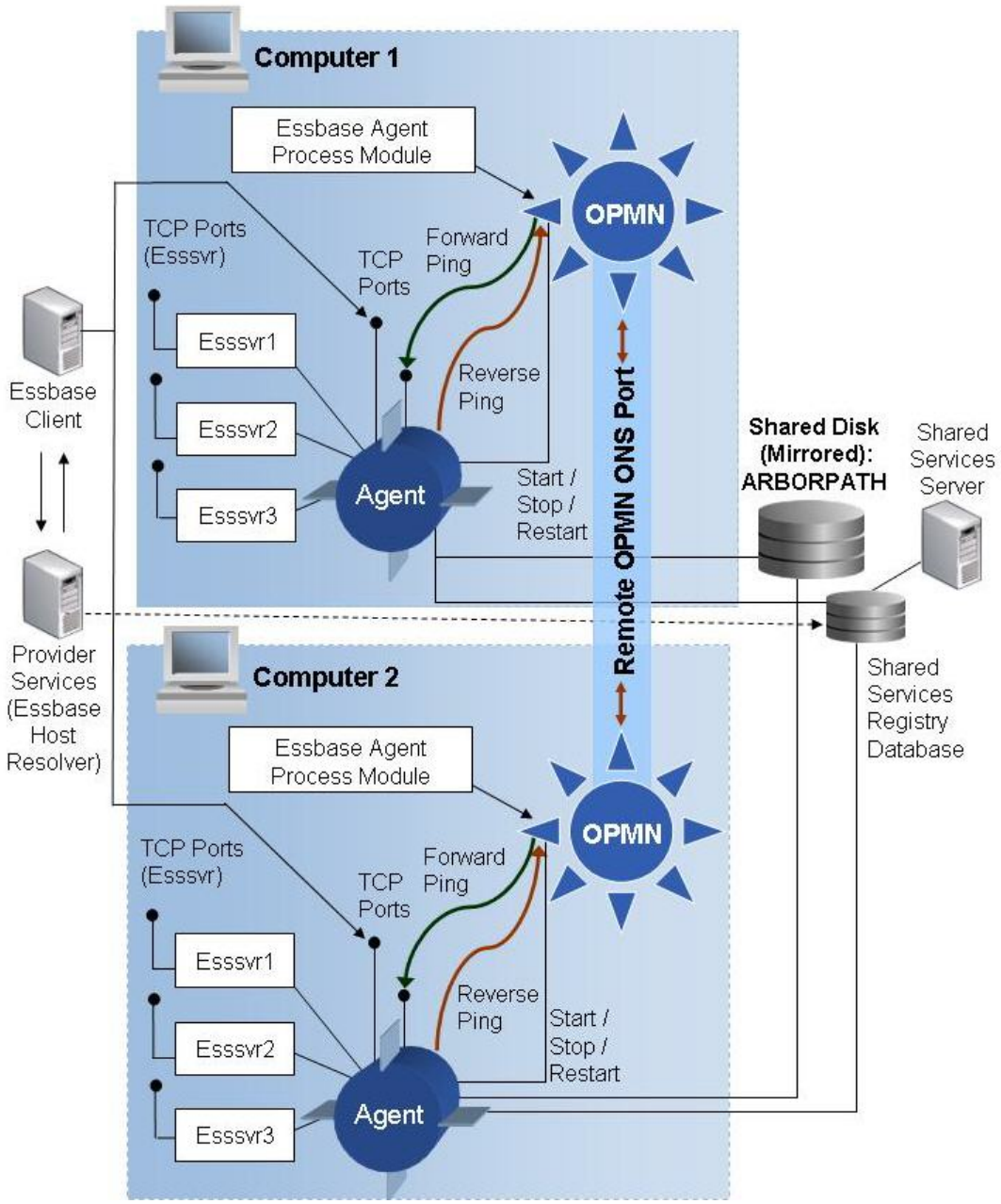
3. サーバー名が見つからない場合は、サーバー名を物理サーバー名として処理します。

Essbase フェイルオーバー・クラスタの理解

Essbase アクティブ-パッシブ・クラスタは、2 つの Essbase インスタンスで構成されます。2 つのインスタンスはアクティブ・ノードとパッシブ・ノードに 1 つずつ置かれ、これらのノードは構成用とデータ用の共通ストレージを共有します。ストレージは 2 台のコンピュータ間で共有されるため(たとえば、SAN を使用)、管理者がストレージを同期する必要がなく、(Provider Services での Essbase アクティブ-アクティブ・クラスタの場合のような)読取り専用サポートの制約もありません。Essbase は、書込み時のデータ破損を回避するため、データベース・テーブルを使用して 1 つのエージェントおよびその関連サーバーのみをアクティブにします。インストールおよび構成時に、クラスタ内に存在する構成データおよびアプリケーション・データに関する情報を保持するテーブルが作成されます。Essbase アクティブ-パッシブ・クラスタでは、ファイル・システムのロック(`essbase.lck` ファイル)は使用されません。

OPMN で使用可能なサービス・フェイルオーバー機能を使用して Essbase サーバーをクラスタリングすることにより、ライトバック機能および API サポートとともにアクティブ-パッシブ・サービス・フェイルオーバーを提供します。Essbase サーバーをクラスタリングすることにより、アクティブ・サーバーに障害が発生したときにパッシブ・サーバーに自動的に切り替える機能を提供し、Essbase エージェントの高可用性を実現します。

図 149　Essbase フェイルオーバー・クラスタのアーキテクチャ



Essbase フェイルオーバー・クラスタおよびリース管理

サブトピック

- リースを使用する理由
- リースの動作
- リースの実装方法
- ロギング

この項のトピックでは、リースの概念について説明します。

リースを使用する理由

Essbase フェイルオーバーでは、リース・メカニズムを使用してスプリット・ブレイン問題を解決します。スプリット・ブレイン状態になるのは、アクティブ-パッシブ・クラスタ環境内の両方のインスタンスが自身の方がアクティブであると考え、お互いに気付かない場合です。これは、アクティブ-パッシブ・クラスタの基本的な前提に反しています。スプリット・ブレインが発生する状況には、ネットワークの停止やネットワーク・パーティションがあります。

リースの動作

Essbase プロセスは、起動時に特定の共有リソースまたはリソース・セットについてのリースを取得します; プロセスは、リースを定期的に更新し、終了時にリースを破棄します。

このリースにより、プロセスがプライマリ・サービス・プロバイダとして確立され、特定の共有リソース・セットの更新、クライアント要求の処理、およびプロセスの生成を行う権限を付与されます。

Essbase サーバーがリースを取得できるのは、権限を付与するエージェントに現在のリースがある場合のみです。

リースの実装方法

Essbase フェイルオーバーには、一元化されたリレーショナル・データベースを使用して次に関する情報を格納、更新および取得するリース・メカニズムが実装されています:

- 共有リソース
- リース所有権
- リースの妥当性

この目的のために使用されるテーブル・セットは、Essbase のインストールおよび構成時に作成されます。これらのテーブルは、Shared Services レジストリと同じデータベースおよびスキーマに格納されます。

Essbase エージェントおよびサーバーは起動時に、他のプロセスが共有リソースに対するリースを現在所有しているかどうかを確認するためにデータベース・テーブルを検査します。他のプロセスが所有している場合、Essbase エージェントおよびサーバーはリースの有効期限が切れるまで待機してから再試行します。他のプロセスが所有していない場合、Essbase エージェントおよびサーバーは共有リソースの所有権を取得します; つまり、リースを取得します。

Essbase エージェントおよびサーバーは、起動時にリースを取得し、終了時にリースを破棄します。Essbase エージェントおよびサーバーは、リースを定期的に更新し、その状態を示します。リースを更新できない場合、Essbase エージェントおよびサーバーは自動的に終了します。

リース所有権は、共有ディスク内に存在する共有リソース・セットを排他的に更新する権限です。リース所有者(Essbase エージェントまたは Essbase サーバー)は、事前構成された時間内にリースを更新しようとします。Oracle Essbase Technical

Reference の AGENTLEASERENEWALTIME および SERVERLEASERENEWALTIME の構成設定に関する項を参照してください。

ロギング

リース管理に固有の診断メッセージは、リース・マネージャのログに記録されます。Oracle Enterprise Performance Management System Installation and Configuration Troubleshooting Guide を参照してください。

フェイルオーバー検出の調整

フェイルオーバー検出を制御する調整可能なパラメータを、表 127、表 128 および表 129 に記載します。デフォルトでは、これらのパラメータは、フェイルオーバーが 1 分未満以内に実行されるよう設定されています。これらのパラメータは、ビジネス要件に応じて調整する必要があります。

表 127　調整可能なエージェント・パラメータ(essbase.cfg 内)

エージェント・パラメータ	値
エージェント・リースの有効期間	20 秒
エージェント・リースの更新間隔	10 秒
エージェント・リースの取得最大試行	5

表 128　調整可能なサーバー・パラメータ(essbase.cfg 内)

サーバー・パラメータ	値
サーバー・リースの有効期間	20 秒
サーバー・リースの更新間隔	10 秒
サーバー・リースの取得最大試行	5

リース取得の各試行失敗から最大試行までの遅延時間は、リースの有効期間が終了するまでに残された時間に基づいて計算されます。リースが使用可能なときに取得が失敗した場合、遅延は 1 秒になる場合があります。Oracle Essbase Technical Reference を参照してください。

表 129　調整可能な OPMN パラメータ(opmn.xml 内)

OPMN パラメータ	デフォルト値
restart-on-death	FALSE。FAILOVERMODE(essbase.cfg 内)が TRUE に設定されている場合、この値を TRUE に変更し、Essbase エージェントが異常終了した際に OPMN に再起動されるようにする必要があります。
shutdown-force-enabled	FALSE。Essbase フェイルオーバー・クラスタを使用可能にした場合は、これを TRUE に設定する必要があります。 これを TRUE に設定すると、Essbase が強制終了されるのは、OPMN による段階的なシャットダウンが失敗したときです。

OPMN パラメータ	デフォルト値
OPMN Forward Ping 間隔	20 秒
OPMN PROC_READY (reverse ping)間隔	20 秒
エージェント起動または再起動試行	2 OPMN が、このサービスに対してこのインスタンスが機能していないと判断するまでに、エージェントの起動または再起動の失敗を 2 回許容することを意味します。デフォルトの回数以内の失敗では、エージェント・サービスのフェイルオーバーがフェイルオーバー・モードで開始されます; デフォルトの回数を超えて失敗した場合は、エラーがスローされます。
OPMN エージェント起動タイムアウト	10 分
OPMN エージェント再起動タイムアウト	10 分
OPMN エージェント停止タイムアウト	10 分

『Oracle Process Manager and Notification Server 管理者ガイド』を参照してください。

エージェントおよびサーバーの終了条件

表 130 は、Essbase エージェントおよび Essbase サーバーが終了する原因となるエラー条件、影響を受けるコンポーネント、OPMN の応答、および Essbase クライアントの応答をリストしています。

表 130　Essbase エージェントおよびサーバーの終了シナリオ

エラー条件	稼働していることが前提のもの	フェイルオーバー機能	必要とされる品質保証契約内容
Essbase サーバーの停止: ● ソフトウェアの不具合 ● リースの有効期限切れ ● 異常なエラー条件	Essbase エージェント ネットワーク 共有ディスク	OPMN は影響を受けません。 Essbase エージェントは、新規要求を取得すると Essbase サーバーを再起動します; リースが使用可能になるまで待機している間にサーバーの起動が若干遅れる可能性があります。	クライアントは、「要求の失敗」エラーを受信します(要求がサーバーに向けられている間)。 クライアントは、要求を再送信する必要があります。

エラー条件	稼働していることが前提のもの	フェイルオーバー機能	必要とされる品質保証契約内容
Essbase エージェントの停止: ● ソフトウェアの不具合 ● リースの有効期限切れ ● 異常なエラー条件	ネットワーク 共有ディスク	Essbase エージェントに対する OPMN の ping が失敗します。 OPMN は、同じノードでの Essbase エージェントの再起動を試行します。 Essbase エージェントが同じノードで再起動しない場合、OPMN はパッシブ・ノードでフェイルオーバーを開始します。 Essbase エージェントは、アプリケーション要求を取得するとサーバーを再起動します。	クライアントが SessionID を使用して Essbase エージェントへのネゴシエーションの試行中にネットワーク接続解除エラーを受信します。 サーバーは孤立し、次のイベントが発生します: 1. サーバーはリースの更新を停止し、終了します。 2. サーバーはパイプの破損を検出し、終了します。 (1)または(2)のどちらかが順不同で発生し、サーバーが終了します。 クライアントは Essbase に再ログインする必要があります(エージェントの再起動後)。 クライアントは要求を再送信します。
ネットワークの停止 (つまり、ネットワークパーティション。プライマリ IP [eth0]がアクティブ・ノードで停止)	Essbase エージェント(現在のアクティブ・ノード上で) Essbase サーバー(現在のアクティブ・ノード上で) 共有ディスク リース・データベース(別のネットワーク・インタフェース(eth0 以外)からアクセス可能な場合には稼働している必要あり)	Essbase エージェントに対する OPMN の forward ping が失敗します。 OPMN はローカル・ノードで Essbase エージェントの再起動を試行しますが、ノードが失敗します(eth0 が停止); Essbase エージェントは終了します。 サーバーは、Essbase エージェントの停止を検出し、終了します。 アクティブ・ノードの ONS ピアとパッシブ・ノードの ONS ピアの間のネットワーク・ハートビートが失敗します(つまり、ネットワーク・パーティション)。 パッシブ・ノードの OPMN が新しいアクティブ・ノードになります。 OPMN は、新しいアクティブ・ノードで Essbase エージェントを起動します; Essbase エージェントのリースが使用可能になるのが若干遅れる可能性があります。 Essbase エージェントは、要求を受信すると Essbase サーバーを起動します。	クライアントは、セッションの再使用の試行中にエラーを受信するか、ネットワークの停止が検出されるまでハングします。 Essbase エージェントおよびサーバーはアクセス不可になります。 クライアントは、パッシブ・ノードで起動した後で Essbase エージェントに再ログインする必要があります; その後で、要求を再送信する必要があります。
エージェントの停止後に、`Essbase.lck` ファイルが存在	適用なし	適用なし フェイルオーバー・モードになると、`Essbase.lck` ファイルは削除されます。	適用なし

エラー条件	稼働していることが前提のもの	フェイルオーバー機能	必要とされる品質保証契約内容
Essbase エージェントの停止後、Essbase.sec ファイルが破損 (一意でないシナリオ; Essbase エージェントのクラッシュまたはネットワーク・パーティションに続いて発生する場合あり)	ネットワーク 共有ディスク	Essbase フェイルオーバーには適用なし。 Essbase エージェントは、管理者が適切な essbase.sec をバックアップから復元するまでは起動しません。	管理者なしではサービス使用不可。 Essbase エージェントが再起動した後で、クライアントは再ログインする必要があります。
ディスク停止 (共有ディスクの停止)	ネットワーク	Essbase フェイルオーバーには適用なし。 カスタマは、共有ディスク内のシングル・ポイント障害を解消する必要があります。 これには、ディスクの冗長性を備えた SAN (RAID 1-0 構成)に含まれているような、ミラーリング設定を持った共有ディスクを稼働します。	アクティブ・ノードとパッシブ・ノードの両方に障害が発生します。 サービスは使用不可です。
リース・データベースの停止	ネットワーク 共有ディスク	Essbase エージェントは、リースを更新できず、終了します。 サーバーは、リースを更新できず、終了します。 リース・データベースのシングル・ポイント障害を解消する必要があります。リース・データベース(リレーショナル)をコールド・フェイルオーバー・クラスタ(CFC) (アクティブ-パッシブ)モードまたは RAC モード(アクティブ-アクティブ)で実行することをお薦めします。	サービス使用不可。 アクティブ・ノードとパッシブ・ノードが両方とも Essbase を実行できません
ノード障害(致命的なハードウェア障害)	ネットワーク 共有ディスク	アクティブ・ノードの ONS ピアとパッシブ・ノードの ONS ピアの間のネットワーク・ハートビートが失敗します(現在のアクティブ・ノードがクラッシュしています)。 パッシブ・ノードの OPMN が新しいアクティブ・ノードになります。 OPMN は、新しいアクティブ・ノードで Essbase エージェントを起動します; Essbase エージェントのリースが使用可能になるのが若干遅れる可能性があります。 Essbase エージェントは、要求を受信すると Essbase サーバーを起動します。	クライアントは、セッションの再使用の試行中にエラーを受信します; エージェントおよびサーバーが停止しています。 クライアントは、スタンバイ・ノードで起動した後で Essbase エージェントに再ログインする必要があります; その後で、要求を再送信する必要があります。

エラー条件	稼働していることが前提のもの	フェイルオーバー機能	必要とされる品質保証契約内容
Shared Services Java Web アプリケーションの停止	ネットワーク 共有ディスク Essbase エージェント Essbase サーバー	Essbase フェイルオーバーには適用なし。 LDAP プロバイダ(OpenLDAP、Oracle LDAP/外部ディレクトリ)が実行中の間は、Essbase エージェントはユーザーを認証できます(Shared Services に対するランタイム依存はありません)	特定のユーザー操作が失敗します; たとえば、アプリケーションの作成および削除(Shared Services Web サーバーによる Shared Services レジストリの更新)が失敗します。 既存のクライアントは稼働し続けます。
Essbase エージェントおよびサーバーがハングします(アプリケーションの不具合)	ネットワーク 共有ディスク Essbase エージェント	Essbase エージェントおよびサーバーのハングは明示的には処理されませんが、フェイルオーバー・クラスタの使用時におけるエージェントおよびサーバーの全体的な堅牢性は向上します。	Essbase エージェントおよびサーバーがそのリースを更新できる間は、既存の動作に変更はありません。

Universal/Uniform Naming Convention (UNC)パス

Essbase クラスタを使用してサーバーの高可用性を得る場合、Essbase アプリケーション・ファイルをシステム間で共有するためにネットワーク・ファイル・システムを使用するのが一般的です。

Windows では、Universal/Uniform Naming Convention (UNC)を使用して、ネットワーク・リソースの場所(共有ディレクトリなど)を指定できます。構文:

```
\\ComputerName\SharedFolder\Resource
```

Essbase が Windows 上で稼働している場合、次の Essbase サーバー・アプリケーション・アーティファクトのネットワーク共有パスを指定する際には UNC パスがサポートされます:

- ARBORPATH
- 集約ストレージ・アプリケーションのテーブルスペースのパス
- ブロック・ストレージ・アプリケーションのディスク・ボリュームのパス

その他すべてのファイル・パス(データ・ロードおよび次元構築用のクライアント側のパスなど)については、UNC パスはサポートされていません。

UNC 構文の詳細は、MSDN を参照してください。

Provider Services との SSL 暗号化通信のための構成

Essbase を Provider Services との暗号化(SSL)通信用に構成するには、Essbase `libcurl` ライブラリを有効にして Provider Services へのセキュア・チャネルを設定するための構成タスクを実行する必要があります。これは、論理 Essbase クラスタ名を使用して Essbase と Provider Services の間にセキュア接続を作成するために必要な作業です。

次の構成オプションを使用して、ピア検証を無効にするか、認証局(CA)証明書の場所を指定します。

➤ この構成を実行するには、オペレーティング・システムのコマンドラインを使用して次の環境変数を設定します:

```
API_DISABLE_PEER_VERIFICATION=1
```

この変数を設定するのは、接続が SSL を経由するが証明書を必要としない場合です。Essbase は、データの暗号化は行いますが、認証は行いません。

`API_DISABLE_PEER_VERFICATION` が設定されていないか、0 に設定されている場合、証明書を設定する必要があります。Provider Services とのセキュア接続を確立する際、Essbase は Provider Services サーバーの認証を行うために、CA による Provider Services 証明書への署名を必要とします。証明書を含むファイルまたはディレクトリを指定できます。次のオプションのいずれかを選択します:

```
API_CAINFO=CA
certificate file path
```

または

```
API_CAPATH=
directory path containing CA certificates
```

ここで説明した変数は、`https` で始まる Provider Services URL を使用してログインする場合にのみ適用されます。`http` URL を使用したログインは TCP/IP 経由でリダイレクトされます。

46 Essbaseサーバー、アプリケーションおよびデータベースの実行

この章の内容

Essbase の実行可能ファイル

表 131 は、Essbase サーバーおよびクライアントの実行可能ファイルを示しています:

表 131　Essbase の主要な実行可能ファイル

実行可能ファイル [1]	説明	場所	参照
essbase.exe	Essbase サーバー・エージェント・プロセス	ESSBASEPATH /bin	758 ページの「エージェントの理解」
esssvr.exe	アプリケーション・プロセス	ESSBASEPATH /bin	763 ページの「アプリケーションの開始と停止」
essmsh.exe	MaxL シェル	ESSBASEPATH /bin	『Oracle Essbase テクニカル・リファレンス』
esscmd.exe	ESSCMD コマンド・ライン・クライアント・インタフェース	ESSBASEPATH /bin	『Oracle Essbase テクニカル・リファレンス』
adminsvr.exe または startEAS.exe	Essbase 管理サーバーの実行可能ファイル	EPM_ORACLE_HOME / products/Essbase/ eas/server/bin	Oracle Essbase Administration Services Online Help

実行可能ファイル[1]	説明	場所	参照
admincon.bat	管理サービス・コンソール・アプリケーション	EPM_ORACLE_HOME / products/Essbase/ eas/server/bin	Oracle Essbase Administration Services Online Help

[1]UNIX では、実行可能ファイルに.exe 拡張子はありません。

エージェントの理解

Essbase エージェントは Essbase サーバーを管理します。エージェント・プロセスは、すべてのアプリケーションを開始および停止し、Essbase サーバーのトラフィック・コーディネータとして機能します。OPMN は Essbase エージェントを管理します。

Essbase サーバーの開始および停止スクリプトは OPMN にリダイレクトされます。デフォルトでは、OPMN はバックグラウンド・サービスとして Essbase を開始します。

インストール中、Oracle Hyperion Enterprise Performance Management System インストーラは OPMN をインストールし、OPMN 用の Essbase サーバーを登録します。第 45 章「Essbase の管理(OPMN を使用)」を参照してください。

エージェント・ログは、Essbase サーバー・ログと呼ばれます。817 ページの「Essbase サーバー・ログおよびアプリケーション・ログの表示」を参照してください。

マルチスレッド処理

essbase.exe および esssvr.exe(UNIX では、ESSBASE および ESSSVR)は、マルチスレッド処理および対称型マルチプロセッシング(SMP)アプリケーションです。マルチスレッド処理によって、クライアント・サーバー環境における高いパフォーマンスが保証されます。SMP は、1 台のサーバー・コンピュータが複数の Essbase アプリケーションをホストするときの拡張性を提供します。Essbase サーバーは、UNIX オペレーティング・システムに含まれている POSIX カーネル・スレッドを使用しています。

表 132 に示すように、デフォルトでは、スレッド数はポートの数に基づいています。ポート数は、Essbase がサポートする同時接続の数を表します。Essbase には、他のポートがすべて使用されているときにユーザーをログオフするために使用される、システム管理者のための予約ポートが 1 つ用意されています。

表 132　ポートとマルチスレッド処理

ポート数	デフォルトのスレッド数
1 - 5 ポート	5
6 - 10 ポート	10
11 ポート以上	20

エージェントまたは Essbase サーバーのスレッド数は、AGENTTHREADS、AGTSVRCONNECTIONS および SERVERTHREADS 構成設定を使用して essbase.cfg ファイルで設定できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： Essbase サーバーが実行されているコンピュータ上でハイパースレッディングを使用可能にすることはお薦めできません。

エージェント・コマンドおよび等価なコマンドのリスト

OPMN コマンドライン・インタフェースにエージェント・コマンドを入力できます。表 133 は、各エージェント・コマンドと、等価な MaxL、ESSCMD または Administration Services コマンドを示しています:

表 133　エージェント・コマンドと、等価な MaxL、ESSCMD または Administration Services コマンド

エージェント・コマンド	関数	等価な MaxL、ESSCMD または Administration Services コマンド
START appname	指定したアプリケーションを開始します。	• MaxL: **alter system load application appname**; • ESSCMD: LOADAPP • Administration Services: 「開始」、次にエンタープライズ・ビュー内のアプリケーション・ノードの「アプリケーション」
STOP appname	指定したアプリケーションを停止します。	• MaxL: **alter system unload application appname**; • ESSCMD: UNLOADAPP • Administration Services: 「停止」、次にエンタープライズ・ビュー内のアプリケーション・ノードの「アプリケーション」
USERS	Essbase サーバーに接続しているユーザーのリストを表示します。次の情報が表示されます: • Essbase サーバーに接続しているユーザーの名前 • インストールされているポート数 • 接続数 • 各ユーザーが接続しているアプリケーション • 各ユーザーが接続しているデータベース	• MaxL: **display user;** (すべてのユーザーをリストし、どのユーザーがログインしているかを示します) • ESSCMD: LISTUSERS (すべてのユーザーをリストします) • Administration Services: 「編集」、次にエンタープライズ・ビュー内のサーバー・ノードの「セッション」

エージェント・コマンド	関数	等価な MaxL、ESSCMD または Administration Services コマンド
PORTS	Essbase サーバーにインストールされているポート数および使用中のポート数を表示します。	● MaxL: **display system;** (使用可能な未使用のポートを表示します) ● ESSCMD: 該当なし ● Administration Services: 「編集」、次にエンタープライズ・ビュー内のサーバー・ノードの「プロパティ」(「Essbase サーバー・プロパティ」ウィンドウ、「統計」タブ)
LOGOUTUSER `user`	Essbase サーバーからユーザーを切断し、ポートを解放します。 このコマンドには、Essbase のシステム・パスワードが必要です。	● MaxL: **alter system logout session by user username**; ● ESSCMD: LOGOUTUSER ● Administration Services: 「編集」、次にエンタープライズ・ビュー内のサーバー・ノードの「セッション」
PASSWORD	Essbase サーバーの開始に必要なシステム・パスワードを変更します。 このコマンドには、Essbase のシステム・パスワードが必要です。	● MaxL: **alter user `system_administrator` set password `password`**; ● ESSCMD: SETPASSWORD ● Administration Services: 該当なし
COMPACT	エージェント実行中のセキュリティ・ファイルのコンパクト化を使用可能にします。696 ページの「Essbase セキュリティ・ファイルの断片化の管理」を参照してください。 **注：** Essbase では、エージェントが停止するたびに、セキュリティ・ファイルが自動的にコンパクト化されます。	● MaxL: **alter system compact security file;** ● ESSCMD: 該当なし ● Administration Services: エンタープライズ・ビュー内のサーバーのセキュリティ・ノードの「セキュリティ・ファイルのコンパクト化」
DUMP filename	Essbase のセキュリティ・ファイル(`essbase.sec`)から、テキスト(ASCII)フォーマットの指定されたファイルに情報をダンプします。ファイル名のパスを指定しない場合は、ファイルが `ARBORPATH/bin` ディレクトリに保存されます。 このコマンドには、Essbase のシステム・パスワードが必要です。 **注：** サービスとして実行されている Essbase サーバーに対して DUMP コマンドを使用できません。サーバーがサービスとして実行されている場合は、Administration Services の「セキュリティ・ファイルのエクスポート」コマンドか、または **export security_file** MaxL ステートメントを使用します。	● MaxL: **export security_file;** ● ESSCMD: 該当なし ● Administration Services: エンタープライズ・ビュー内のサーバーのセキュリティ・ノードの「セキュリティ・ファイルのエクスポート」 697 ページの「Essbase セキュリティ・ファイルの読取り可能フォーマットへのエクスポート」を参照してください。 **注：** サービスとして実行されている Essbase サーバーに対して「セキュリティ・ファイルのエクスポート」コマンドを使用できません。
VERSION	Essbase サーバー・ソフトウェアのバージョン番号を表示します。	● MaxL: **display system;** ● ESSCMD: GETVERSION ● Administration Services: 「編集」、次にエンタープライズ・ビュー内のサーバー・ノードの「プロパティ」(「Essbase サーバー・プロパティ」ウィンドウ、「ライセンス」タブ)

エージェント・コマンド	関数	等価な MaxL、ESSCMD または Administration Services コマンド
HELP	有効なすべてのエージェント・コマンドとそれぞれの機能をリストします。[Enter]キーを押すのと同じです。	該当なし
QUIT および EXIT	開いているすべてのアプリケーションをシャットダウンして、Essbase サーバーを停止します。	• MaxL: **alter system shutdown;** • ESSCMD: SHUTDOWNSERVER • Administration Services: エンタープライズ・ビュー内のサーバー・ノードの「停止」

Essbase サーバーの開始と停止

サブトピック

- ホスト名で修飾されたポートでの Essbase の開始
- HP-UX および Solaris での Essbase サーバー・パスワードの非表示
- Essbase サーバーのシステム・パスワードの変更
- Essbase サーバーの停止

Essbase サーバーを開始するには、管理者権限を持っている必要があります。

注： ESSCMD または MaxL から Essbase サーバーを開始できません。

ホスト名で修飾されたポートでの Essbase の開始

`essbase.cfg` 内の ESSBASESERVERHOSTNAME 構成設定を使用することにより、Essbase のバインド先のホスト名を指定できます。ホスト名が指定されていない場合、Essbase はシステム API を使用してホスト名を取得します。ホスト名を使用すると、コンピュータのすべてのネットワーク・インタフェース上のネットワーク・トラフィックを分割できます。これは、単一のネットワーク・カードまたは複数の NIC カードが装備されているコンピュータでは便利です。

HP-UX および Solaris での Essbase サーバー・パスワードの非表示

HP-UX および Solaris では、`ps -ef` ユーティリティによって、システム・パスワードを含むプロセス・リストが作成されます。

注： IBM AIX では、Essbase サーバーのシステム・パスワードは自動的に非表示になります。

➤ Essbase サーバーのシステム・パスワードを非表示にするには:

1 次のコマンドを記述した、essbase.secure という名前のシェル・スクリプトを作成します:

```
#!/bin/sh

PASS=$1

ESSBASE -b -secure << EOF &

${PASS}

EOF
```

2 エージェントを起動するには、次のコマンドを使用します:

```
essbase.secure password
```

- 手動で改行を入力することなくこのスクリプトをコマンド・プロンプトに戻すようにするか、またはこのスクリプトをさらに大きなスクリプト内に埋め込むには、このスクリプトを次のコマンドで実行します:

```
essbase.secure &
```

- (オプション)標準出力を、このスクリプトをアクティブでないターミナル・セッションで実行する場合に有効な nohup.out という名前のファイルにリダイレクトするには、次のコマンドを使用します:

```
nohup essbase.secure &
```

Essbase サーバーのシステム・パスワードの変更

Essbase サーバーの開始に必要なパスワードを変更できます。

注: システム・パスワードを変更しても、Essbase システム管理者の接続パスワードは変更されません。

➤ Essbase サーバーのシステム・パスワードを変更するには、次のツールを使用します:

ツール	トピック	場所
エージェント	パスワード	OPMN コマンドライン・インタフェースにエージェント・コマンドを入力します。 現在のシステム・パスワードを入力します。 新しいシステム・パスワードを入力した後、再度入力します。
MaxL	**alter user system_administrator set password password**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETPASSWORD	『Oracle Essbase テクニカル・リファレンス』

Essbase は、システム・パスワードが更新されたことを確認します。

Essbase サーバーの停止

Essbase サーバーを停止またはシャットダウンするには、管理者権限が必要です。

➤ Essbase サーバーおよび実行中のすべてのアプリケーションを停止するには、次のツールを使用します:

ツール	トピック	場所
エージェント	quit exit	OPMN コマンドライン・インタフェースにエージェント・コマンドを入力します。
MaxL	**alter system shutdown**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SHUTDOWNSERVER	『Oracle Essbase テクニカル・リファレンス』

「エージェント」ウィンドウを閉じるか、または[Ctrl] + [C]を押すことによってエージェントを停止した場合は、次回データベースを開始したときに、進行中であったすべてのトランザクションが Essbase によってロールバックされます。866 ページの「コミット・アクセスでのロールバック」または868 ページの「アンコミット・アクセスでのロールバック」を参照してください。

注意 Essbase サーバーを Windows サービスとして実行している場合は、このサービスを Windows のコントロール パネルから停止しないでください。これは、UNIX プラットフォームで kill コマンドを発行することに相当し、データ破損を引き起こす場合があります。

アプリケーションの開始と停止

アプリケーションが開始されると、Essbase は、そのアプリケーションおよび関連付けられたすべてのデータベースを Essbase サーバー・コンピュータ上のメモリー

にロードします。データ・ロード、計算、レポート、スプレッドシートのロックや送信などの、データに対するクライアント要求はすべて、アプリケーション・プロセス(ESSSVR)を介して処理されます。アプリケーション・プロセスは常に、エージェント・プロセス(ESSBASE)によって開始されます。

Essbase サーバー上で、複数のアプリケーション・プロセスを同時に実行できます。Windows では、実行されているアプリケーション・プロセスごとに個別のウィンドウが開きます。アプリケーションに複数の実行中データベースが含まれている場合は、すべてのデータベースが 1 つのアプリケーション・プロセスによって管理されます。

アプリケーションを停止すると、Essbase は、Essbase サーバー・コンピュータ上のメモリーからすべての情報およびデータベースをアンロードし、アプリケーション・プロセスを閉じます。

アプリケーションの開始

アプリケーションを開始すると、次のアクションが可能になります:

- ユーザーはアプリケーションに接続できます。
- アプリケーションはエージェント・コマンドに応答できます。
- ユーザーはアプリケーションの設定を変更できます。
- データとユーザーのセキュリティが使用可能になります。
- アプリケーション内の個々のデータベースを開始できます。

➤ アプリケーションを開始するには、次のツールを使用します:

ツール	トピック	場所
エージェント	START appname	OPMN コマンドライン・インタフェースにエージェント・コマンドを入力します。
Administration Services	アプリケーションの開始	Oracle Essbase Administration Services Online Help
MaxL	**alter system load application**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	LOADAPP または SELECT	『Oracle Essbase テクニカル・リファレンス』

アプリケーションが開始し、さらに Windows 上で実行している場合は、Essbase サーバー・コンピュータ上でアプリケーション・サーバー・ウィンドウが開きます。

また、次のいずれかのアクションを実行した場合にも、アプリケーションを開始できます:

- アプリケーション内でのデータベースの開始。766 ページの「データベースの開始」を参照してください。
- アウトラインの Essbase サーバーへの保存。(アウトラインを開いても、アプリケーションは開始されません。)

アプリケーションの開始方法を制御する次のオプションを設定できます:

- startup (アプリケーションの開始をユーザーに許可): アプリケーションが停止されているときに、ユーザーがそのアプリケーション内の任意のデータベースからデータを取得しようとすると、そのアプリケーションが Essbase サーバー・コンピュータ上で自動的に開始されます。
- autostartup (Essbase の開始時にアプリケーションを開始): ユーザーがそのアプリケーション内のデータベースを要求したときの初期パフォーマンスが向上する場合があります。これは、アプリケーションおよびデータベースが Essbase サーバー・コンピュータ上のメモリーにすでにロードされているためです。

➤ アプリケーションの開始方法を制御するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アプリケーションの自動開始の構成	Oracle Essbase Administration Services Online Help
MaxL	**alter application enable startup** **alter application enable autostartup**	『Oracle Essbase テクニカル・リファレンス』

アプリケーションの停止

含まれているデータベースが破損しないように、アプリケーションを正しく停止してください。アプリケーションを停止するときに、トランザクションが実行されている可能性があります。いずれかの正しい方法(次の表を参照)を使用してアプリケーションを停止した場合は、計算またはデータ・ロードが進行中であると、アプリケーションは停止されません。かわりに、Essbase によって、エージェント・ウィンドウにメッセージが表示されます。

ウィンドウを閉じるか、または[Ctrl] + [C]を押してエージェントを停止した場合は、アプリケーションが停止し、次回アプリケーションを開始したときに、進行中であったすべてのトランザクションが Essbase によってロールバックされます。866 ページの「コミット・アクセスでのロールバック」または868 ページの「アンコミット・アクセスでのロールバック」を参照してください。

➤ アプリケーションを正しく停止するには、次のツールを使用します:

ツール	トピック	場所
エージェント	stop appname	OPMN コマンドライン・インタフェースにエージェント・コマンドを入力します。
Administration Services	アプリケーションの停止	Oracle Essbase Administration Services Online Help
MaxL	**alter system unload application**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	UNLOADAPP	『Oracle Essbase テクニカル・リファレンス』

アプリケーションの誤った停止

「アプリケーション・サーバー」ウィンドウを閉じることによって、アプリケーション・サーバー・プロセスを不正に停止することが必要な場合があります。たとえば、アプリケーション・サーバーが破損しており、クライアント要求を処理していない場合などです。

➤ アプリケーションを正しい方法以外で停止するには、Essbase サーバーが実行されているオペレーティング・システムのメソッドを使用します:

- UNIX プラットフォームでは、ESSSVR プロセスを強制終了します。

  ps の出力を使用して、個々のアプリケーションを識別できます。アプリケーションがフリーズした場合は、次のコマンドを使用してそのアプリケーションを停止できます:

  ```
  kill -9 <pid>
  ```

- Windows の場合は、次のメソッドを選択します:
  - Windows オペレーティング・システムの「タスクの終了」を実行します。

    Windows では、個々の Essbase アプリケーションのプロセス ID は表示されません。実行中のすべての Essbase アプリケーションが、区別されない ESSSVR プロセスとして表示されるため、単一のアプリケーションがフリーズした場合にはそのアプリケーションを停止できません。
  - アプリケーション・サーバー・ウィンドウの「閉じる」ボタンをクリックします。
  - プロセス ID に対して Taskkill を実行します。

    個々のアプリケーション・サーバーのプロセス ID は、EPM_ORACLE_HOME/logs/essbase/essbase.log ファイル内にあります。サーバーが開始したとき、Essbase サーバー・ログに次のような行が表示されます:

    ```
    Application [Sample] started with process id [225]
    ```

データベースの開始と停止

データベースを開始すると、そのデータベースが Essbase サーバー・コンピュータ上のメモリーにロードされます。データベースを停止すると、メモリーからすべてのデータベース情報がアンロードされます。

データベースの開始

Essbase によってデータベースが開始され、メモリーにロードされると、そのデータベースのインデックス・キャッシュ全体がメモリー内に自動的に割り当てられます。データ・キャッシュとデータ・ファイル・キャッシュは、Essbase クライアントから要求されたブロックとして割り当てられます。

アプリケーションを開始すると、Essbase によって、そのアプリケーションおよびそのアプリケーションのデータベースが Essbase サーバー・コンピュータ上のメモリーにロードされます。まだ開始されていないアプリケーションのデータベースを開始すると、そのアプリケーションが、関連するすべてのデータベースとともにメモリーにロードされます。

➤ データベースを開始するには、次のツールを使用します:

ツール	トピック	場所
エージェント	START appname	OPMN コマンドライン・インタフェースにエージェント・コマンドを入力します。
Administration Services	データベースの開始	Oracle Essbase Administration Services Online Help
MaxL	**alter application load database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	LOADDB または SELECT	『Oracle Essbase テクニカル・リファレンス』

➤ データベースを、その親アプリケーションの開始時に自動的に開始するように構成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データベースの自動開始の構成	Oracle Essbase Administration Services Online Help
MaxL	**alter database enable autostartup**	『Oracle Essbase テクニカル・リファレンス』

データベースの停止

データベースを停止すると、メモリーからすべてのデータがアンロードされ、更新済のデータはすべてディスクにコミットされます。データベースが停止されているときに、ユーザーがそのデータベースからデータを取得しようとすると、明示的なコマンドを発行しなくても、そのデータベースが Essbase サーバー上で自動的に開始されます。

データベースを停止するとき、トランザクションが現在実行されている可能性があります。いずれかの正しい方法(次の表を参照)を使用してデータベースを停止した場合は、計算またはデータ・ロードが進行中であると、データベースは停止されません。かわりに、Essbase によって、Essbase サーバー・ウィンドウにメッセージが表示されます。

Essbase サーバー・ウィンドウを閉じるか、または[Ctrl] + [C]を押してエージェントを停止した場合は、データベースが停止し、次回データベースを開始したときに、進行中であったすべてのトランザクションが Essbase によってロールバックされます。866 ページの「コミット・アクセスでのロールバック」または868 ページの「アンコミット・アクセスでのロールバック」を参照してください。

➤ データベースを停止するには、次のツールを使用します:

ツール	トピック	場所
エージェント	STOP appname	OPMN コマンドライン・インタフェースにエージェント・コマンドを入力します。
Administration Services	データベースの停止	Oracle Essbase Administration Services Online Help
MaxL	**alter application unload database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	UNLOADDB	『Oracle Essbase テクニカル・リファレンス』

ポートの管理

サブトピック

- ユーザーおよび使用可能ポートのリストの表示
- 非デフォルト・ポート値の指定
- ポートのデフォルト値の変更
- ポート統計の表示
- Essbase 管理サーバーの通信ポートの管理

エージェントを使用すると、Essbase サーバー上のポートを管理できます。

ユーザーおよび使用可能ポートのリストの表示

任意の時点で Essbase サーバーに接続しているユーザーのリスト、使用可能なポート数および接続数を表示できます。

➤ Essbase サーバーに接続しているユーザーのリストを表示するには、次のツールを使用します:

ツール	トピック	場所
エージェント	USERS	Oracle Process Manager and Notification Server コマンドライン・インタフェースにエージェント・コマンドを入力します。
Administration Services	Essbase サーバー・ユーザーおよびグループの表示	Oracle Essbase Administration Services Online Help
MaxL	**display user**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	LISTUSERS	『Oracle Essbase テクニカル・リファレンス』

➤ Essbase サーバーにインストールされているポート数および使用中のポート数を表示するには、次のツールを使用します:

ツール	トピック	場所
エージェント	PORTS	コマンドライン・インタフェースにエージェント・コマンドを入力します。
Administration Services	**使用可能なポートの確認** 「Essbase の接続およびポートについて」も参照してください	Oracle Essbase Administration Services Online Help
MaxL	**display user**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	LISTUSERS	『Oracle Essbase テクニカル・リファレンス』

非デフォルト・ポート値の指定

エージェントで使用されるデフォルトのポート値を変更するには、次の構成設定を 1 つ以上設定する必要があります:

- AGENTPORT を使用して、エージェントが使用するポートを指定します。
- SERVERPORTBEGIN を使用して、コンピュータ上のエージェントが、最初のサーバー・プロセスで最初に使用を試みるポート番号を指定します。
- SERVERPORTEND を使用して、エージェントが、サーバー・プロセスを開始しようとするときにポートに対して使用を試みる最も大きな値を指定します。この値が使用できない場合、このサーバー・プロセスは失敗します。
- PORTINC を使用して、エージェントが使用するポート番号の増分値を指定します。

デフォルト値の変更が必要になる場合もあります。これには、次の 2 つの理由が考えられます:

- デフォルト値が、すでに使用されているポート番号を指定している場合。
- テストを容易にするために、コンピュータに Essbase サーバーの 2 番目のインスタンスをインストールする必要があるとします。最初とは異なるポートに 2 番目の Essbase サーバー・インスタンスを割り当てるには、Oracle Hyperion Enterprise Performance Management System コンフィグレータを使用します。Oracle Enterprise Performance Management System Deployment Options Guide を参照してください。

ポートのデフォルト値の変更

エージェントおよびサーバー・ポートに関連付けられたデフォルト値を 1 つ以上変更したり、MaxL を使用してポートの範囲を拡張する必要がある場合は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

ポート統計の表示

Essbase で、使用されているポート数を指定した間隔でログに記録するように指定できます。このログの情報を分析することによって、ポートの使用率を監視したり、ユーザーが接続不可能になる前にポート数を増やす必要性を識別したりできます。

Essbase サーバーでポート使用の統計を確認し、これらの統計を Essbase サーバー・ログに書き込むようにするには、PORTUSAGELOGINTERVAL 構成設定を使用します:

```
PORTUSAGELOGINTERVAL
n
```

ここで、n は使用中のポート数を確認する間隔(分数)を表します。n の値には、1 から 60 までの任意の整数を指定できます。5 が推奨される最小値であり、またデフォルト値でもあります。Essbase では、整数以外の部分はすべて無視されます。たとえば、Essbase では 2.5 は 2 分として評価されます。統計は、それぞれの確認の直後に Essbase サーバー・ログに書き込まれます。ログ・ファイルの出力は次のようになります:

```
[Mon Apr 22 00:48:50 2003]Local/ESSBASE0///Info(1056214)
[3] ports in use, [10] ports allowed
```

『Oracle Essbase テクニカル・リファレンス』を参照してください。

Essbase 管理サーバーの通信ポートの管理

Essbase 管理サーバーには、Essbase サーバー・ポートとは別の構成可能な通信ポートがいくつかあります。109 ページの「Essbase 管理サーバーについて」を参照してください。

SSL を使用した Essbase との通信

Essbase は、すべてのクライアント/サーバー間およびサーバー/サーバー間通信のためのセキュア・ソケット・レイヤー(SSL)プロトコルをサポートしています。SSL により、Essbase は、傍受および改ざんを回避しながら Provider Services、Administration Services、Essbase Studio などの Essbase クライアントと一方向ネットワーク通信を行うことができます。エンドポイント認証および通信では、CipherSuite 暗号化を使用してインターネット上で機密性が保持されます。

次の構成設定は、SSL 通信を可能にするパラメータを提供します:

- AGENTSECUREPORT
- CLIENTPREFERREDMODE
- ENABLECLEARMODE

- ENABLESECUREMODE
- NETSSLHANDSHAKETIMEOUT
- SSLCIPHERSUITES
- WALLETPATH

保護モード(SSL)でパーティション化する場合は、次の考慮事項が適用されます:

- パーティションのソースとターゲットのセキュリティ・プロトコルは同じである必要があります(両方ともが SSL を使用するか、どちらも SSL を使用しない)。
- Essbase が SSL 接続を使用できるようにするには、ENABLESECUREMODE を TRUE に設定する必要があります。
- CLIENTPREFERREDMODE を SECURE に設定することを検討します。

  CLIENTPREFERREDMODE が設定されていないか、または、FALSE に設定されているが、ENABLESECUREMODE が TRUE に設定されている場合は、`:secure` を HOST-NAME 文字列に追加することにより、MaxL でパーティションを安全に作成およびリフレッシュできます。たとえば、

```
login esbuser esbpassword on “localhost:6423:secure”;
```

保護モード(SSL)でロケーション別名を使用する場合は、次の考慮事項が適用されます:

- Essbase が SSL 接続を使用できるようにするには、ENABLESECUREMODE を TRUE に設定する必要があります。
- 保護ポートにロケーション別名が設定されている場合は、サーバー名の指定に`:secure` を追加する必要があります。たとえば、MaxL を使用した場合、

```
create location alias EasternDB from Sample.Basic to East.Sales at Easthost:
6423:secure as User1 identified by password1;
```

Essbase を Provider Services との暗号化(SSL)通信用に構成するには、Essbase `libcurl` ライブラリを有効にして Provider Services へのセキュア・チャネルを設定するための構成タスクを実行する必要があります。755 ページの「Provider Services との SSL 暗号化通信のための構成」を参照してください。

SSL と Essbase の使用の詳細は、Oracle Enterprise Performance Management System User Security Administration Guide を参照してください。

クエリーのサイズおよび実行時間の制御

取得するには大きすぎるか、または複雑すぎるためにクエリーのパフォーマンスが低下するか、または正常に完了できない情報を、ユーザーが誤って要求する可能性があります。クエリー・サイズまたは実行時間を制御するには、

QRYGOVEXECTIME および QRYGOVEXECBLK 構成設定(クエリー・ガバナーと呼ばれます)を使用します:

- `QRYGOVEXECTIME [appname [dbname]] n`

  クエリーが終了されるまでに、Essbase サーバーによってそのクエリーの実行が許可される時間を制限します。

- `QRYGOVEXECBLK [ appname [dbname]] n`

  クエリーが終了されるまでに、そのクエリーがアクセスできるブロック数を制限します。

これらの設定を、Essbase サーバー上のすべてのアプリケーションおよびデータベース、単一のアプリケーション上のすべてのデータベース、または 1 つのデータベースに適用できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

Essbase サーバーへのエージェント接続の増加

Essbase サーバーとエージェントの間の可能性のある最大スレッド数を増やすと、複数のユーザーがアプリケーションやデータベースに同時にログオンしたり、接続したりできるようになります。

Essbase サーバーへの初期接続を実行するために作成されるスレッドの最大数を制御するには、AGENTTHREADS および AGTSVRCONNECTIONS 構成設定を使用します:

- `AGENTTHREADS maximum_number_of_threads`
- `AGTSVRCONNECTIONS maximum_number_of_threads`

  AGTSVRCONNECTIONS の `maximum_number_of_threads` 値は、リソースの浪費を防ぐために、AGENTTHREADS の値以下に維持してください。各接続にはサーバーとエージェントからのスレッドがそれぞれつ必要であるため、AGTSVRCONNECTIONS の値の方を大きくする必要はありません。『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： 初期接続から切断までに行われるすべての要求は、サーバー・スレッドの異なるセットによって処理されます。サーバー・スレッドの最大数は、SERVERTHREADS 構成パラメータによって制御されます。

ユーザー・セッション数の制限

Essbase サーバーへの同時ユーザー・セッション接続の最大数を制限するには、MAXLOGIN 構成設定を使用します。この数には、同じユーザーの複数のインスタンスが含まれます。たとえば、同じ Essbase サーバーに接続して Excel ワークシートを 5 つ開いている 1 人のユーザーは、使用しているポートは 1 つですが、5 つのセッションを使用しています。

MAXLOGIN の値を、コンピュータ・リソースに一致するように、または同時ポートおよびユーザー・セッションをより緊密に管理するために調整できます。同時ポートは、クライアント・コンピュータ、Essbase サーバーおよびログイン名の各固有の組合せに対して使用されます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： ユーザー・セッションでは、最大数が SERVERTHREADS 構成設定で制御されるスレッドが使用されます。最大数が AGENTTHREADS および AGTSVRCONNECTIONS 構成設定で制御されるスレッドには関連していません。

47 アプリケーションとデータベースの管理

この章の内容

『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』も参照してください。

アプリケーションおよびデータベースの理解

アプリケーションは、1 つ以上の Essbase データベースおよび関連するファイルを含む管理構造です。Essbase アプリケーションおよびデータベースは通常、Essbase サーバー上に存在します。サーバー・コンピュータは、複数のアプリケーションを保管できます。

Essbase データベースは、多次元データ・ストレージ・アレイを含むデータ・リポジトリです。多次元データベースは、ユーザーがデータを分析して、意味のあるビジネス上の意思決定を行えるように、データの複数のビューをサポートします。

Essbase データベースに関連したファイルは、アーティファクトまたはオブジェクトと呼ばれます。データベース・アーティファクトは、計算の定義やデータに対するレポート作成などの、1 つ以上の Essbase データベースに対するアクションを実行します。デフォルトでは、アーティファクトは、サーバー上の関連付けられたデータベース・フォルダに保管されます。また、一部のアーティファクトは、クライアント・コンピュータまたはその他の使用可能なネットワーク・ディレクトリにも保存できます。776 ページの「Essbase ファイルの保管方法の理解」を参照してください。

Essbase でのデータベース・アーティファクトの一般的なタイプには、次のものがあります:

- データベース・アウトライン(ストレージ構造の定義)
- データ・ソース
- データのロードや動的な次元構築に関する規則(ルール・ファイル)
- データの計算方法を定義するスクリプト(計算スクリプト)

- データに関するレポートを生成するスクリプト(レポート・スクリプト)
- セキュリティ定義
- セキュリティ・フィルタ
- LRO
- パーティション定義

これらのアーティファクトの一部(計算スクリプト、フィルタ、LRO など)はオプションです。

各データベース・アーティファクトの完全な説明については、114 ページの「データベース・アーティファクトの理解」を参照してください。

Essbase ファイルの保管方法の理解

Essbase のインストール・ファイルや、Essbase の使用時に作成されるファイルは、次の場所に保管されます:

- MIDDLEWARE_HOME - Oracle WebLogic Server ホーム(および、オプションで、EPM Oracle ホームを含む 1 つ以上の Oracle ホーム)。

  デフォルト・インストールでは、MIDDLEWARE_HOME は次のとおりです:

  ```
  Oracle/Middleware
  ```

- EPM_ORACLE_HOME-すべての EPM System 製品がインストールされているディレクトリです。

  デフォルト・インストールでは、EPM_ORACLE_HOME は次のとおりです:

  ```
  MIDDLEWARE_HOME
  /EPMSystem11R1
  ```

  例:

  ```
  Oracle/Middleware/EPMSystem11R1
  ```

- EPM_ORACLE_INSTANCE - 配置されている製品の場所(データとアプリケーション、配置されている Java Web アプリケーションおよびログ・ファイルが格納される)。

  デフォルト・インストールでは、EPM_ORACLE_INSTANCE は次のとおりです:

  ```
  Oracle/Middleware/user_projects/epmsystem1
  ```

- ESSBASEPATH - Essbase のインストール・ディレクトリ。

デフォルト・インストールでは、ESSBASEPATH ディレクトリは次のとおりです:

- Essbase サーバーの場合:

```
EPM_ORACLE_HOME
/products/Essbase/EssbaseServer
```

- Essbase クライアントの場合:

```
EPM_ORACLE_HOME
/products/Essbase/EssbaseClient
```

ESSBASEPATH の下に作成されるディレクトリのリストについては、Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

- ARBORPATH:
  - Essbase のアプリケーション・ファイル(作成時の状態)やサンプル・アプリケーションおよびデータベース(Essbase に付属)が保管される、app ディレクトリの場所。
  - bin ディレクトリの場所。ここには、Essbase の構成設定ファイル(essbase.cfg)およびセキュリティ・ファイル(essbase.sec および essbase_timestamp.bak)(作成時の状態)が保管されます。

  デフォルト・インストールでは、ARBORPATH は次のとおりです:

```
EPM_ORACLE_INSTANCE
/EssbaseServer/essbaseserver1
```

- EPM_ORACLE_INSTANCE/common - 共通の内部およびサードパーティ・コンポーネントが保管されるディレクトリ。
- EPM_ORACLE_INSTANCE/diagnostics/logs - ログ・ファイルが保管されるディレクトリ。

MIDDLEWARE_HOME、EPM_ORACLE_HOME および ESSBASE_ORACLE_INSTANCE の詳細は、Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

サーバー・ソフトウェアのファイル・タイプ

表 134 は、Essbase サーバー・ファイルのタイプを示しています:

表 134　Essbase ファイル・タイプ

ファイル拡張子	場所	説明
bak	ARBORPATH /bin	セキュリティ・ファイルのバックアップ
bnd	ESSBASEPATH /bin	DB2 データベースを使用した SQL インタフェース・インストール用の Microsoft ODBC ファイル
cfg	ARBORPATH /bin	Essbase サーバーの構成ファイル
cnt	ESSBASEPATH /bin	オンライン・ヘルプ目次ファイル
cpl	ESSBASEPATH /bin	Windows プラットフォーム用の Microsoft ODBC ドライバ
dll	ESSBASEPATH /bin	Microsoft Windows ダイナミック・リンク・ライブラリ
eqd	ESSBASEPATH /bin	クエリー・デザイナ・ファイル
exe	ESSBASEPATH /bin	実行可能ファイル
hlp	ESSBASEPATH /bin	オンライン・ヘルプ・ファイル
lck	ESSBASEPATH /bin	ロック・ファイル
lic	ESSBASEPATH /bin	ODBC のライセンス情報ファイル
pl	ESSBASEPATH /bin	サンプル Perl スクリプト
pm	ESSBASEPATH /bin	Perl モジュール
mdb	ESSBASEPATH /bin	メッセージ・データベース・ファイル
sec	ARBORPATH /bin	セキュリティ・ファイル
sl	ESSBASEPATH /bin	HP-UX 共有ライブラリ・ファイル
so	ESSBASEPATH /bin	Solaris 共有ライブラリ・ファイル

アプリケーションおよびデータベースのファイル・タイプ

表 135 は、Essbase でアプリケーション、データベースおよびそれに関連するアーティファクトを保管するために使用するファイル・タイプを示しています。

表 135　アプリケーションおよびデータベース用の Essbase ファイル・タイプ

ファイル拡張子	説明
alg	スプレッドシート監査履歴情報
apb	アプリケーション・ファイルのバックアップ
app	アプリケーション・ファイル(アプリケーションの名前、場所およびその他の設定を定義)
arc	アーカイブ・ファイル

ファイル拡張子	説明
atx	スプレッドシート監査トランザクション
chg	アウトライン同期変更ファイル
csc	Essbase 計算スクリプト
db	データベース・ファイル(データベースの名前、場所およびその他の設定を定義)
dbb	データベース・ファイルのバックアップ
ddb	パーティション定義ファイル
ddm	一時パーティション・ファイル
ddn	一時パーティション・ファイル
esm	データ・ブロックへのポインタを管理するとともに、データベース・リカバリに使用される制御情報が含まれている Essbase カーネル・ファイル
esr	一時データベース・ルート・ファイル
esn	一時的な Essbase カーネル・ファイル
ind	Essbase インデックス・ファイル
inn	一時的な Essbase インデックス・ファイル
log	サーバー・ログまたはアプリケーション・ログ
lro	データ・セルにリンクされた LRO ファイル
lst	バックアップ対象ファイルのカスケード目次またはリスト
mxl	MaxL スクリプト・ファイル(Administration Services で保存される)
ocl	データベース変更ログ
ocn	増分再構築ファイル
oco	増分再構築ファイル
olb	アウトライン変更ログのバックアップ
olg	アウトライン変更ログ
otl	Essbase アウトライン・ファイル
otm	一時的な Essbase アウトライン・ファイル
otn	一時的な Essbase アウトライン・ファイル
oto	一時的な Essbase アウトライン・ファイル
pag	Essbase データベース・データ(ページ)ファイル
pan	一時的な Essbase データベース・データ(ページ)ファイル

ファイル拡張子	説明
rep	Essbase レポート・スクリプト
rul	Essbase ルール・ファイル
scr	Essbase ESSCMD スクリプト
sel	保存メンバー選択ファイル
tct	データのすべてのコミットを管理するとともに、すべてのトランザクションに従い、それを保持する Essbase データベース・トランザクション制御ファイル
tcu	一時データベース・トランザクション制御ファイル
trg	トリガー定義ファイル。Extensible Markup Language (XML)フォーマット
txt	テキスト・ファイル(ロードするデータ・ファイルや、LRO としてリンクするテキスト・ドキュメントなど)
xcp	例外エラー・ログ
xls	Microsoft Excel ファイル

API ファイルのタイプ

表 136 は、ESSBASEPATH/api サブディレクトリに保管される Essbase ファイルのタイプを示しています:

表 136　api ディレクトリ内の Essbase ファイル・タイプ

ファイル拡張子	説明
a	UNIX スタティック・ライブラリ・ファイル
bas	Microsoft Visual Basic プログラムのソース・ファイル(Essbase API のヘッダー定義を含む)
h	C または C++のヘッダー・ファイル(Essbase API のヘッダー定義を含む)
lib	C または C++のプログラム・ライブラリ
np	名前付きパイプ・ネットワーク・ライブラリ
tcp	TCP/IP ネットワーク・ライブラリ

アプリケーション、データベースおよびデータベース・アーティファクトの管理

サブトピック

Essbase アプリケーション、データベースおよびデータベース・アーティファクトの説明については、114 ページの「アプリケーションおよびデータベースの理解」および114 ページの「データベース・アーティファクトの理解」を参照してください。

データベースのバックアップおよびリカバリのための戦略

定期的な Essbase バックアップ(本番環境サーバーのメンテナンスに統合される必要があります)は、データベースのメンテナンスにとって重要です。バックアップの頻度は、データベースおよびサーバー環境の非持久性や、迅速なデータベース復元(サーバーの中断が発生した場合)の必要性によって決定される必要があります。

ブロック・ストレージ・データベースをバックアップおよび復元するには、次のいずれかの方法を使用できます:

- 自動化されたデータベース・バックアップおよび復元と、トランザクションのロギングおよび再生

  バックアップおよび復元は、手動によるデータベースのバックアップおよび復元と等価な機能を提供します。バックアップされたデータベースが復元されるとき、バックアップ手順の後に発生したトランザクションはリカバリされません。ただし、トランザクションのロギングおよび再生を使用すると、バックアップの後のトランザクションが取得されるため、それらのトランザクションを再生できます。そのため、バックアップされたデータベースを、中断が発生した前の最も最近の状態にリカバリできます。

  データベース・バックアップおよび復元と、トランザクションのロギングおよび再生機能を使用すると、手動による様々な手順が不要になるため、管理者はデータベースをより効率的にバックアップおよびリカバリできるようになります。これらの機能をバックアップおよびリカバリ戦略に組み込むことをお薦めします。

- 手動によるバックアップおよび復元

  手動による手順を使用したバックアップおよび復元戦略を設計済で、トランザクションのロギングおよび再生の機能を必要としない Essbase の顧客は、引き続き手動の戦略を使用できます。

集約ストレージ・アプリケーションをバックアップおよび復元するには、手動による手順を使用する必要があります。

『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

ファイル・システムを使用したアプリケーションとデータベースの管理(バックアップ時)

プラットフォームのファイル・システムを使用して、アプリケーションおよびデータベースのコピー、移動、名前変更または削除はできません。アプリケーションまたはデータベースがファイル・システムを使用して変更されると、Essbase セキュリティ・ファイルではその変更を認識できません。この状況によって、ハード・ドライブ上に実際に存在するアプリケーションまたはデータベースと、Essbase 上で存在するアプリケーションまたはデータベースの間に不一致が生まれます。

注意 essn.ind、essn.pag、dbname.ind、dbname.esm、dbname.tct のファイルを移動、コピー、変更または削除しないでください。これらの操作を行うと、データが破損する場合があります。

次のアプリケーションおよびデータベース・ファイルは、ファイル・システムを使用して正常に管理できます:

- 次元構築およびデータ・ロードのルール・ファイル(.rul)
- データ・ロードまたは次元構築ファイル
- 計算スクリプト(.csc)
- レポート・スクリプト(.rep)
- MaxL スクリプト(.mxl または任意の拡張子)

➤ アウトライン・ファイル(.otl)をコピーまたは移動するには、Administration Services を使用する必要があります。『Oracle Essbase Administration Services オンライン・ヘルプ』の「アウトラインのコピー」を参照してください。

アプリケーションおよびデータベースの管理にファイル・システムを使用するのは、アプリケーションまたはデータベースのディレクトリ全体をコピーして別の場所に保管する、バックアップ中の期間のみにしてください。『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

アプリケーションの監視

ロードされている各アプリケーションは、オペレーティング・システムのオープンなタスクまたはプロセスです。Windows プラットフォームでは、アプリケーションはコマンドライン・ウィンドウに表示されます。UNIX プラットフォームでは、アプリケーション・サーバーは ESSBASE の子プロセスになります。アプリケー

ションが開始されると、ESSBASE によって esssvr プロセスが開始されます。764 ページの「アプリケーションの開始」を参照してください。

Windows プラットフォームでは、アプリケーションが開始されると、タスクバーに新しいアイコンが表示されます。サーバー・ウィンドウを表示するには、そのアイコンをダブルクリックします。

Essbase サーバーでは、アプリケーションレベルのアクティビティがアプリケーション・ログに記録されます。800 ページの「Essbase ログの使用」を参照してください。

➤ 発生したアプリケーション・アクティビティを表示するには、次のツールを使用します:

ツール	指示
Windows プラットフォームでは、アプリケーション・プロセス・ウィンドウを使用します	アプリケーションの名前が表示されたコマンドライン・ウィンドウを選択します。
UNIX	`tail -f logfile`

Essbase を使用したアプリケーションおよびデータベースの管理

この項では、アプリケーションとデータベースの管理について説明します。

アプリケーションとデータベースの表示

管理サービス・コンソールを開始すると、ナビゲーション・パネルにエンタープライズ・ビュー・ツリーが表示されます。エンタープライズ・ビューは、Essbase 環境のグラフィカルなツリー・ビューです。ここには、選択した管理サーバーと Essbase サーバーが表示されます。Essbase 環境の実際のビューが、他の管理者のビューとは異なる可能性があります。

アプリケーションとデータベースおよびそれに関連付けられたアーティファクトが、「Essbase サーバー」ノードの下のノードとして表示されます。アーティファクトは、コンテナ・ノードにグループ化されます。たとえば、個々のアプリケーションは「アプリケーション」ノードに含まれ、データベースは「データベース」コンテナ・ノードに含まれています。Essbase サーバーにサンプル・アプリケーションおよびデータベースがインストールされている場合は、組織のアプリケーションおよびデータベースとともにエンタープライズ・ビューに表示されます。

Oracle Essbase Administration Services Online Help の「エンタープライズ・ビューについて」を参照してください。

➤ アプリケーションを作成するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アプリケーションの作成」を参照してください。

アプリケーションのコピーまたは移行

アプリケーションは、適切なアクセス権がある任意の Essbase サーバーにコピーできます。アプリケーション全体を別の Essbase サーバーにコピー(移行)するか、または同じ Essbase サーバー上のアプリケーションをコピーできます。たとえば、アプリケーション全体を開発サーバーから本番環境サーバーに移行することが必要な場合があります。あるいは、テストまたはバックアップの目的で、同じサーバー上のアプリケーションをコピーすることが必要になる場合もあります。

Essbase では、同じ Essbase サーバーまたは別の Essbase サーバーのどちらにコピーするかに応じて、アプリケーションを異なる方法でコピーします。アプリケーションを移行する場合は、計算スクリプト、レポート・スクリプト、ルール・ファイル、カスタム定義のマクロおよび関数、代替変数、フィルタなどの、移行するアーティファクトを選択できます。また、ユーザーおよびグループ・セキュリティの移行方法を指定できます。

Administration Services には、アプリケーションの移行に役立つ移行ウィザードが用意されています。Oracle Essbase Administration Services Online Help の「移行ウィザード」を参照してください。

➤ アプリケーションをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アプリケーションのコピー	Oracle Essbase Administration Services Online Help
MaxL	**create application as**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYAPP	『Oracle Essbase テクニカル・リファレンス』

アプリケーション名の変更

アプリケーションの名前を変更すると、そのアプリケーションとそれに関連付けられたディレクトリ(ARBORPATH/app/appname)の名前が変更されます。そのアプリケーションと同じ名前を持つアプリケーション内のすべてのアーティファクト(たとえば、データベースや計算スクリプト)は、名前が変更されません。アプリケーションの名前を変更する前に、1201 ページの「アプリケーションとデータベースの命名規則」を参照してください。

➤ アプリケーションの名前を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アプリケーション名の変更	Oracle Essbase Administration Services Online Help
MaxL	**alter application**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	RENAMEAPP	『Oracle Essbase テクニカル・リファレンス』

アプリケーションの削除

アプリケーションを削除すると、そのアプリケーション内のすべてのアーティファクトも削除されます。ARBORPATH/app/appname ディレクトリと、そのディレクトリ内のすべてのファイルが削除されます。

➤ アプリケーションを削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**アプリケーションの削除**	Oracle Essbase Administration Services Online Help
MaxL	**drop application**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	DELETEAPP	『Oracle Essbase テクニカル・リファレンス』

データベースのコピー

アプリケーション内のデータベースは、適切なアクセス権がある任意の Essbase サーバーおよびアプリケーションにコピーできます。データベース全体を別の Essbase サーバーにコピー(移行)するか、または同じ Essbase サーバー上のデータベースをコピーできます。たとえば、データベース全体を開発サーバーから本番環境サーバーに移行することが必要な場合があります。あるいは、テストまたはバックアップの目的で、同じサーバー上のデータベースをコピーすることが必要になる場合もあります。Essbase は、同じ Essbase サーバーまたは別の Essbase サーバーのどちらにコピーするかに応じて、データベースを異なる方法でコピーします。Oracle Essbase Administration Services Online Help の「データベースのコピー」を参照してください。

Administration Services には、アプリケーションおよびデータベースの移行に役立つ移行ウィザードが用意されています。Oracle Essbase Administration Services Online Help の「移行ウィザード」を参照してください。

データベースをコピーすると、データ・ファイル(.pag および.ind)を除く、そのデータベースに関連付けられたすべてのファイルが宛先アプリケーションにコピーされます。コピーする前に、データベースおよび関連するファイルの完全なコピーを保管するための十分なディスク・スペースがあることを確認してください。

➤ データベースをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**データベースのコピー**	Oracle Essbase Administration Services Online Help
MaxL	**create database as**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYDB	『Oracle Essbase テクニカル・リファレンス』

注： Essbase では、非 Unicode データベースを Unicode アプリケーションにコピーできます。ただし、Unicode データベースの非 Unicode アプリケーションへのコピーは許可されていません。

集約ストレージ・アプリケーションをコピーする前に、すべての増分データ・スライスをメイン・データベース・スライスにマージする必要があります。マージされていない増分データ・スライスのデータはコピーされません。1065 ページの「増分データ・スライスのマージ」を参照してください。

データベース名の変更

データベースの名前を変更すると、そのデータベースとそれに関連付けられたディレクトリ(ARBORPATH/app/appname/dbname)およびアウトライン・ファイル(.otl)の名前が変更されます。そのデータベースと同じ名前を持つデータベース内の他のすべてのアーティファクト(たとえば、計算スクリプト)は、名前が変更されません。

➤ データベースの名前を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データベース名の変更	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	RENAMEDB	『Oracle Essbase テクニカル・リファレンス』

データベースの削除

データベースを削除すると、そのデータベース内のすべてのアーティファクトも削除されます。ARBORPATH/app/appname/dbname ディレクトリと、そのディレクトリ内にあるすべてのファイルが削除されます。

➤ データベースを削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データベースの削除	Oracle Essbase Administration Services Online Help
MaxL	**drop database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	DELETEDB	『Oracle Essbase テクニカル・リファレンス』

Essbase を使用したアーティファクトの管理

この項では、アウトライン、計算スクリプト、レポート・スクリプト、ルール・ファイル、データ・ソースなどのアーティファクトのコピー、名前変更および削除について説明します。114 ページの「データベース・アーティファクトの理解」を参照してください。

注意 アプリケーションの管理にファイル・システムを使用するのは、アプリケーションまたはデータベースのディレクトリ全体をコピーして別の場所に保管する、バックアップ中の期間のみにしてください。

アーティファクトのコピー

アウトラインを除く任意のデータベース・アーティファクトを、別のアプリケーション、データベース、サーバーまたはクライアントの場所にコピーできます。アウトラインをコピーする手順については、126 ページの「アウトラインの作成および編集」を参照してください。

➤ アーティファクトをコピーするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	特定のアーティファクトのコピーに関する項目(「ルール・ファイルのコピー」など)	Oracle Essbase Administration Services Online Help
MaxL	**alter object**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COPYOBJECT	『Oracle Essbase テクニカル・リファレンス』

アーティファクトの名前変更

アウトラインを除く、任意のアーティファクトの名前を変更できます。アウトラインの名前は常にデータベースと同じであるため、アウトラインの名前を変更するには、データベースの名前を変更する必要があります。

➤ アーティファクトの名前を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	特定のアーティファクトの名前変更に関する項目(「ルール・ファイルの名前変更」など)	Oracle Essbase Administration Services Online Help
MaxL	**alter object**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	RENAMEOBJECT	『Oracle Essbase テクニカル・リファレンス』

アーティファクトの削除

アウトラインを除く、任意のアーティファクトを削除できます。アウトラインはデータベースの必須の要素であるため、アウトラインを削除するには、データベースを削除する必要があります。

➤ アーティファクトを削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	特定のアーティファクトの削除に関する項目(ルール・ファイルの削除に関する項など)	Oracle Essbase Administration Services Online Help
MaxL	**drop object**	『Oracle Essbase テクニカル・リファレンス』

ツール	トピック	場所
ESSCMD	アーティファクトを削除するための DELETE コマンド	『Oracle Essbase テクニカル・リファレンス』

アーティファクトのロックおよびロック解除

Essbase では、データベース・アーティファクトのチェックアウト機能を使用して、アーティファクトが一度に 1 人のユーザーによってのみ変更されることを保証します。この項では、アウトラインを除くアーティファクトのロックおよびロック解除を行う方法について説明します。127 ページの「アウトラインのロックおよびロック解除」を参照してください。

注： アーティファクトのロックは、データ・ブロックのロックとは異なります。Essbase カーネルでは、データ・ブロックのロックは処理されますが、アーティファクトのロックは処理されません。862 ページの「データ・ロック」を参照してください。

デフォルトでは、データベース・アーティファクトを開くと常に、Essbase からそのアーティファクトをロックするよう求められます。このデフォルトの動作を、一部のアーティファクトに対して変更できます。『Oracle Essbase Administration Services オンライン・ヘルプ』の「Oracle Essbase Administration Services Online Help のデフォルトのオプションの設定」を参照してください。

アーティファクトがロックされていると、他のユーザーによるそのアーティファクトの上書き、名前変更または削除は Essbase によって許可されません。ロックされたアーティファクトを開いて編集することは可能ですが、既存のアーティファクトを上書きすることはできません。ロックされたアーティファクトに加えた変更を保存するには、変更されたアーティファクトを別の場所に保存します。ロックされたアーティファクトの実行およびコピーは可能です。

アーティファクトのロック解除

権限に応じて、アーティファクトのロックを解除できます。Administration Services では、Essbase サーバー、アプリケーションまたはデータベースのすべてのアーティファクト・ロックを表示できます。

➤ アーティファクトのロックを解除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	オブジェクトのロックおよびロック解除	Oracle Essbase Administration Services Online Help
MaxL	**alter object**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	UNLOCKOBJECT	『Oracle Essbase テクニカル・リファレンス』

Administration Services を使用したアプリケーションの移行

Administration Services を使用すると、プラットフォームには関係なく、適切なアクセス権がある任意の Essbase サーバーにアプリケーションを移行できます。たとえば、アプリケーションを開発サーバーから本番環境サーバーに移行することが必要な場合があります。アプリケーションを移行する場合は、計算スクリプト、レポート・スクリプト、ルール・ファイル、カスタム定義のマクロおよび関数、代替変数、フィルタなどの、移行するアーティファクトを選択できます。また、ユーザーおよびグループ・セキュリティの移行方法を指定できます。

➤ アプリケーションを移行するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アプリケーションのコピー」を参照してください。

プラットフォーム間でのアプリケーションの移植

Essbase は、Windows や UNIX を含む、複数のプラットフォーム上で動作します。サポートされているプラットフォームのリストについては、Oracle Hyperion Enterprise Performance Management System 動作保証マトリックス(http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html)を参照してください。

アプリケーションを作成した後、そのアプリケーションを別のオペレーティング・システムを実行しているサーバーに移植することが必要になる場合があります。この項では、アプリケーションを別の Essbase コンピュータに移植する方法について説明します。

サーバー間での Essbase アプリケーションの移植には、次の手順が必要になります:

1. 789 ページの「互換性のあるファイルの特定」
2. 790 ページの「ファイル名のチェック」
3. 792 ページの「互換性のあるファイルの転送」
4. 793 ページの「データベースの再ロード」

互換性のあるファイルの特定

別のオペレーティング・システムを使用するサーバーに Essbase アプリケーションを移植している場合は、新しいオペレーティング・システムと互換性のある Essbase ファイルを特定する必要があります。

次のタイプのファイルは、オペレーティング・システム間で互換性があります:

- テキスト・ファイル。Essbase のテキスト・ファイルは、計算スクリプト(`.csc`)とレポート・スクリプト(`.rep`)、およびユーザーが開発した任意の MaxL または ESSCMD スクリプトです。また、データ・ファイルもテキスト・ファイルである場合があります。

- ルール・ファイル。これらのファイルはバイナリ・ファイルですが、オペレーティング・システム間で互換性があります。ルール・ファイルの拡張子は.rul です。
- アウトライン・ファイル。これらのファイルはバイナリ・ファイルですが、オペレーティング・システム間で互換性があります。アウトライン・ファイルの拡張子は.otl です。

次のタイプのファイルは、オペレーティング・システム間で互換性がありません。したがって、新しいサーバー上で再定義または再ロードする必要があります:

- データベース・ファイル(拡張子は.db および.dbb)
- データ・ファイル(拡張子は.pag)
- インデックス・ファイル(拡張子は.ind)
- セキュリティ・ファイル(拡張子は.sec)
- アプリケーション・ファイル(拡張子は.app および.apb)
- Essbase カーネル・ファイル(拡張子は.esm)

注： リンク・レポート・オブジェクト機能を使用している場合は、新しいサーバー上のファイルまたはセル・ノートをすべて再リンクする必要があります。LRO の使用方法の詳細は、第 11 章「Essbase データへのオブジェクトのリンク」を参照してください。

ファイル名のチェック

ファイルを UNIX システムに転送する場合は、ファイル名の大文字と小文字の区別に注意してください。UNIX は大文字と小文字が区別されるオペレーティング・システムであるため、ファイルは、ファイル名の大文字と小文字が一致している場合にのみ認識されます。たとえば、特定の MaxL および ESSCMD 操作ではファイル名を指定する必要があり、ファイル名の大文字と小文字を正しく入力する必要があります。

Essbase のシステム・ファイルは、UNIX システム上で次の命名ルールを使用します:

- 実行可能ファイルには拡張子は付加されず、ファイル名はすべて大文字です(ESSBASE、ESSCMD など)。
- スタティック・ライブラリの拡張子は.a であり、ファイル名はすべて小文字です(libessnet.a など)。
- 共有ライブラリ・ファイルの拡張子は、HP-UX 上では.sl、Solaris 上では.so、AIX 上では.a です。これらのファイル名はすべて小文字です(libesscur.sl など)。
- セキュリティ・ファイルの拡張子は.sec であり、ファイル名はすべて小文字です(essbase.sec など)。
- メッセージ・データベース・ファイルの拡張子は.mdb であり、ファイル名はすべて小文字です(essbase.mdb など)。

- オンライン・ヘルプ・ファイルの拡張子は.hlp であり、ファイル名はすべて小文字です(esscmd.hlp など)。

UNIX システムでは、Essbase ファイルの大文字と小文字が正しく設定されます。最初の文字が大文字であり、残りの文字は小文字です。表 137 は、各種のファイル・タイプに対する名前の例を示しています:

表 137　UNIX におけるファイルの名前付けの例

ファイル・タイプ	例
データベース・ファイル	Mydb.db
データ・ファイル	Mydb.pag
インデックス・ファイル	Mydb.ind
アウトライン・ファイル	Mydb.otl
ルール・ファイル	Atlanta.rul
ロード元のデータ・ファイル	Atlanta.txt
計算スクリプト	Mycalc.csc
レポート・スクリプト	Myrepo.rep
アーカイブ・ファイル	Mydb.arc
アプリケーション・ログ	Myapp.log

注： アプリケーション名は、上のルールの例外です。アプリケーション名は、小文字にできます。

表 138 は、UNIX 上システムで有効なファイル名と無効なファイル名のいくつかの例を示しています:

表 138　UNIX 上で有効なファイル名と無効なファイル名

有効なファイル名	無効なファイル名
Model.csc	MODEL.CSC
Monthly.rep	Monthly.Rep
Forecast.otl	forecast.otl
Actuals.rul	AcTuAlS.rUl
My_File.txt	My_File.Txt

注： Essbase では、アプリケーション、データベース、計算スクリプト、レポート、その他のデータベース・ファイルに長いファイル名は許可されません。アーティファクトに対して作成するファイル名はすべて、Windows 8.3 の規則に従う必要があります。

互換性のあるファイルの転送

2 つのサーバーが接続されている場合は、新しいサーバー上にアプリケーションおよびデータベース・ディレクトリを作成し、FTP (File Transfer Protocol)または Administration Services のどちらかを使用して互換性のあるアプリケーション・ファイルを転送できます。これらのサーバーが接続されていない場合は、データベースを再ロードする前に、新しいサーバー上でサーバー情報を再定義する必要があります。

FTP を使用したファイルの転送

FTP を使用すると、オペレーティング・システム間で直接ファイルを転送できます。オペレーティング・システム間で互換性のあるファイルのみを転送する必要があり、またバイナリ・モードでファイルを転送する必要があります。

UNIX サーバー上でファイルの大文字と小文字が間違っていると、Administration Services はそれらのファイルを認識できますが、開くことはできません。FTP を使用してファイルを転送した後、サーバー上でそれらのファイルの名前を変更して、ファイルの大文字と小文字が正しく設定されるようにします。または、ファイルの転送時に、FTP を使用してファイルの名前を変更できます。例:

```
ftp>put oldfile Newfile
```

Administration Services を使用したファイルの転送

Administration Services を使用すると、次の方法でファイルをクライアント・コンピュータからサーバーに転送できます:

- アプリケーションの移行の一部として。Oracle Essbase Administration Services Online Help の「移行ウィザード」を参照してください。
- データベースの移行の一部として。Oracle Essbase Administration Services Online Help の「データベースのコピー」を参照してください。
- アーチファクトを一度に 1 つずつ。たとえば、Oracle Essbase Administration Services Online Help の「ルール・ファイルのコピー」などがあります。

サーバー情報の再定義

移植先のサーバーが元のサーバーに接続されていない場合は、新しいサーバー上でサーバー情報を再定義する必要があります。

➤ サーバー情報を再定義するには:

1 **ユーザーを作成して各ユーザーの権限を指定するには、新しいサーバー上で Administration Services を使用します。**

674 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーおよびグループへの権限の付与」を参照してください。

2 移植対象のアプリケーションおよびデータベースを作成するには、新しいサーバー上で Administration Services を使用します。

第 6 章「アプリケーションとデータベースの作成」を参照してください。

3 移植対象のデータベースのアウトライン・ファイル(.otl)を、元のサーバーから新しいサーバー上の同じディレクトリの場所にコピーします。これらのファイルのコピー中にアプリケーションが実行されていないことを確認してください。

126 ページの「アウトラインの作成および編集」を参照してください。

4 互換性のあるファイルを、元のサーバーから新しいサーバーにコピーします。

789 ページの「互換性のあるファイルの特定」を参照してください。

5 データベースを再ロードします。

793 ページの「データベースの再ロード」を参照してください。

データベースの再ロード

サーバー間でアプリケーションまたはデータベースをコピーする際、.db、.pag、.esm、.ind などのデータベース・ファイルは含まれず、オペレーティング・システム間で互換性がありません。サーバー間でアプリケーションやデータベースをコピーする場合、または別のオペレーティング・システム上のサーバーにアプリケーションを移植する場合は、データ・ファイルおよびルール・ファイル(該当する場合)のデータを再ロードすることによってデータベースを設定しなおす必要があります。再ロードの 1 つの方法として、データをテキスト・ファイルにエクスポートし、そのテキスト・ファイルを新しいサーバーに転送した後、そのテキスト・ファイルを使用してデータをロードする方法があります。ロードが完了したら、新しいデータベースを計算します。

48 データ、アプリケーションおよびデータベースの監視

この章の内容

データ変更の監視(トリガーを使用)

Essbase で提供されるトリガー機能によって、データベース内のデータ変更の効率的な監視が可能になります。

データがトリガーで指定されているルールに違反した場合、Essbase では関連情報をファイルに記録するか、またはトリガーによっては電子メール・アラートを(ユーザーまたはシステム管理者に)送信できます。たとえば、西部地域で、ある月の販売が前の年の同じ月の販売を下回った場合は販売マネージャに通知します。

トリガーには、次の 2 つのタイプがあります。更新時トリガーは、トリガーによって参照されるデータを含むブロックがメモリーに読み込まれたとき、更新処理中にアクティブ化されます。更新完了時トリガーは、更新トランザクションが完了した後にアクティブ化されます。

注： 更新時トリガーは、ブロック・ストレージ・データベースのみでサポートされています。

トリガーの管理

トリガーを管理するには、ユーザーにデータベース・マネージャのセキュリティ権限が必要です。Essbase では、次のアクティビティ中にトリガーを監視するため、これらのアクティビティによってトリガーがアクティブ化される可能性があります:

- データ・ロード
- 計算
- Oracle Essbase Spreadsheet Add-in からロックして送信(ブロック・ストレージ・データベースに対する更新時トリガーでのみ使用可能)

データベースの再構築中に Essbase によってトリガーがアクティブ化されることはありません。

➤ トリガーを作成、変更、表示および削除するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	トリガーの作成 トリガーの編集 トリガーの表示 トリガーの有効化および無効化 トリガー・スプール・ファイルの表示 トリガーの削除	Oracle Essbase Administration Services Online Help
MaxL	**create trigger** **create or replace trigger** **alter trigger** **display trigger** **drop trigger**	『Oracle Essbase テクニカル・リファレンス』

使用可能なトリガーおよび使用不可のトリガーに関する情報は、Essbase サーバーを開始したときにアプリケーション・ログ・ファイルで確認できます。

更新時トリガーの作成

更新時トリガーを作成する際は、次の点に注意してください:

- トリガーの領域指定を定義する場合は、対称的な WHERE 句を使用する必要があります。『Oracle Essbase テクニカル・リファレンス』の MaxL の項にある MDX のドキュメンテーションを参照してください。
- Essbase で電子メール・アラートを送信できるようにするには、システムに JVM がインストールされている必要があります。

  注: Unicode モードのアプリケーションに関連する電子メールは UTF-8 でエンコードされるので、UTF-8 に対応した電子メール・リーダーが必要です。
- 次の Essbase 機能からのデータが必要なトリガーを定義することはできません:
  - 動的計算
  - ハイブリッド分析
  - パーティション化
- スプール・ファイルに、すべてのトリガー・データ値または最新の値のみのどちらを保管するかを指定できます(たとえば、MaxL の **create trigger** ステートメントまたは **create and replace trigger** ステートメントで log_value パラメータを使用します)。log_value パラメータが ON に設定されている場合は、新しい値と古い値の両方がスプール・ファイルに記録されます。log_value パラメータが OFF に設定されている場合は、値がスプール・ファイルに記録されませ

ん。log_value パラメータは、データ・ロードおよびロックして送信アクティビティに対してのみ有効です。

電子メール・アラートを送信する場合は、データのセキュリティを考慮してください。Essbase では、トリガーをアクティブ化して電子メールを送信する場合、電子メールの受信者にトリガー条件で参照されるデータを表示する権限があるかどうかを確認できません。

トリガー条件で疎次元メンバーを参照することは避けてください。Essbase では、あるデータ・ブロックに対するトリガーを実行する場合、更新されているかどうかわからない別のデータ・ブロックを読み取る必要があります。2 つ目のブロックの状態によっては、Essbase によってトリガーが誤ってアクティブ化される可能性があります。

次の例は、Sample.Basic データベースに基づいています。East と West が疎次元であり、次のステートメントによってトリガー条件が定義されるとします:

```
   FIX (East, West, Sales, Jan, Actual, Cola)
IF ((Sales -> East + Sales -> West) > 20)
EMAIL sales@.com
```

East と West について、次の Sales データ値が保管されているとします:

- Sales -> East = 5
- Sales -> West = 6

次に、ユーザーがロックして送信の要求を実行し、データ値を次のように更新するとします:

- Sales -> East = 15
- Sales -> West = 3

Sales -> East が 15 に更新された時点で、データベースには一時的に次の値が含まれます:

- Sales -> East = 15
- Sales -> West = 6

15 プラス 6 は 20 を超えているため、Essbase によってトリガーがアクティブ化されます。その後、Essbase によって Sales -> West が 3 に更新されます。現在は 15 + 3 < 20 であるため、データはトリガー条件を満たしません。ただし、Essbase によって、トリガーはすでにアクティブ化されています。

データ・ロードに続いて計算処理が実行され、ロードされたデータと計算されたデータの両方がトリガー条件を満たした場合は、Essbase によって、トリガーがデータ・ロード時に 1 回と計算時に 1 回の計 2 回アクティブ化されます。

更新完了時トリガーの作成

更新完了時トリガーを作成する際は、次の点に注意してください:

- トリガーの領域指定を定義する場合は、対称的な WHERE 句を使用する必要があります。『Oracle Essbase テクニカル・リファレンス』の MaxL の項にある MDX のドキュメンテーションを参照してください。
- Essbase ハイブリッド分析機能からのデータが必要なトリガーを定義することはできません。
- スプレッドシートのロックして送信操作によって更新完了時トリガーはアクティブ化されません。

パフォーマンスとメモリー使用率に対するトリガーの影響

データベースで使用可能なトリガーの数によっては、計算やデータ・ロードのパフォーマンスが若干低下することがあります。essbase.cfg ファイル内の TRIGMAXMEMSIZE 構成設定を指定することによって、トリガー機能で使用されるメモリーの最大容量を制御できます。デフォルトでは、TRIGMAXMEMSIZE は 4096 バイトに設定されています。情報の電子メールでの送信ではなく、ファイルへの書込みを選択すると、ネットワーク・トラフィックの削減により、計算やデータ・ロードのパフォーマンスが向上する場合があります。

トリガーの例

次の例は Sample Basic データベースに基づいています。

注： ハイブリッド分析メンバーからのデータが必要なトリガーを定義することはできません。動的計算メンバーまたは別のパーティションのメンバーからのデータが必要な更新時トリガーを定義することはできません。

例 1: 1 月の売上の追跡

例 1 では、次の月、製品および地域の Actual および Sales 値が追跡されます:

- 1 月(年次元メンバー Jan)
- コーラ(製品次元メンバー 100)
- 東部地域(市場次元メンバー East)

計算されているメンバーが Jan であり、コーラの 1 月の Actual および Sales 値が 20 を超えている場合、この例では 2 つの電子メール・アカウントに電子メールが送信されます。

```
   create trigger Sample.Basic.Trigger_Jan_20
where "(Jan,Sales,[100],East,Actual)"
when Jan > 20 and is(Year.currentmember,Jan) then
mail ([Docs.Company.com],[trgsales@company.com],
 [inventory@company.com],
[Mail sent by trigger_Jan_20])
end;
```

例 2: 第 1 四半期の売上の追跡

例 2 では、次の月、製品および地域の Actual および Sales 値が追跡されます:

- 1 月、2 月および 3 月(年次元メンバー Qtr1 の子)
- コーラ(製品次元メンバー 100)
- 東部地域(市場次元メンバー East)

計算されているメンバーが Jan、Feb または Mar であり、コーラの 1 月、2 月または 3 月のいずれかの月の Actual および Sales 値が 20 を超えている場合、この例ではファイル Trigger_Jan_Sales_20、Trigger_Feb_Sales_20 または Trigger_Mar_Sales_20 にエントリが記録されます。その後トリガーがアクティブ化されると、古いログの値と新しいログの値の両方がログ・ファイルに保持されます。

```
   create or replace trigger Sample.Basic.Trigger_Qtr1_Sales
log_value on
Where "(crossjoin(Qtr1.children, {(Measures.Sales, [100],
 East, Scenario.Actual)}))"
When Year.Jan > 20 and is(Year.currentmember, Jan) then
 spool Trigger_Jan_Sales_20
When Year.Feb > 20 and is(Year.currentmember, Feb) then
 spool Trigger_Feb_Sales_20
When Year.Mar > 20 and is(Year.currentmember, Mar) then
 spool Trigger_Mar_Sales_20
end;
```

例 3: 在庫レベルの追跡

例 3 では、次の製品、地域および月の在庫レベルが追跡されます:

- コーラ(製品 100)
- 東部地域(市場 East)
- 1 月、2 月および 3 月(Qtr1 の子)

トリガーは、更新アクションが完了した後にアクティブ化されます。東部地域におけるコーラの在庫が 500,000 を下回ると、この例ではファイル Inventory_East にエントリが記録されます。

```
   create after update trigger Sample.Basic.Inventory_east
where "(crossjoin ({children([Qtr1])},
{([Market].[East], [Product].[100],
 [Inventory].[Ending Inventory])}))"
when [Ending Inventory] < 500000 then
 spool Inventory_East
end;
```

Essbase ログの使用

このトピックでは、サーバー、アプリケーションおよびデータベース・アクティビティに関する情報を記録するために Essbase サーバーで作成されるログについて説明します。表 139 に、各ログの概要を示します。

Essbase では、ロギングのために Oracle Diagnostic Logging フレームワーク(ODL)も使用されます。ODL の Essbase 実装の詳細は、『Oracle Hyperion Enterprise Performance Management System インストールおよび構成トラブルシューティング・ガイド』を参照してください。

表 139　ログの要約

ログのタイプ	ログの場所	記録される情報
Essbase サーバー・ログ	EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/essbase_0/ESSBASE.LOG	サーバー・アクティビティとエラー
アプリケーション・ログ	EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/essbase_0/app/ appname / appname .LOG	アプリケーション・アクティビティとエラー
クエリー・ログ	EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/essbase_0/app/ appname / dbname / dbname 00001.qlg	Essbase データベース取得の問合せパターン
アウトライン変更ログ	EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/essbase_0/app/ appname / dbname / dbname .olg	アウトラインへの変更
例外ログ	次の場所のいずれか: ARBORPATH /log00001.xcp EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/essbase_0/app/log00001.xcp EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/essbase_0/app/ appname /log00001.xcp EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/essbase_0/app/ appname / dbname /log00001.xcp	Essbase サーバーの異常停止の原因となるエラー
次元構築およびデータ・ロードのエラー・ログ	次の場所のいずれか(表 149 を参照): EAS_HOME /client/dataload.err EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/app/ appname / appname .log	次元構築またはデータ・ロードによるエラー

このトピックでは、ログに書き込まれる情報と、それらの情報を使用して Essbase サーバーの管理、調整またはトラブルシューティングを行う方法について説明します。

Administration Services のログについては、Oracle Essbase Administration Services Online Help の「Administration Server のログについて」を参照してください。

Essbase サーバー・ログおよびアプリケーション・ログ

Essbase サーバーでは、Essbase サーバーおよびアプリケーションで発生するアクティビティを、拡張子が.log のテキスト・ファイルに書き込みます(つまり、ログに記録します)。

標準の Essbase インストールでは、ESSBASE.LOG という名前の Essbase サーバー・ログが次のディレクトリに存在します:

```
EPM_ORACLE_INSTANCE
/diagnostics/logs/essbase/essbase_0/ESSBASE.LOG
```

Essbase サーバー上の各アプリケーションには、そのアプリケーションに従って名前が付けられた独自のアプリケーション・ログがあります。たとえば、Sample アプリケーションのログ・ファイルには、sample.log という名前が付けられます。標準の Essbase インストールでは、DEFAULTLOGLOCATION 構成パラメータの値に応じて、アプリケーション・ログは次のいずれかのディレクトリにあります:

- EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/essbase_0/app/appname
- ARBORPATH/app/appname

アプリケーション・ログ内の情報は、エラーが発生した場所や原因を正確に特定するのに役立ちます。

サーバー・ログおよびアプリケーション・ログに関して実行できるアクションについては、814 ページの「Essbase サーバー・ログおよびアプリケーション・ログの使用」を参照してください。Administration Services を使用したログの表示または分析については、Oracle Essbase Administration Services Online Help の「ログ・ビューアについて」または「ログ・アナライザについて」を参照してください。特定のエラー・メッセージの詳細は、『Oracle Essbase エラー・メッセージ・リファレンス』を参照してください。

Essbase サーバー・ログのコンテンツ

ESSBASE.LOG ファイル内の情報は、次の項目の評価に役立ちます:

- 操作を実行したユーザー
- 操作が実行された日時
- 操作が実行または試行されたときに発生したエラー

表 140 は、記録されるアクションのタイプと、ログ・メッセージに含まれる情報を示しています。特定のエラー・メッセージについては、『Oracle Essbase エラー・メッセージ・リファレンス』を参照してください。

表 140 Essbase サーバー・ログのコンテンツ

アクションのタイプ	記録される情報
Essbase サーバーのレベルで実行されたアクション(Essbase サーバーへのログオンや、セキュリティの設定など)	● ユーザー名 ● 日付と時刻 ● 権限を変更した場合- ユーザーおよびグループの情報と、ユーザーの IP アドレス ● ログオンした場合- ユーザーが最後にログオンした日時 ● ログオフした場合- ユーザーがログオンしていた時間 ● アプリケーションまたはデータベースを作成または変更した場合- アーティファクトを作成したり開いたりする要求。作成したアプリケーションまたはデータベースのロード、接続、起動。作成または変更したアーティファクトのロック ● アプリケーションまたはデータベースを作成、変更または表示した場合- アプリケーションおよびデータベースのリスト表示の要求およびアクセス情報取得の要求 ● シャットダウンまたは起動の要求 ● Essbase サーバー・ログの取得および削除の要求
アプリケーション・レベルで実行されるアクション(アプリケーション設定の表示やカスタム定義マクロ、カスタム定義関数および代替変数の表示や変更など)	● ユーザー名 ● 日付と時刻 ● オペレーティング・システムのリソース、ライセンス情報およびシステム規模の構成の取得の要求 ● グローバルな値の取得および設定 ● 代替変数を変更した場合- 代替変数のリストおよび設定の要求 ● カスタム定義マクロを変更した場合- カスタム定義マクロのリストおよび削除の要求 ● アプリケーション・ログの取得および削除の要求
データベース・レベルで実行されるアクション(ルール・ファイル、アウトライン、レポートまたは計算スクリプトの作成など)	● ユーザー名 ● 日付と時刻 ● クライアント設定、ユーザーおよびグループ情報の取得の要求 ● アプリケーションおよびデータベースのリストの要求 ● アクセス情報の取得の要求 ● アーティファクトのロックの要求 ● アーティファクトの戻し

Essbase サーバー・ログの例

次の `essbase.log` の例は、Essbase サーバーの開始時に書き込まれるエントリを示しています。最初に、Sample アプリケーションと Basic データベースがロードされます。このログ・ファイルには、アプリケーションやデータベースがロードされた時刻、オペレーティング・システムによってアプリケーションに割り当てられたプロセス ID、およびセキュリティ認証モジュールの起動に関する情報が含まれています。

注： 正常なシャットダウンを実行できない場合は、このプロセス ID を使用してアプリケーションを不正に停止できます。766 ページの「アプリケーションの誤った停止」を参照してください。

```
[Tue Nov 06 07:54:16 2001]Local/ESSBASE0///Info(1054014)

Database Basic loaded


[Tue Nov 06 07:54:16 2001]Local/ESSBASE0///Info(1051061)

Application Sample loaded - connection established


[Tue Nov 06 07:54:16 2001]Local/ESSBASE0///Info(1054027)

Application [Sample] started with process id [1300]


[Tue Nov 06 07:54:18 2001]Local/ESSBASE0///Info(1054014)

Database Basic loaded


[Tue Nov 06 07:54:23 2001]Local/ESSBASE0///Info(1051134)

External Authentication Module: [LDAP] enabled


[Tue Nov 06 07:54:23 2001]Local/ESSBASE0///Info(1051051)

Essbase Server - started
```

次のログは、単一のエラーを示しています。管理ユーザーが、Essbase サーバー上にすでに存在する名前を使用して、アプリケーションの名前を変更しようとしました。このログ・ファイルには、ユーザー名、エラーの時刻、および失敗してエラーが発生した操作に関する情報が含まれています。

```
[Tue Nov 06 08:00:04 2001]Local/ESSBASE0///Info(1051001)

Received client request: Rename Application (from user admin)
```

```
[Tue Nov 06 08:00:04 2001]Local/ESSBASE0///Error(1051031)

Application Testing already exists


[Tue Nov 06 08:00:04 2001]Local/ESSBASE0///Warning(1051003)

Error 1051031 processing request [Rename Application] - disconnecting
```

次のログは、シャットダウンを示しています。このログ・ファイルには、シャットダウンしたアプリケーションの名前と、シャットダウンの時刻に関する情報が含まれています。

```
[Tue Nov 06 08:00:46 2001]Local/ESSBASE0///Info(1054005)

Shutting down application Sample


[Tue Nov 06 08:00:52 2001]Local/ESSBASE0///Info(1051052)

Essbase Server - finished
```

アプリケーション・ログのコンテンツ

appname.log ファイル内の情報は、次の項目の評価に役立ちます:

- 特定の操作を実行したユーザー
- 操作が実行された日時
- 操作が実行または試行されたときに発生したエラー
- 最適化に役立つ次元およびメンバーの詳細
- Essbase のインスタンス上にアーティファクトが存在する場合は、操作を実行するために使用されたアーティファクトの名前(計算やデータ・ロードを実行するために使用された計算スクリプトまたはロード・ファイルなど)

表 141 は、記録されるアクションのタイプと、ログ・メッセージに含まれる情報を示しています。

表 141 アプリケーション・ログのコンテンツ

アクションのタイプ	記録される情報
データベース・レベルで実行されたアクション(データのロード、データの消去、データの計算など)	• ユーザー名 • 日付と時刻 • アプリケーションおよびデータベースの名前と時間 • アプリケーションを開始した場合- Java モジュールのロード、データベース情報とアプリケーション情報の読取りおよび書込み • データベースをロードした場合- 次元サイズ、動的計算メンバー、ブロック、キャッシュ・サイズ、インデックス・ページ・サイズおよび I/O 情報に関する情報 • データベースを開始した場合- アプリケーションおよびデータベースの設定情報(空き領域情報の読取り、データベース・パラメータの書込み、状況情報の取得、アプリケーションおよびデータベースの定義情報の書込み、データベース・ボリュームの取得、データベース・マッピングの書込みなど) • ロードした場合- ロード・コマンド、並列データ・ロード情報、更新されたセル、ロードにかかった時間、およびルール・ファイルとデータ・ファイルの名前 **注：** Essbase によってロード・ルール名が提供されるのは、要求を発行しているクライアント(たとえば、Administration Services)が Essbase と同じリリース・レベル(たとえば、リリース 7.0)にある場合のみです。 • 計算した場合- 計算を実行するために使用した計算スクリプトの名前 • レポートした場合- レポート・スクリプトの名前
アウトライン・レベルで実行されたアクション(再構築など)	• ユーザー名 • 日付と時刻 • 次元サイズ、動的計算メンバー、ブロック、キャッシュ・サイズ、インデックス・ページ・サイズ、I/O 情報および再構築にかかった時間
スプレッドシート・レベルで実行されたアクション(ロックおよび送信など)	• ユーザー名 • 日付と時刻 • 実行されたアクション

アプリケーション・ログの例

次の項目では、標準的な起動およびシャットダウンを行った場合のアプリケーション・ログのエントリの例や、エラー発生時に記録されるメッセージの例などを示しています。

例 1: アプリケーション・ログの起動メッセージ

次のログの例は、Essbase サーバーの開始時に `appname.log` に書き込まれるすべてのエントリを示しています。このログ・ファイルには、アプリケーションが開始された時刻、アプリケーションおよびデータベース情報が読み取られた時刻と書き込まれた時刻、ログイン要求に対するアプリケーションの準備ができた時刻、データベースがロードされた時刻などの情報が含まれています。

```
[Tue Nov 06 08:47:14 2001]Local/Sample///Info(1002035)

Starting Essbase Server - Application [Sample]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1200480)

Loaded and initialized JVM module


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019008)

Reading Application Definition For [Sample]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019009)

Reading Database Definition For [Basic]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019021)

Reading Database Mapping For [Sample]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019010)

Writing Application Definition For [Sample]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019011)

Writing Database Definition For [Basic]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019022)

Writing Database Mapping For [Sample]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1013202)
```

```
Waiting for Login Requests

[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1013205)

Received Command [Load Database]

[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019018)

Writing Parameters For Database [Basic]

[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019017)

Reading Parameters For Database [Basic]
```

Essbase サーバーでは、開始後、次元サイズや動的計算情報などの、アウトライン内の次元とメンバーに関する情報がアプリケーション・ログに書き込まれます。その例を次に示します:

```
[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019012)

Reading Outline For Database [Basic]

[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1007043)

Declared Dimension Sizes = [20 17 23 25 5 3 5 3 15 8 6 ]

[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1007042)

Actual Dimension Sizes = [20 14 20 25 4 3 5 3 15 8 5 ]

[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1007125)

The number of Dynamic Calc Non-Store Members = [8 6 0 0 2 ]

[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1007126)
```

```
The number of Dynamic Calc Store Members = [0 0 0 0 0 ]
```

Essbase サーバーでは次に、ブロック・サイズ、宣言された見込みブロックの数、計算を実行するために必要なブロックの数(この情報を使用すると、動的計算とタグ付けされた疎次元のメンバーの取得パフォーマンスを見積ることができます)などの、データベース内のブロックに関する情報がアプリケーション・ログに書き込まれます:

```
[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1007127)

The logical block size is [1120]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1010008)

Maximum Declared Blocks is [575] with data block size of [1700]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1010007)

Maximum Actual Possible Blocks is [500] with data block size of [192]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1200481)

Formula for member [Opening Inventory] will be executed in [CELL] mode


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1012710)

   Essbase needs to retrieve [1] Essbase Kernel blocks in order to calculate the top
dynamically calculated block.
```

Essbase サーバーでは次に、データベースごとに設定されたキャッシュに関する情報がアプリケーション・ログに書き込まれます:

```
[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1012736)

The Dyn.Calc.Cache for database [Basic] can hold a maximum of [2340] blocks.


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1012737)
```

```
    The Dyn.Calc.Cache for database [Basic], when full, will result in [allocation
from non-Dyn.Calc.Cache memory].


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019018)

    Writing Parameters For Database [Basic]


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019017)

    Reading Parameters For Database [Basic]


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1070013)

    Index cache size ==> [1048576] bytes, [1024] index pages.


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1070014)

    Index page size ==> [8192] bytes.


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1070081)

    Using buffered I/O for the index and data files.


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1070083)

    Using waited I/O for the index and data files.


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019019)

    Reading Data File Free Space Information For Database [Basic]...


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1006025)

    Data cache size ==> [3145728] bytes, [2048] data pages
```

```
[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1006026)

Data file cache size ==> [0] bytes, [0] data file pages
```

起動時に記録される最後のメッセージは、一般的なデータベース情報です:

```
[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1013205)

Received Command [Get Database Volumes]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1013205)

Received Command [Set Database State]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019018)

Writing Parameters For Database [Basic]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019018)

Writing Parameters For Database [Basic]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1013205)

Received Command [Get Database State]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1013205)

Received Command [Get Database Info]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1013205)

Received Command [SetApplicationState]


[Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019010)
```

```
    Writing Application Definition For [Sample]


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019011)

    Writing Database Definition For [Basic]


    [Tue Nov 06 08:47:15 2001]Local/Sample///Info(1019022)

    Writing Database Mapping For [Sample]
```

例 2: アプリケーション・ログ内のエラー

次の例は、単一のエラーを示しています。データ・ロード・ファイル内に不明なメンバーが見つかり、この不明なメンバーが存在したためにロードが失敗しました。最初にデータ・ロードに対する要求が表示され、次にエラー・メッセージが表示され、最後に、データ・ロードによって変更されたデータ値とデータ・ロードの経過時間を記述した情報メッセージが表示されます。

```
   [Tue Nov 06 08:49:52 2001]Local/Sample///Info(1013210)
User [admin] set active on database [Basic]

[Tue Nov 06 08:49:52 2001]Local/Sample/Basic/admin/Info(1013091)
Received Command [DataLoad] from user [admin]

[Tue Nov 06 08:49:52 2001]Local/Sample/Basic/admin/Info(1003040)
Parallel dataload enabled: [1] block prepare threads,
[1] block write threads.

[Tue Nov 06 08:49:52 2001]Local/Sample/Basic/admin/Error(1003000)
Unknown Item [500-10] in Data Load, [0] Records Completed

[Tue Nov 06 08:49:52 2001]Local/Sample/Basic/admin/Warning(1003035)
No data values modified by load of this data file

[Tue Nov 06 08:49:52 2001]Local/Sample/Basic/admin/Info(1003024)
Data Load Elapsed Time : [0.11] seconds

[Tue Nov 06 08:49:52 2001]Local/Sample/Basic/admin/Info(1019018)
Writing Parameters For Database [Basic]
```

例 3: アプリケーション・ログ内のシャットダウン・メッセージ

次のメッセージは、Essbase サーバーで正常なシャットダウンが実行されると記録されます。最初に、データベースに関する情報が取得されます。次に、データベー

スがアンロードされ、空きスペース情報が書き込まれて、サーバーがシャット・ダウンします。

```
   [Tue Nov 06 08:50:26 2001]Local/Sample///Info(1013214)
Clear Active on User [admin] Instance [1]

[Tue Nov 06 08:50:34 2001]Local/Sample///Info(1013205)
Received Command [Get Database Info]

[Tue Nov 06 08:50:34 2001]Local/Sample///Info(1013205)
Received Command [Get Database State]

[Tue Nov 06 08:50:34 2001]Local/Sample///Info(1013205)
Received Command [Get Database Volumes]

[Tue Nov 06 08:50:34 2001]Local/Sample///Info(1013205)
Received Command [Get Database State]

[Tue Nov 06 08:50:34 2001]Local/Sample///Info(1013205)
Received Command [Get Database Volumes]

[Tue Nov 06 08:50:34 2001]Local/Sample///Info(1013205)
Received Command [Unload Database]

[Tue Nov 06 08:50:34 2001]Local/Sample///Info(1019018)
Writing Parameters For Database [Basic]

[Tue Nov 06 08:50:34 2001]Local/Sample///Info(1019020)
Writing Free Space Information For Database [Basic]

[Tue Nov 06 08:50:34 2001]Local/Sample///Info(1013207)
RECEIVED SHUTDOWN COMMAND - SERVER TERMINATING
```

Essbase サーバー・ログおよびアプリケーション・ログのメッセージのカテゴリ

表 142 は、最初の列に示すエラー番号の範囲ごとのエラー・メッセージのカテゴリを示しています。エラー・メッセージを受信した場合は、この表を使用して、そのエラーに関連した Essbase コンポーネントを特定します。『Oracle Essbase エラー・メッセージ・リファレンス』を参照してください。

表 142　エラー・メッセージのカテゴリ

エラー・メッセージ番号の範囲	エラーを生成したコンポーネント
1001000-1001999	レポート・ライター
1002000-1002999	サーバー一般
1003000-1003999	データ・ロード
1004000-1004999	サーバー一般

エラー・メッセージ番号の範囲	エラーを生成したコンポーネント
1005000-1005999	バックアップ、エクスポート、検証
1006000-1006999	データ・キャッシュ
1007000-1007999	アウトライン再構築
1008000-1008999	システム・コール、ポータブル層、ASD、エージェント
1009000-1009999	ASCII データの復元
1010000-1010999	内部(ブロックの番号付け)
1011000-1011999	内部(ユーティリティ)
1012000-1012999	計算機
1013000-1013999	リクエスタ
1014000-1014999	ロック・マネージャ
1015000-1015999	別名テーブル
1016000-1016999	レポート・ライター
1017000-1017999	通貨
1018000-1018999	現在は使用されていません
1019000-1019999	データベース・アーティファクト
1020000-102999	スプレッドシート・エクストラクタ
1021000-1021999	Oracle Essbase SQL インタフェース
1022000-1022999	セキュリティ
1023000-1023999	パーティション化
1024000-1024999	クエリー・エクストラクタ
1030000-1030999	API
1040000-1040999	ネットワーク一般
1041000-1041999	ネットワーク- 名前付きパイプ
1042000-1042999	ネットワーク- TCP
1043000-1049999	現在は使用されていません
1050000-1055999	エージェント
1056000-1059999	現在は使用されていません
1060000-1060999	アウトライン API
106100-1069999	現在は使用されていません

エラー・メッセージ番号の範囲	エラーを生成したコンポーネント
1070000-1070999	インデックス・マネージャ
1071000-1079999	現在は使用されていません
1080000-1080099	トランザクション・マネージャ
1081000-1089999	現在は使用されていません
1090000-1099999	ルール・ファイルの処理
1010000-1019999	現在は使用されていません
1100000-1100999	現在は使用されていません
1110000-1119999	Oracle Hyperion Web Analysis
1120000-1129999	グリッド API
1130000-1139999	その他
1140000-1149999	リンク・レポート・オブジェクト(LRO)
1150000-1159999	アウトライン同期
1160000-1169999	アウトライン変更記録
1170000-1179999	属性
1180000-1189999	ショーケース
1190000-1199999	Integration Services
1200000-1200999	計算機フレームワーク

Essbase サーバー・ログおよびアプリケーション・ログの使用

次の項目では、サーバー・ログおよびアプリケーション・ログで実行可能なアクションについて説明します。

サーバー・ログおよびアプリケーション・ログの詳細は、801 ページの「Essbase サーバー・ログおよびアプリケーション・ログ」を参照してください。特定のエラー・メッセージの詳細は、『Oracle Essbase エラー・メッセージ・リファレンス』を参照してください。

また、Administration Services を使用してログを表示できます。Oracle Essbase Administration Services Online Help の「ログ・ビューアについて」を参照してください。

Essbase サーバー・ログおよびアプリケーション・ログの最大ログ・ファイル・サイズの設定

essbase.cfg ファイル内の次の設定を使用すると、Essbase サーバー(エージェント)およびアプリケーション・ログ・ファイルの最大ログ・ファイル・サイズを指定できます:

- AGTMAXLOGFILESIZE
- APPMAXLOGFILESIZE

デフォルトの最大ログ・ファイル・サイズは 2GB です。

『Oracle Essbase テクニカル・リファレンス』を参照してください。

Essbase サーバー・ウィンドウに記録または表示される Essbase サーバー・メッセージのタイプの設定

Essbase サーバー・メッセージを Essbase サーバー・ログ(EPM_ORACLE_INSTANCE/diagnostics/logs/essbase/essbase_0/essbase.log)、または Essbase サーバー・ウィンドウ、あるいはその両方に書き込むかどうかを指定できます。

ログ・レベル:

- 情報メッセージ(ユーザー・アクションの通知やアプリケーションまたはデータベースの詳細など)

  次の例は、admin ユーザーがログアウトしたことを示しています。

```
[Sun Oct 21 16:00:55 2001]Local/ESSBASE0///Info(1051037)
Logging out user admin, active for 144 minutes
```

- 警告メッセージ(完了されなかった操作の通知など)

  多くの場合は、エラーに続いて警告が発生します。次の例は、ログに含まれている以前のエラーのために、名前変更操作が完了しなかったことを示しています。

```
[Fri Nov 02 13:38:14 2001]Local/ESSBASE0///Warning(1051003)
Error 1051031 processing request [Rename Application] - disconnecting
```

- エラー・メッセージ(Essbase サーバーで実行できないアクションを試みた場合など)

  次の例は、そのアプリケーション名がすでに存在しているために、名前変更操作が失敗したことを示しています。

```
[Fri Nov 02 13:38:14 2001]Local/ESSBASE0///Error(1051031)
Application Testing already exists
```

- OPMN ping メッセージなどのデバッグ・メッセージ。

表 143 には、Essbase サーバーが Essbase サーバー・ログに書き込むメッセージのタイプ、または Essbase サーバー・ウィンドウに表示するメッセージのタイプを決めるために、essbase.cfg ファイルに指定する構成設定がリストされています。essbase.cfg 設定を変更した場合は、Essbase サーバーを再起動して変更を適用します。

表 143　Essbase サーバー・メッセージの構成設定のリスト

設定	定義
AGENTLOGMESSAGELEVEL	Essbase サーバー・ログに書き込まれるメッセージのレベルを決定する essbase.cfg の設定。
AGENTDISPLAYMESSAGELEVEL	Essbase サーバー・ウィンドウに書き込まれるメッセージのレベルを決定する essbase.cfg の設定。
PORTUSAGELOGINTERVAL	使用中のポート数を確認する頻度を指定する essbase.cfg の設定。

記録するアプリケーション・メッセージのタイプの設定

デフォルトでは、アプリケーション・ログには、次のタイプのメッセージがリストされます:

- 情報メッセージ。これは、Essbase サーバーによって実行された定期的なアクションを詳述するメッセージです

  次の例では、Essbase サーバーがデータ・ロードにかかった時間をアプリケーション・ログに書き込んでいます:

```
     [Fri Nov 02 13:04:15 2001]
Local/Sample/Basic/admin/Info(1003024)
Data Load Elapsed Time : [3.014] seconds
```

- 警告メッセージ。これは、Essbase サーバーによって重大でない、と判断された状況をリストするメッセージです

  次の例では、データ・ロード中に変更されたデータ値がなかったことを示すステートメントが Essbase サーバーによってアプリケーション・ログに書き込まれています:

```
     [Fri Nov 02 12:43:44 2001]
Local/Sample/Basic/admin/Warning(1003035)
No data values modified by load of this data file
```

- エラー・メッセージ。これは、タスクの実行中に発生したエラーを記述するメッセージです

  エラー・メッセージには、ファイルが正しくロードされないなどの重大なエラーから、Essbase サーバーのクラッシュを引き起こすディスク・スペース・エラーなどの非常に重大なエラーまでの幅があります。次の例では、アウトライン内のすべての次元が指定される前にデータ値が見つかったことを示すステートメントが、Essbase サーバーによってアプリケーション・ログ・ファイルに書き込まれています:

```
        [Fri Nov 02 12:53:32 2001]
Local/Sample/Basic/admin/Error(1003007)
Data Value [678] Encountered Before All Dimensions Selected, [2] Records
Completed
```

表 144 は、Essbase サーバーによってアプリケーション・ログに書き込まれるメッセージのタイプを決定するために、essbase.cfg ファイルおよび計算スクリプトで使用できるコマンドで指定する設定を示しています。essbase.cfg 設定を変更した場合は、Essbase サーバーを再起動して変更を適用します。

表 144　アプリケーション・ログ・メッセージの構成設定のリスト

設定	定義
LOGMESSAGELEVEL	Essbase サーバーで、すべてのメッセージ、警告メッセージまたはエラー・メッセージをアプリケーション・ログに書き込むかどうかを決定する essbase.cfg の設定
TIMINGMESSAGES	Essbase サーバーで、各スプレッドシートとレポートのクエリー実行時間をアプリケーション・ログに書き込むかどうかを決定する essbase.cfg の設定
SSLUNKNOWN	Essbase サーバーで、スプレッドシート操作中に不明なメンバー名が検出された場合にエラー・メッセージをアプリケーション・ログに書き込むかどうかを決定する essbase.cfg の設定
SET MSG	Essbase サーバーで、計算スクリプトの実行中に次のアイテムをアプリケーション・ログに書き込むかどうかを決定する計算スクリプト設定: ● 計算統計の要約 ● 計算統計の詳細 ● すべてのメッセージ、エラー・メッセージまたはメッセージなし

Essbase サーバー・ログおよびアプリケーション・ログの表示

ログを表示しているときは、スナップショットが表示されています。ログの更新されたバージョンを表示するには、ログを閉じてから、再度開きます。特定の日付から現在までのログを表示するか、またはログ全体を表示できます。

Essbase サーバー・ログを表示するには、管理者権限が必要です。また、アプリケーション・ログを表示するには、アプリケーション・マネージャ以上の権限が必要です。

➤ サーバー・ログまたはアプリケーション・ログを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ログの表示	Oracle Essbase Administration Services Online Help
任意のテキスト・エディタ		(テキスト・エディタのドキュメンテーションを参照)

Administration Services における Essbase サーバー・ログ・レベルおよびアプリケーション・ログ・レベルの変更

Essbase のログ・レベルは、Administration Services コンソールで、サーバー・レベルとアプリケーション・レベルに変更できます。Oracle Essbase Administration Services Online Help のログ・レベルの変更に関する項を参照してください。

システムおよびアプリケーションのメッセージ・レベルの表示

message_level 文法を使用した MaxL ステートメントで、現在のシステム全体およびアプリケーションのメッセージ・レベルを表示できます。次の 2 つの列があるテーブルが出力されます: `component` および `message_level`。

- システムのメッセージ・レベルを表示するには、次の MaxL ステートメントを使用します:

  ```
  display system message_level;
  ```

  出力:

  ```
       component      message_level
  +-------------------+-------------------
  system        info
  lease_manager     error
  ```

- アプリケーションのメッセージ・レベルを表示するには、次の MaxL ステートメントを使用します:

  ```
  display application
  appname
   message_level;
  ```

  たとえば、アプリケーションが Sample であるとします。出力:

  ```
       component      message_level
  +-------------------+-------------------
  Sample        info
  lease_manager     error
  ```

Essbase サーバー・ログおよびアプリケーション・ログの即時消去

サーバー・ログおよびアプリケーション・ログは、サーバー上のディスク・スペースを使用します。場合によっては、ログが大きくなりすぎる前に、ログからエントリを消去することが必要な場合があります。サーバー・ログを消去すると、ログ内のすべてのエントリが削除されますが、そのサーバー・ログ自体は削除されません。アプリケーション・ログを消去すると、アプリケーション・ログ内のす

べてのエントリが削除され、そのアプリケーション・ログも削除されます。消去する前に、各ログをバックアップしてください。

サーバー・ログおよびアプリケーション・ログを消去するには、管理者権限が必要です。

➤ サーバー・ログまたはアプリケーション・ログをただちに消去するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ログの削除	Oracle Essbase Administration Services Online Help
MaxL	Essbase サーバー・ログの消去: **alter system clear logfile;** アプリケーション・ログの消去: **alter application appname clear logfile;**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	サーバー・ログの消去: DELETELOG ""; アプリケーション・ログの削除: DELETELOG "appname";	『Oracle Essbase テクニカル・リファレンス』

注: サーバーまたはアプリケーションの再起動のたびにサーバー・ログまたはアプリケーション・ログを消去するには、819 ページの「再起動時の Essbase サーバー・ログおよびアプリケーション・ログの消去」を参照してください。Essbase サーバー・ログまたはアプリケーション・ログに記録されるアイテムを制限することによって領域を節約するには、815 ページの「Essbase サーバー・ウィンドウに記録または表示される Essbase サーバー・メッセージのタイプの設定」または816 ページの「記録するアプリケーション・メッセージのタイプの設定」を参照してください。

再起動時の Essbase サーバー・ログおよびアプリケーション・ログの消去

デフォルトでは、Essbase サーバーにより、サーバー・ログおよびアプリケーション・ログの最後にメッセージが追加されます。CLEARLOGFILE 構成設定を使用して、サーバーが再起動されるたびに Essbase サーバー・ログを、またはアプリケーションが再起動されるたびにアプリケーション・ログを消去するように Essbase サーバーを設定できます。`essbase.cfg` ファイルを変更した場合は、その変更を適用するために Essbase サーバーを再起動します。

注： サーバーまたはアプリケーションを再起動することなく、サーバー・ログまたはアプリケーション・ログを消去するには、818 ページの「Essbase サーバー・ログおよびアプリケーション・ログの即時消去」を参照してください。Essbase サーバー・ログまたはアプリケーション・ログに記録されるアイテムを制限することによって領域を節約するには、815 ページの「Essbase サーバー・ウィンドウに記録または表示される Essbase サーバー・メッセージのタイプの設定」または 816 ページの「記録するアプリケーション・メッセージのタイプの設定」を参照してください。

Essbase サーバー・ログおよびアプリケーション・ログ内の区切り記号の設定

サーバー・ログおよびアプリケーション・ログのログ・エントリを区切るために使用される記号を変更できます。区切り記号を変更すると、Essbase サーバー・ウィンドウに記録されるメッセージにも影響を与えます。デフォルトでは、Essbase サーバーにより、スペースを使用してログ内のフィールドが区切られます。その例を次に示します:

```
   [Thu May 10 20:14:46 2001]Local/ESSBASE0///Info(1051051)
Essbase Server - started
```

サーバー・ログおよびアプリケーション・ログ内のエントリを区切るために、チルダ、キャレット、コロン、アンパサンドまたはアスタリスクも使用できます。可能であれば、データ、アプリケーションまたはデータベースではあまり使用されない区切り記号を選択してください。次の例は、チルダ(~)で区切られたログ・エントリを示しています:

```
   Thu~May~10~20:16:13~2005~Local~ESSBASE0~~~Info~(1051051)~ \\ Oracle Essbase Server
- started
```

注： スペース以外の区切り記号設定を選択した場合は、ログ・ビューアでログ・エントリを日付順にソートできません。

表 145 は、Essbase サーバーのサーバー・ログおよびアプリケーション・ログで使用する区切り記号を決定するために、essbase.cfg ファイルで指定する設定を示しています。essbase.cfg 設定を変更した場合は、Essbase サーバーを再起動して変更を適用します。

表 145　ログ・メッセージ区切り記号の構成設定のリスト

設定	説明
DELIMITEDMSG	essbase.cfg の設定。TRUE に設定すると、サーバーおよびアプリケーション・ログ内の各フィールド間にチルド(~)が追加されます。 別の区切り記号を指定するには、DELIMITER 設定を使用します。

設定	説明
DELIMITER	essbase.cfg の設定。サーバーおよびアプリケーション・ログ内の各フィールド間で使用する区切り記号を指定します。 Essbase サーバーでは、ログの区切り記号として、デフォルトの区切り記号であるチルダ(~)、キャレット(^)、コロン(:)、アンパサンド(&)およびアスタリスク(*)の文字を使用できます。 DELIMTER 設定は、DELIMITEDMSG が TRUE に設定されている場合にのみ有効です。

ログ・アナライザによるログの分析

ログ・アナライザを使用して、Essbase サーバー・ログおよびアプリケーション・ログをフィルタ、検索および分析できます。選択または作成したフィルタに基づいて、ログの強力なグラフを表示できます。自動リフレッシュ・オプションを使用すると、ログ・ファイル情報を動的に監視できます。

➤ ログ・アナライザを使用するには、『Oracle Essbase Administration Services オンライン・ヘルプ』のログ・アナライザに関する項を参照してください。

クエリー・ログの導入

クエリー・ロギングを使用すると、Essbase 管理者は、Essbase データベースの問合せパターンを追跡できます。クエリー・ログ・ファイルでは、クエリーが Spreadsheet Add-in またはレポート・ライターのどちらから発行されているかには関係なく、データベースに対して実行されたすべてのクエリーを追跡します。クエリー・ロギングは、メンバー、特定の世代またはレベルに属するメンバーの世代またはレベル番号、およびハイブリッド分析メンバーを追跡できます。クエリー・ロギングは、世代またはレベルに属する特定の次元およびメンバーのロギングを除外する柔軟性も備えています。クエリー・ログ・ファイルの出力は XML ドキュメントであるため、ログを表示するために、XML に対応した任意のツールにログ・ファイルをインポートできます。クエリー・ログ・ファイルの構造については、ESSBASEPATH/bin ディレクトリ内の querylog.dtd を参照してください。

クエリー・ロギングは、ブロック・ストレージ・データベースにも集約ストレージ・データベースにも使用できます。

クエリー・ロギングを使用可能にするには、クエリー構成ファイルを (essbase.cfg ファイルとは別個に)作成し、そのファイルにクエリー・ロギングの実行方法を制御する構成設定を追加します。

ARBORPATH/app/appname/dbname ディレクトリ内に、クエリー・ログ構成ファイルを作成します。この構成ファイルには、dbname.cfg (dbname は、データベースの名前に一致させます)という名前を付ける必要があります。たとえば、Sample.Basic のクエリー・ログ構成ファイルは basic.cfg になります。出力のクエリー・ログ・ファイルは、デフォルトでは ARBORPATH/app/appname/dbname00001.qlg にあります。

『Oracle Essbase テクニカル・リファレンス』を参照してください。

アウトライン変更ログとその使用

アウトライン変更ログを作成するように Essbase サーバーを設定できます。これは、アウトライン変更情報をテキスト・ファイルに保存する、dbname.olg という名前のテキスト・ファイルです。ログの作成後にアウトラインに加えられた変更を参照するために、このアウトライン変更ログをいつでも確認できます。この情報は、アウトラインを以前のバージョンにロール・バックするのに役立ちます。

アウトライン変更ログは、次のディレクトリにあります:

```
EPM_ORACLE_INSTANCE
/diagnostics/logs/essbase/essbase_0/app/
appname
/
dbname
```

次の項目では、アウトライン変更ログ、およびアウトライン変更ログで実行可能なアクションについて説明します。

アウトライン変更ログのコンテンツの理解

表 146 は、アウトライン変更ログに書き込まれるアクションのタイプと、ログ・メッセージに含まれる情報を示しています。

表 146　アウトライン変更ログのコンテンツ

変更のタイプ	記録される情報
次元の追加	• 次元の名前 • 次元のタイプ(密次元または疎次元) • 次元タグ(タグが付加されている場合) • 次元の左側の兄弟の名前(存在する場合) • 次元のレベル番号 • 次元の世代番号
次元の削除	次元の名前
次元の更新	• 次元の名前 • 次元タグ • 次元のタイプ(密次元または疎次元) • レベル名の変更情報(適用可能な場合) • 世代名の変更情報(適用可能な場合)
次元の名前変更	• 変更前の次元の名前 • 変更後の次元の名前
新しい位置への次元の移動	• 次元の名前 • 移動前の場所(次元の左側の兄弟を含む) • 移動後の場所(次元の左側の兄弟を含む)

変更のタイプ	記録される情報
次元へのメンバーの追加	• 追加されたメンバーの名前 • メンバーに対する単項計算記号 • メンバーのレベル番号 • メンバーの世代番号 • メンバーのステータス(保管、共有) • メンバーの別名(適用可能な場合) • メンバーの会計タイプ(適用可能な場合) • メンバーの UDA (適用可能な場合) • メンバー用の計算式(適用可能な場合)
次元のメンバーの更新	• 更新されたメンバーの名前 • 更新されたメンバー・プロパティ
次元のメンバーの名前変更	• 変更前のメンバーの名前 • 変更後のメンバーの名前
新しい位置への次元のメンバーの移動	• 移動されたメンバーの名前 • 移動後の場所 • 移動後の場所における親と左側の兄弟の名前

アウトライン変更ログのプログラムでは、アウトライン情報を左から右に読み取ります。アウトラインを参照した場合、左側の兄弟は、新しく追加された次元またはメンバーの真上(左)にある兄弟です。このルールは、直前の次元またはメンバーが親である場合は適用されません。新しく追加(または移動)されたメンバーがその親の最初の子である場合、またはそのメンバーがアウトライン内の最初の次元である場合、アウトライン変更ログではその次元またはメンバーの古い場所が「なし」と識別されます。

アウトライン変更ログの例の確認

ユーザーがアウトラインを変更して保存すると、Essbase では、その変更情報がエントリのグループとしてアウトライン変更ログに書き込まれます。各改訂を最新のものから最も古いものまで容易に識別できるように、各エントリのグループは識別情報で開始および終了します。

次の変更ログは、アウトライン変更ログが開始されて以降、アウトラインが 1 回変更されたことを示す 1 つのレコードを示しています。最初に、アウトライン変更ログには、変更されたアプリケーションとデータベース、変更の時刻、および変更を行ったユーザーを含む変更の開始がリストされています。次に、アウトライン変更ログには、100-50 という名前のメンバーが製品次元のメンバー 100 に追加されたという内容の変更がリストされています。最後に、アウトライン変更ログには、変更されたアプリケーションとデータベース、変更の時刻、および変更を行ったユーザーを含む変更の終了がリストされています。

```
[Begin Outline Change for Sample/Basic, Sat Nov 03 12:49:31 2001, By admin]

Number of member changes for dimension "Product" : 1
```

```
Added new member "100-50" to "100" :

Left sibling - "100-40"

Status - Store Data

Added alias "Cherry Cola" to alias table "Default"

Unary calc symbol - Add

Level Number - 0

Generation Number - 3

[End Outline Change for Sample/Basic, Sat Nov 03 12:49:32 2001, By admin]
```

アウトライン変更ログの作成

デフォルトでは、Essbase サーバーでアウトライン変更ログは作成されません。アウトライン変更ログを作成するには、essbase.cfg 内の OUTLINECHANGELOG TRUE 構成設定を使用します。『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： 再構築中、アウトライン変更ログへのすべての更新が完了するまで、Essbase ではアウトライン変更情報をメモリー内に保持します。そのため、アウトライン変更ログを有効にすると、再構築のパフォーマンスに影響を与える可能性があります。935 ページの「データベースの再構築に影響する条件」を参照してください。

アウトライン変更ログの表示

アウトライン変更ログは、ログ・ファイルをテキスト・エディタで開くことによって表示できます。

アウトライン変更ログのサイズの設定

アウトライン変更ログ(dbname.olg)のデフォルト・サイズは 64,000 バイトです。最大サイズを変更するには、essbase.cfg ファイル内の OUTLINECHANGELOGFILESIZE 構成設定を使用します。『Oracle Essbase テクニカル・リファレンス』を参照してください。

アウトライン変更ログが最大サイズに達すると、Essbase では.olg ファイルのコンテンツを新しいファイルにコピーし、その拡張子を.olb に変更します。たとえ

ば、basic.olg が最大サイズに達すると、そのコンテンツは basic.olb にコピーされます。

コピー操作の後、.olg ファイルが消去され、Essbase によってこのファイルに新しいログ・エントリが書き込まれます。

.olg ファイルが最大ファイル・サイズに達するたびに、Essbase では、既存の.olb ファイルが.olg の現在のコンテンツで上書きされます。そのため、.olb ファイルからの情報の取得は、このファイルが上書きされる前に実行するようにしてください。

.olg ファイルのデフォルト、最小および最大ファイル・サイズは、.olb ファイルにも自動的に適用されます。たとえば、アウトライン変更ログの最大サイズを 2MB に変更した場合は、.olb ファイルも同じ最大サイズに自動的に設定されます。

例外ログとその使用

Essbase サーバー、アプリケーションまたはデータベースが異常シャット・ダウンすると、Essbase サーバーによって、例外ログが log0000n.xcp という名前のテキスト・ファイルとして作成される場合があります。以降のトピックでは、サーバー、アプリケーションおよびデータベースの例外ログと、それらのログに対してユーザーが実行できるアクションについて説明します。

例外ログのコンテンツ

Essbase サーバー、アプリケーションまたはデータベースが異常シャット・ダウンし、再起動できない場合は、問題のトラブルシューティングに役立つ例外ログが Essbase サーバーによって生成されます。例外ログの場所は、異常シャット・ダウンしたコンポーネントと、その時点で Essbase サーバーが取得できた情報の量によって異なります。表 148 は、異常シャットダウンのタイプごとの例外ログの場所を示しています。

表 147 は、例外ログのセクションと、各セクションに含まれる情報を示しています。シャットダウンの終了前に Essbase サーバーで情報をすべて取得できなかった場合は、後の方の一部のセクションが空白になることがあります。

表 147　例外ログ(log00001.xcp)のコンテンツ

ログのセクション	記録される情報
一般情報	• 日付と時刻 • アプリケーション名およびデータベース名 • 例外ログの場所 • プロセス・タイプ この情報を使用して、異常シャット・ダウンされたコンポーネント、およびシャット・ダウンの日時を判別します。

ログのセクション	記録される情報
コンピュータのレジスタおよびスタック情報	● 一般的なレジスタ ● 浮動小数点レジスタ ● Hex レジスタ ● スタック・トレース ● スタック・ダンプ Oracle サポートでは、このセクションを調べて、異常シャットダウンの理由の判別に役立てます。
アプリケーション全体の構成	● サーバー名およびアプリケーション名 ● アプリケーションの経過時間(アプリケーションが実行されていた時間) ● モジュールのリスト この情報を使用して、アプリケーションがすばやくシャット・ダウンしたかどうか、およびすべてのモジュールが正しいことを判別します。モジュールに関する詳細情報は、例外ログのシステム全体の構成セクションにあります。
オペレーティング・システムのリソース	● システムの日付と時刻 ● オペレーティング・システムの経過時間(オペレーティング・システムが実行されていた時間) ● リソース情報(CPU タイプ、メモリー情報、スワップ情報およびドライブ情報を含む) この情報によって、オペレーティング・システムの問題(メモリー不足など)かどうかを確認します。
システム全体の構成	● Essbase の経過時間(Essbase が実行されていた時間) ● Essbase のリリース情報 ● ネットワーク情報 ● 環境変数 ● モジュール情報(モジュール名およびリリース情報を含む) この情報を使用して、リリースが Essbase と各モジュールで同じであること、および環境変数が正しく設定されていることを確認します。
`essbase.cfg` の値	`essbase.cfg` ファイル内のすべての設定値。 この情報を使用して、Essbase サーバーが正しく構成されていることを確認します。
ライセンス情報	● シリアル番号およびライセンスの有効期限 ● 購入済ポート数および使用中のポート数 ● 使用可能な Essbase のオプション ● 使用可能な他の Oracle Hyperion 製品 この情報を使用して、Essbase の正しいオプションがインストールされていること、および十分なポートが購入されていることを確認します。

ログのセクション	記録される情報
クライアント要求アクティビティ	● サーバー名 ● アプリケーション名 ● スレッド情報(スレッド数を含む) ● 要求情報 ● 各スレッドの詳細情報(実行中のアクション、データベース、ユーザー名、開始時刻、終了時刻を含む) この情報を使用して、クライアント要求に基づくサーバーの負荷の高さを判別します。
ファイル情報	● ページ・ファイルのサイズ ● インデックス・ファイルのサイズ この情報によって、ページ・ファイルまたはインデックス・ファイルが大きすぎないかどうかを判別します。
データベース情報	この情報を使用して、データベースが正しく設定されていることを確認します。
データベース統計	この情報によって、次元情報を表示し、データベース内のデータ・ブロックの特性を確認します。

例外ログの例の検討

次の例は、例外ログを示しています。ログの最初のセクションには、アプリケーションおよびデータベースに関する一般的な情報がリストされます。この例では、Essbase サーバーがシャット・ダウンしました:

```
   ----- Exception Error Log Begin -----

Current Date & Time:  Sat Nov 24 13:25:13 2001
Process Type:     Server
Application Name:    Sample
Database Name:     Basic
Exception Log File:   C:\HYPERION\ESSBASE\log00001.xcp
Current Thread Id:   1116
Exception Code:    0xC0000005=Access Violation
Exception Flags:    0x00000000=Continuable
Exception Address:   0x002D2249
Exception Parameters:  2
Exception Parameter 0: 0x00000000=Read Violation
Exception Parameter 1: 0x0000220A (Virtual Address)
```

ログの次のセクションには、レジスタとスタックのトレース情報がリストされます。Oracle サポートは、ログのこのセクションを調べて、異常シャットダウンが発生した理由の判別に役立てます。

```
   ----- Machine Registers -----

General Registers:
 EAX=0x00000000  EBX=0x01358008  ECX=0x00002200

Control Registers:
```

```
 CS =0x0000001B  EIP=0x002D2249  Flg=0x00010202

Segment Registers:
 DS =0x00000023  ES =0x00000023  FS =0x00000038

Floating Point Registers:
 CWD=0xFFFF027F  SWD=0xFFFF0000  TWD=0xFFFFFFFF

Register Area (Hex):
 00 00 00 00 00 00 00 00 00 00

...continued hexadecimal listings...

Debug Registers:
 DR0=0x2F75C73B  DR1=0x75E07D39  DR2=0x1475FF16
 DR3=0x00000000  DR6=0x0000E00B  DR7=0x00000000
----- Stack -----

Stack Trace:
  0: 0x002D2249
  1: 0x002D202D
...continued stack trace listings...

Stack Dump (Hex):
 (Stack dump truncated from 1397 to 1024 bytes.)
 0x0012EA2C:  00000000 01358008 01358008 FFFFFFFF
 0x0012EA3C:  00002200 002D728C 0012EC6C 002D202D
...continued stack dump listings...
```

次のセクションには、アプリケーション情報がリストされます:

```
   ----- Application-Wide Configuration -----

Server Name:      ASPEN
Application Name:    Sample
Elapsed App Time:    00:00:01:28
Module Count:      6
Module  0:       0x00241000 = C:\HYPERION\ESSBASE\BIN\ESSSEC.DLL
Module  1:       0x002C1000 = C:\HYPERION\ESSBASE\BIN\ESSNET.DLL
...continued module listings...
```

ログの次のセクションには、オペレーティング・システムの情報がリストされます。使用可能なメモリー量、使用されているスワップ・スペースの量および各ドライブで使用可能なメモリー量を判別できます:

```
   ----- Operating System Resources -----

System Date & Time:   Sat Nov 24 13:25:13 2001
Elapsed OS Time:    02:03:03:10
OS Name & Version:    Windows NT 5.00
CPU Count:       1
CPU Type:        Pentium
Total Physical Memory: 327024 KB (334872576)
Free Physical Memory:  155760 KB (159498240)
```

```
Used Physical Memory:  171264 KB (175374336)
Swap Flags:
 Enabled:      Y
 Disabled:     N
 File Found:     Y
 Denied:       N
Swap file(s):     C:\pagefile.sys
Total Swap Space:    467192 KB (478404608)
Free Swap Space:    421528 KB (431644672)
Used Swap Space:    45664 KB (46759936)
Total Drives:     5
Current Drive:     3
Drive  1:
 Drive Name:     C
 Volume Label:
 Drive Type:     Fixed
 File System:    NTFS
 Total Drive Space:  11778448 KB
 Free Drive Space:  8592548 KB
 Used Drive Space:  3185900 KB
...continued drive listings...
```

ログの次のセクションには、システム構成情報(パスや essbase.cfg の設定など)がリストされます:

```
   ----- System-Wide Configuration -----

Elapsed Essbase Time:  00:00:01:33
Essbase Version:    6.2.0
Essbase Description:  Ess62P0B128
Network Type:     Windows Sockets
Environment Variable:  ARBORPATH = C:\HYPERION\ESSBASE
Environment Variable:  ARBORMSGPATH = C:\HYPERION\ESSBASE\bin
Module Count:     13
Module  0:
 Module Name:    C:\HYPERION\ESSBASE\BIN\ESSUTL.DLL
 Module Version:   6.2.0.1
 Module Description: Ess62P0B128.1
 Module Use Count:  5
...continued module listings...

----- ESSBASE.CFG Configuration Values -----

Configuration Value:  JvmModuleLocation = C:\Hyperion\Essbase\java\jre13\bin\hotspot
\jvm.dll
Configuration Value:  AuthenticationModule = LDAP essldap.dll x
Configuration Value:  OUTLINECHANGELOG = TRUE
```

ログの次のセクションには、ライセンス情報(シリアル番号など)、Essbase オプション(購入済ポートなど)および購入済の Oracle Hyperion 製品がリストされます:

```
   ----- License Information -----

Serial Number:    xxx
```

```
License Expiry Date:
Port Count:      10
Ports In Use Count:   0
Limited Use Version:  N
Read-Only SS:     N

...continued Essbase options and Hyperion product listings...
```

ログの次のセクションには、クライアントのアクティビティ(Administration Services を使用したデータベースの表示など)がリストされます:

```
   ----- Client Request Activity -----

Server Name:      ASPEN
Application Name:    Sample
Total Request Threads: 5
Avail Request Threads: 6
Total Requests:    56
Average Requests:    48.000000
Weighted Average:    7.440000
Statistics Per Minute:
 Current Requests:  48
 Minimum Requests:  48.000000
 Maximum Requests:  48.000000
Thread Count:     5
Thread Id 1444:
 Request Name:     List Objects
 Database Name:     Basic
 User Name:      admin
 Start Time:      Sat Nov 24 13:24:37 2001
 End Time:       Sat Nov 24 13:24:37 2001
...continued thread listings...

----- Exception Error Log End -----
```

例外ログの表示

例外変更ログは、そのログを任意のテキスト・エディタで開くことによって表示できます。例外ログの場所は、異常シャット・ダウンしたコンポーネントと、その時点で Essbase サーバーが取得できた情報の量によって異なります。

表 148 は、例外ログの場所を示しています。

表 148　例外ログの場所

シャット・ダウンされたコンポーネント	例外ログの場所
Essbase サーバー	ログは ARBORPATH ディレクトリにあります。たとえば、次のようになります: EPM_ORACLE_INSTANCE /EssbaseServer/log00001.xcp
アプリケーション	アプリケーション名が不明な場合、ログは ARBORPATH/app ディレクトリにあります。例: EPM_ORACLE_INSTANCE /EssbaseServer/essbaseserver1/app/log00001.xcp アプリケーション名が既知の場合、ログはアプリケーション・ディレクトリにあります。たとえば、Sample アプリケーションが異常シャット・ダウンした場合は、次のようになります: EPM_ORACLE_INSTANCE /EssbaseServer/essbaseserver1/app/Sample/log00001.xcp
データベース	データベース名が不明な場合、ログはアプリケーション・ディレクトリにあります。たとえば、Basic データベースが異常シャット・ダウンした場合は、次のようになります: EPM_ORACLE_INSTANCE /EssbaseServer/essbaseserver1/app/sample/log00001.xcp データベース名が既知の場合、ログはデータベース・ディレクトリにあります。たとえば、Basic データベースが異常シャット・ダウンした場合は、次のようになります: EPM_ORACLE_INSTANCE /EssbaseServer/essbaseserver1/app/sample/basic/log00001.xcp

既存のログの上書きまたは保持

デフォルトでは、サーバーが異常シャット・ダウンするたびに、Essbase サーバーで 1 つ以上の例外ログが作成されます。後続の例外ログには、連続した番号が付けられます。たとえば、log00001.xcp が存在する場合、次のログには log00002.xcp という名前が付けられます。

essbase.cfg 内の EXCEPTIONLOGOVERWRITE TRUE 構成設定を使用すると、新しいログ・ファイルを作成するかわりに既存の例外ログ・ファイルを上書きできますが、FALSE のデフォルト設定を使用することをお薦めします。異常シャットダウンによって複数の例外ログ・ファイルが作成される可能性がありますが、一般には、シャットダウン中に作成された最初のログ・ファイルに最も多くの情報が含まれています。『Oracle Essbase テクニカル・リファレンス』を参照してください。

次元構築およびデータ・ロードのエラー・ログとその使用

Essbase サーバーでは、次元構築またはデータ・ロード中に発生したエラーがエラー・ログに書き込まれます。Essbase サーバーでエラー用に選択されるログは、次元構築やデータ・ロードなどの実行する操作と、Administration Services または MaxL の使用などの実行方法によって異なります。以降のトピックでは、次元構築およびデータ・ロードのエラーの場所と、次元構築およびデータ・ロードのエラー・ログに対して実行できるアクションについて説明します。

次元構築およびデータ・ロードのエラー・ログとその表示

dataload.err のログには、次元構築またはデータ・ロード中に発生したエラーが含まれています。これらのログにはまた、ロードに失敗したレコードも含まれています。エラーを修正した後、これらのログを再ロードできます。833 ページの「次元構築およびデータ・ロードのエラー・ログのロード」を参照してください。

Essbase サーバーでは、次元構築またはデータ・ロード中に発生したエラーは次のエラー・ログに書き込まれます:

表 149　次元構築およびデータ・ロードのエラー・ログの場所

操作	エラー・ログの場所
次元構築	EAS_HOME /client/dataload.err
ルール・ファイルを使用したデータ・ロード	EAS_HOME /client/dataload.err
ルール・ファイルを使用しないデータ・ロード	ARBORPATH /app/ appname / appname .log

注： データ・ロードまたは次元構築が失敗したが、エラー・ログが存在しない場合は、考えられる原因について 324 ページの「エラー・ログの確認」を参照してください。

次元構築およびデータ・ロードのエラー・ログの場所とファイル名を設定するには、Oracle Essbase Administration Services Online Help の「データ・ロードまたは次元構築の実行」を参照してください。

➤ 次元構築およびデータ・ロードのエラー・ログを表示するには、テキスト・エディタでこれらのログを開いてください。

次元構築およびデータ・ロードのエラー・ログの例の検討

次のデータ・ログ・エントリは、アウトラインにメンバー 500 が存在しなかったため、アウトラインにデータがロードされなかったことを示しています。

```
\\ Member 500 Not Found In Database


500  500-10  500-10-10
```

この問題を解決するために、次のいずれかのアクションを実行し、ロードを再起動できます:

- 存在しないメンバーを作成するための次元構築を実行します。318 ページの「データ・ロードまたは次元構築の実行」を参照してください。
- アウトライン・エディタで、存在しないメンバーを追加します。128 ページの「アウトラインへの次元およびメンバーの追加」を参照してください。

次の次元構築ログ・エントリは、メンバー 600-20 がメンバー 600 の親でないことを示しています。正しいテキスト・ファイルを含む正しいルール・ファイルを使用していることを確認します。このレコードはレベル(ボトムアップ)構築に対するもののように見えますが、エラー・メッセージは、Essbase サーバーが世代(トップダウン)構築を実行しようとしていることを示しています。問題を修正した後、次元構築を再起動します。

```
\\Record #2 - Incorrect Parent [600-20] For Member [600] (3307)

600-20-10  600-20  600
```

エラーの最大数の設定

dataload.err ファイルのデフォルト・サイズは 1,000 レコードです。ログがこのサイズに達すると、Essbase サーバーではそれ以上ログにエラーが書き込まれなくなります。ただし、次元構築またはデータ・ロードは続行されます。

DATAERRORLIMIT 構成設定を使用して、dataload.err ログ・ファイルに記録されるレコード数(1 から 65,000)を指定できます。essbase.cfg ファイルを変更した場合は、その変更を適用するために Essbase サーバーを再起動します。

次元構築およびデータ・ロードのエラー・ログのロード

次元構築またはデータ・ロードが失敗した場合は、分離レベルのトランザクション設定が「コミット」または「アンコミット」のどちらになっているかを判定する必要があります。トランザクション設定が「コミット」の場合は、データ・ロードを最初から再起動する必要があります。「アンコミット」の場合は、エラー・ログをロードすることによって、失敗したレコードのみをロードできます。失敗したレコードのみの再ロードは、すべてのレコードの再ロードよりはるかに高速です。

➤ エラー・ログを再ロードするには:

1 **サーバーからロードを実行する場合は、ファイル拡張子を .err から .txt に変更します。たとえば、dataload.err ファイルを dataload.txt に変更します。**

   クライアントからロードを実行する場合は、.err の拡張子のままでかまいません。

2 **次元構築またはデータ・ロードの失敗の原因となった問題を修正します。問題の修正には、アウトライン、エラー・ログ内のテキストまたはルール・ファイルの変更が必要になることがあります。**

次の条件が当てはまるかどうかを確認します:

- ルール・ファイルを検証できるか?

  297 ページの「次元構築操作の設定についての説明」を参照してください。

- データ・ソースは使用可能か?

  323 ページの「データ・ソースが使用可能かどうかの確認」を参照してください。

- サーバーは使用可能か?

  323 ページの「Essbase サーバーが使用可能かどうかの確認」を参照してください。

3 **適切なルール・ファイルを使用して、エラー・ログをロードします。**

第 17 章「データ・ロードおよび次元構築の理解」を参照してください。

49 データベース設定の管理

この章の内容

Essbase サーバー・カーネルの理解

カーネルは、データ・ロード、計算、スプレッドシートをロックして送信、パーティション化、再構築を含む、Essbase サーバーの様々な機能の基礎を提供します。カーネルは、データの読取り、キャッシュおよび書込み、トランザクションの管理、データの一貫性とデータの整合性を保証するためのトランザクション・セマンティクスの適用を行います。

カーネルには、次の機能があります:

- Essbase ファイルのディスク・ストレージおよびキャッシュの処理
- データ取得の処理
- データ更新の処理
- Essbase に関連した入力/出力機能の制御
- 再使用のための空き領域の集計
- 同時操作の管理
- サーバー・クラッシュ後のデータベースのリカバリ
- ロックの発行
- トランザクションの管理

参照:

- 836 ページの「バッファ I/O と直接 I/O の理解」
- 836 ページの「I/O アクセス・モードの表示」
- 837 ページの「I/O アクセス・モードの設定」

注： 1155 ページの「致命的なエラーの処理の理解」を参照してください。

バッファ I/O と直接 I/O の理解

直接 I/O およびバッファ I/O は、ユーザーがデータベース・レベルで設定できるアクセス・モードです。Essbase カーネルは、デフォルトではバッファ I/O(入力/出力)を使用しますが、Essbase によってサポートされているオペレーティング・システムおよびファイル・システム(Linux を除く)では直接 I/O も使用できます。Oracle Hyperion Enterprise Performance Management System 動作保証マトリックス(http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html)を参照してください。

バッファ I/O では、ファイル・システムのバッファ・キャッシュが使用されます。

直接 I/O では、ファイル・システムのバッファ・キャッシュがバイパスされ、I/O を非同期でオーバーラップして実行できます。これには、次のメリットがあります:

- 応答時間の高速化。ユーザーが Essbase からのデータの戻りを待つ時間が短くなります。
- 拡張性と予測可能性。Essbase では、データベースの最適なキャッシュ・サイズをカスタマイズできます。

直接 I/O を使用するようにデータベースを設定すると、そのデータベースが次回開始されたとき、Essbase では直接 I/O を使用しようとします。データベースが開始されたときにプラットフォーム上で直接 I/O が使用できない場合、Essbase では、デフォルトであるバッファ I/O を使用します。ただし、Essbase では、I/O アクセス・モードの選択がセキュリティ・ファイルに保管され、データベースが開始されるたびにその I/O アクセス・モードを使用しようとします。

注: キャッシュ・メモリーのロックは、直接 I/O が使用されている場合にのみ使用できます。また、オペレーティング・システムの待機なし(非同期) I/O を使用する場合も、直接 I/O を使用する必要があります。

I/O アクセス・モードの表示

バッファ I/O が、すべてのデータベースでのデフォルト設定です。

➤ データベースで現在使用または設定されている I/O アクセス・モードを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	I/O アクセス・モードの選択	Oracle Essbase Administration Services Online Help
MaxL	**display database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	GETDBINFO	『Oracle Essbase テクニカル・リファレンス』

I/O アクセス・モードの設定

➤ 任意のデータベースにデフォルトのバッファ I/O のかわりに直接 I/O を使用するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	I/O アクセス・モードの選択	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM	『Oracle Essbase テクニカル・リファレンス』

また、キャッシュのサイズを増やすことが必要な場合もあります。917 ページの「キャッシュのサイズ設定」を参照してください。

カーネルのコンポーネントの理解

カーネルには、データの取得および保管に関するすべての処理を制御するコンポーネントが含まれています:

- インデックス・マネージャは、要求されたデータの場所を検索および追跡します。837 ページの「インデックス・マネージャ」を参照してください。
- インデックス・マネージャの一部である割当てマネージャは、領域を割り当て、一部のファイル操作を管理します。838 ページの「割当てマネージャ」を参照してください。
- データ・ブロック・マネージャは、インデックスによってポイントされたデータを取得し、そのデータを保管します。839 ページの「データ・ブロック・マネージャ」を参照してください。
- LRO マネージャは、LRO の取得および格納を処理します。840 ページの「LRO マネージャ」を参照してください。
- ロック・マネージャは、同時データ・アクセスを規制するためのデータ・ブロックのロックを処理します。840 ページの「ロック・マネージャ」を参照してください。
- トランザクション・マネージャは、トランザクションを追跡し、内部のコミットおよび中止操作を処理します。840 ページの「トランザクション・マネージャ」を参照してください。

インデックス・マネージャ

インデックス・マネージャは、データベース・インデックスを管理し、Essbase のデータ・ブロックをすばやく検索するための方法を提供します。インデックス・マネージャは、データベース・インデックスのどの部分をインデックス・キャッシュ内にキャッシュするかを決定し、そのインデックス・キャッシュを管理します。

表 150 は、インデックス・マネージャが制御するコンポーネントを示しています:

表 150　インデックス・マネージャのコンポーネント

コンポーネント	説明
インデックス	Essbase でデータを検索および取得するために使用する方法。インデックスという用語は、インデックス・ファイルのことも指します。
インデックス・ファイル	Essbase でデータ取得情報を保管するために使用するファイル。ディスク上に存在し、インデックス・ページを含みます。Essbase では、essxxxxx.ind(xxxxx は番号)の命名ルールを使用して、各ディスク・ボリューム上でインデックス・ファイルに漸増的に名前が付けられます。各ディスク・ボリューム上の最初のインデックス・ファイルには、ess00001.ind という名前が付けられます。
インデックス・ページ	インデックス・ファイルの下位区分。データ・ブロックを指すインデックス・エントリが保管されています。
インデックス・エントリ	データ・ブロックへのポインタ。疎次元のすべての交差に対してインデックス項目が存在します。
インデックス・キャッシュ	インデックス・ページを保持するメモリー内のバッファ。

インデックスという用語は、単一のデータベースのすべてのインデックス・ファイルを指します。インデックスは複数のボリュームにまたがることができ、単一のボリューム上に複数のインデックス・ファイルが存在できます。ディスク・スパン・パラメータを指定するには、ディスク・ボリューム設定を使用します。インデックス・キャッシュ・サイズの設定については、917 ページの「インデックス・キャッシュ・サイズの設定」を参照してください。ディスク・ボリューム設定を使用したストレージ・スペースの割当てについては、848 ページの「ディスク・ボリュームの指定」を参照してください。

割当てマネージャ

インデックス・マネージャの一部である割当てマネージャでは、次のタスクが実行されます:

- ディスク上のインデックスとデータ・ファイルの生成および拡張
- ファイルを開く操作と閉じる操作
- 新しいファイルで使用するボリュームの指定
- 使用するボリュームの順序

これらのいずれかのタスクを実行する必要がある場合、割当てマネージャでは、次のプロセスを使用してスペースを割り当てます:

1. 既存のファイル内の空きスペースの使用が試みられます。
2. 十分な空きスペースが使用できない場合は、既存のファイルの拡張が試みられます。
3. 既存のファイル内で十分な空きスペースが使用できない場合は、現在のボリューム上にファイルが作成されます。
4. ファイル拡張をできない場合、または指定されたボリューム上にファイルを作成できない場合は、次に指定されたボリュームの使用が試みられます。

5. 指定したすべてのボリュームがいっぱいの場合、エラー・メッセージが表示され、トランザクションが異常終了します。

割当てマネージャでは、ストレージのデータベース設定に基づいて、インデックスおよびデータ・ファイル用のスペースが割り当てられます。

➤ 現在の値を確認し、新しい値を設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データベース・プロパティの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM 23	『Oracle Essbase テクニカル・リファレンス』

848 ページの「ディスク・ボリュームの指定」を参照してください。

Essbase でデータが保管される方法については、847 ページの「ストレージの割当て」を参照してください。

データ・ブロック・マネージャ

データ・ブロック・マネージャは、データ・ブロックのメモリー内への読込みやデータ・ファイルへの書込み、データ圧縮の処理、およびデータ・ファイルのディスクへの書込みを行います。データ・ブロック・マネージャは、4 つのコンポーネントを制御します。次の表は、これらの各コンポーネントを示しています:

表 151 は、データ・ブロック・マネージャが制御するコンポーネントを示しています:

表 151　データ・ブロック・マネージャのコンポーネント

コンポーネント	説明
データ・ファイル	データ・ブロックを含むファイル。Essbase は、データのロード時にデータ・ファイルを生成してディスクに保管します。Essbase は、データ・ファイルに essxxxxx.pag という漸増的な名前を付けます。ここで、xxxxx は 00001 から始まる数字です。
データ・ブロック	Essbase 内のプライマリ・ストレージ・ユニット。データ・ブロックは、疎次元の特定の交差に対応する密次元のセルを表す多次元の配列です。
データ・キャッシュ	非圧縮データ・ブロックを保持するメモリー内のバッファです。
データ・ファイル・キャッシュ	圧縮されたデータ・ファイル(.pag)を保持する、メモリー内のバッファ。

データ・ファイル・キャッシュのサイズによって、一度にメモリー内に収容できるデータ・ファイル内のデータ量が決定されます。データ・キャッシュ・サイズとデータ・ブロック・サイズによって、一度にメモリー内に収容できるデータ・ブロック数が決定されます。単一のデータベースのデータ・ファイルが複数のボリュームにまたがることができ、また同じボリューム上に複数のデータベースを配置できます。918 ページの「データ・ファイル・キャッシュ・サイズの設定」

および919 ページの「データ・キャッシュ・サイズの設定」を参照してください。また、848 ページの「ディスク・ボリュームの指定」も参照してください。

LRO マネージャ

LRO を使用すると、データ・セルに、フラット・ファイルなどのオブジェクトを関連付けることができます。ユーザーは、Smart View を使用して、.lro の拡張子を持つ LRO ファイルを作成したり保管したりできます。

LRO ファイルは、データベース・ディレクトリ(ARBORPATH/app/appname/dbname。たとえば、app/Sample/Basic)に保管されます。

Essbase では、LRO に関する情報は LRO カタログに保管されます。各カタログは、独自の Essbase インデックス・ページに配置され、インデックス・ファイル内で LRO 以外の他の Essbase インデックス・ページと共存しています。

第 11 章「Essbase データへのオブジェクトのリンク」および Oracle Hyperion Smart View for Office User's Guide を参照してください。

ロック・マネージャ

ロック・マネージャでは、データ・ブロックのロックを発行して、データへの同時アクセスを制御します。

コミット・アクセスとアンコミット・アクセスの分離レベルでは、異なるロック方式が使用されます。分離レベルとロックの詳細は、第 51 章「データの整合性の確保」を参照してください。

トランザクション・マネージャ

トランザクション・マネージャでは、トランザクション操作およびコミット操作の制御、およびデータベース・リカバリの管理が行われます。

Essbase は、データを自動的にコミットします。コミットは、データのロード、計算、再構築、スプレッドシートのロックして送信操作など、データを変更するトランザクションによってトリガーされます。

Essbase でデータがコミットされる方法は、トランザクションの分離レベルがコミット・アクセスとアンコミット・アクセス(デフォルト)のどちらに設定されているかによって異なります。863 ページの「コミット・アクセス」および866 ページの「アンコミット・アクセス」を参照してください。

トランザクション・マネージャでは、トランザクションの追跡のためにトランザクション制御テーブル(dbname.tct)が保持されています。

コミット操作およびリカバリについては、873 ページの「クラッシュしたデータベースのリカバリ」を参照してください。

カーネル起動の理解

カーネル起動中のイベントのシーケンスは次のとおりです:

1. Essbase サーバーが開始された後、ユーザーがクライアントからこのサーバーに接続します。
2. ユーザーがデータベースを起動します。
3. Essbase によってデータベースがロードされます。
4. Essbase エージェントによって、このサーバーにデータベース設定が渡されます。
5. カーネルによって、初期化プロセスが開始されます。
6. カーネルによって、カーネルのコンポーネント(インデックス・マネージャ、ロック・マネージャ、LRO マネージャ、データ・ブロック・マネージャおよびトランザクション・マネージャ)が起動されます。

起動中にエラーが発生すると、Essbase カーネルは自身をシャットダウンします。

データベース設定の優先度の理解

Essbase では、essbase.cfg ファイルで、一部のデータベース・ストレージ設定のデフォルト値が提供されます。デフォルト設定のままにすることも、次の 2 つの方法で値を変更することもできます:

- essbase.cfg ファイル内の値を変更することで、Essbase サーバー上のすべてのデータベースのストレージ設定を定義できます。
- 843 ページの「データベース設定の指定および変更」で指定されている任意の方法を使用すると、単一のデータベースのストレージ設定を定義できます。

個々のデータベースを変更すると、essbase.cfg の設定および関連するデータベースの Essbase のデフォルト値が完全に上書きされ、次の変更または破棄まで永続的に適用されます。

MaxL または管理サービス・コンソールを使用してデータベース・レベルで設定を変更すると、これらの変更は、表 152 に示すように、異なるタイミングで有効になります:

表 152　データベース設定の優先度

設定	設定が有効になる時期
• インデックス・キャッシュ • データ・ファイル・キャッシュ • データ・キャッシュ • キャッシュ・メモリーのロック • ディスク・ボリューム	データベースを停止して再起動した後
分離レベル・パラメータ、並行性パラメータ	値を設定後、アクティブなトランザクションが初めて存在しなくなった時点

設定	設定が有効になる時期
その他のすべての設定	即時

essbase.cfg 内のこれらのデータベース設定を手動で変更した場合は、変更を有効にするために、関連するアプリケーションを停止して再起動する必要があります。

注： インデックス・ページのサイズは、8KB に固定されています。これは、データベースの移行を簡単にすると同時に、入出力のオーバーヘッドを削減するためです。

Essbase による設定の読取り方法の理解

Essbase では、ユーザーが Essbase サーバーを開始し、データベース設定の指定および変更で説明されている方法を使用して作成した設定をデータベースに適用した時点で843 ページの「データベース設定の指定および変更」が読み取られます。

Administration Services、ESSCMD または MaxL を使用して指定したデータベース設定は、データベースの設定を適用した後に essbase.cfg 内の設定を変更した場合でも、常に essbase.cfg 設定を上書きします。Essbase で essbase.cfg が使用されるようにするには設定を削除するしかなく、削除しても Essbase サーバーを再起動しないと有効になりません。

最後に入力された設定の表示

➤ 最後に入力された設定を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データベース・プロパティの設定	Oracle Essbase Administration Services Online Help
MaxL	**display database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	GETDBSTATE GETDBINFO	『Oracle Essbase テクニカル・リファレンス』

データベース設定のカスタマイズ

Essbase サーバー上の各データベースの設定をカスタマイズできます。この項の情報は、各設定によって制御される内容や、各設定の指定方法を理解するのに役立ちます。また、いくつかの例も示されています。パフォーマンスに関連した設定の表については、第 54 章「Essbase のパフォーマンスの向上」を参照してください。

注： Essbase サーバー全体に適用される設定は、essbase.cfg で構成します。詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

表 153 に、カスタマイズできる主要なデータベース設定を示します:

表 153　主要なカーネル設定

設定	参照
インデックス・キャッシュ・サイズ	917 ページの「インデックス・キャッシュ・サイズの設定」
データ・ファイル・キャッシュ・サイズ	918 ページの「データ・ファイル・キャッシュ・サイズの設定」
データ・キャッシュ・サイズ	919 ページの「データ・キャッシュ・サイズの設定」
キャッシュ・メモリーのロック	916 ページの「キャッシュ・メモリーのロックを使用するかどうかの決定」
ディスク・ボリューム	847 ページの「ストレージの割当て」
データ圧縮	853 ページの「データ圧縮」
分離レベル	861 ページの「分離レベルの理解」

次の項では、カーネル設定の変更方法について説明し、例を示します。

データベース設定の指定および変更

データベース設定を変更する前に、Essbase の別の部分で変更された設定の優先度や、Essbase によるこれらの設定の読取り方法に関する情報を確認してください:

- 841 ページの「カーネル起動の理解」
- 842 ページの「Essbase による設定の読取り方法の理解」

➤ ほとんどのデータベース設定の指定に、次のツールを使用します:

ツール	トピック	場所
Administration Services	データベース・プロパティの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM SETDBSTATE	『Oracle Essbase テクニカル・リファレンス』

これらの方法により、同じデータベース設定を変更するための異なる方法が提供されます。まれに、essbase.cfg を使用して設定を指定することもあります。

注意 以前のバージョンの Essbase では、Essbase サーバー上の essbase.cfg ファイルで多くのデータベース設定を指定できました。バージョン 5.x 以降では、ほとんどの.cfg 設定が Essbase によって上書きされます。より新しいバージョンの Essbase による設定の処理方法の説明については、841 ページの「データベース設定の優先度の理解」および 842 ページの「Essbase による設定の読取り方法の理解」を参照してください。

MaxL の alter database の使用

変更するデータベース設定ごとに個別の **alter database** ステートメントを発行します。たとえば、次の MaxL スクリプトは Essbase にログオンし、3 つのデータベース設定を変更した後、ログオフします:

```
   login admin identified by secretword;
alter database sample.basic enable committed_mode;
alter database sample.basic set lock_timeout immediate;
alter database sample.basic disable create_blocks;
logout;
```

注： MaxL シェルを使用して MaxL ステートメントを発行する場合は、各 MaxL ステートメントをセミコロンで終端させます。ただし、MaxL ステートメントを Perl スクリプトに埋め込む場合は、セミコロン・ステートメント終端文字を使用しないでください。

MaxL を使用すると、データベース設定の変更を自動化するためのバッチ・スクリプトを記述できます。『Oracle Essbase テクニカル・リファレンス』にある「MaxL 言語リファレンス」を参照してください。

ESSCMD での SETDBSTATEITEM の使用

単純なアイテムの場合は、次のように、コマンド、パラメータを表すアイテム番号、アプリケーション、データベースおよびパラメータの値を指定します:

```
SETDBSTATEITEM 2 "SAMPLE" "BASIC" "Y";
```

分離レベル(アイテム 18)など複数の値を設定する必要のあるパラメータには、値を複数指定します。次の例で「BASIC」の後の値はすべてパラメータです:

```
SETDBSTATEITEM 18 "SAMPLE" "BASIC" "1" "Y" "-1";
```

パラメータ番号がわからない場合は、その番号を省略すると、すべてのパラメータとそれに対応する番号が Essbase によってリストされます。また、Essbase によって、データベースとアプリケーション名の入力も求められます。

各パラメータには、別個の SETDBSTATEITEM コマンドを使用します。複数のパラメータ番号を、同じ行で続けることはできません。

SETDBSTATE コマンドと SETDBSTATEITEM コマンドのパラメータについては、『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： SETDBSTATEITEM や SETDBSTATE は、指定されたデータベースにのみ影響を与えます。

SETDBSTATEITEM (または SETDBSTATE)をバッチ・スクリプトに含めることができます。バッチ処理の詳細は、1195 ページの「バッチ・プロセスにおけるスクリプト・ファイルおよびバッチ・ファイルの使用」を参照してください。特定の ESSCMD 構文については、『Oracle Essbase テクニカル・リファレンス』を参照してください。

50 ストレージの割当てとデータの圧縮

この章の内容

ストレージの割当て

Essbase は、データ・ファイルを使用してデータ・ブロックを保管します。デフォルトでは、データ・ファイルは、関連付けられたデータベース・フォルダに配置されます。データ・ファイルは、essn.pag という命名規則に従います。ここで、n は 1 以上、65,535 以下です。

Essbase は、インデックス・ファイルを使用して、データベースのインデックスを保管します。デフォルトでは、インデックス・ファイルは、関連付けられたデータベース・フォルダに配置されます。インデックス・ファイルは、essn.ind という命名規則に従います。ここで、n は 1 以上、65,535 以下です。

Essbase は、データおよびインデックス・ファイルのためのストレージを自動的に割り当てます。ディスク・ボリュームを使用して、これらのファイルへのストレージの割当て方法を制御できます。

➤ ストレージの割当て方法を制御するようにディスク・ボリュームを指定するには:

1 **Essbase がインデックス・ファイルとデータ・ファイルを保管するために使用するスペースのサイズを確認します。サイズを確認する方法については、848 ページの「インデックス・ファイルとデータ・ファイルのサイズ確認」を参照してください。**

2 **ストレージを制御する方法を選択します:**

- Essbase がこれらのファイルを保管するために使用するボリューム(ドライブ)を指定します。848 ページの「ディスク・ボリュームの指定」を参照してください。
- 1 つのボリュームに Essbase をインストールし、別のボリュームにファイルを保管します。

インデックス・ファイルとデータ・ファイルのサイズ確認

➤ インデックス・ファイル(.ind ファイル)やデータ・ファイル(.pag ファイル)の名前、カウント、サイズおよび合計を表示したり、各ファイルが Essbase で開いているかどうかを判断したりするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	インデックス・ファイルとデータ・ファイルのサイズ確認	Oracle Essbase Administration Services Online Help
ESSCMD	LISTFILES	『Oracle Essbase テクニカル・リファレンス』

注： NTFS ボリューム上に存在するインデックス・ファイルやデータ・ファイル用に Windows で提供されるファイル・サイズ情報は、正確でない場合があります。Administration Services および LISTFILES によって提供されるファイル・サイズ情報は正確です。

ディスク・ボリュームの指定

ディスク・ボリュームを使用して、Essbase のインデックス・ファイル(essn.ind)およびデータ・ファイル(essn.pag)を保管する場所を指定します。

ファイルは、次のディレクトリ構造でディスク・ボリュームに書き込まれます:

```
.../app/app_name/db_name
```

ディスク・ボリューム設定を使用しない場合は、Essbase によって、ARBORPATH ディレクトリが存在するボリューム上にデータのみが保管されます。ARBORPATH 変数が設定されていない場合は、Essbase では、サーバーが起動されたボリューム上にデータのみが保管されます。

新しいファイルの場合、ディスク・ボリューム設定への変更は、そのデータベースの次回の起動時に有効になります。既存のファイルやボリュームには影響を与えません。

注： インデックス・ファイルとデータ・ファイルのサイズを確認する方法については、848 ページの「インデックス・ファイルとデータ・ファイルのサイズ確認」を参照してください。

ディスク・ボリュームは、Administration Services、MaxL または ESSCMD を使用して指定できます。ディスク・ボリュームを使用する場合は、Essbase によって、ディスク・ボリュームごとに次のオプションが提供されます:

- ボリューム名

- ボリューム上で使用する最大スペース(Administration Services では「パーティション・サイズ」、ESSCMD では「ボリューム・サイズ」と呼ばれます)
- ファイル・タイプ

  インデックス・ファイル、データ・ファイルまたはその両方を指定できます。デフォルトは、同じボリューム上のインデックス・ファイルとデータ・ファイルです。
- 最大ファイル・サイズ

  デフォルト値および推奨値は、2,097,152KB(2GB)です。最大ファイル・サイズに達すると、Essbase によってファイルが作成され、そのファイルに漸増的に名前が付けられます。たとえば、`ess00001.ind` が最大サイズに達すると、Essbase によって `ess00002.ind` が作成されます。

注意 ボリューム名を指定して、ボリューム・サイズを指定しなかった場合は、Essbase によってそのボリューム上の使用可能なすべてのスペースが使用されます。

Essbase は、次の状況でデータ・ファイルとインデックス・ファイルを作成します:

- すべてのファイルの合計サイズが、ディスク・ボリューム設定で指定されている最大サイズに達した場合

  デフォルトでは、この合計は、すべてのインデックス・ファイルおよびデータ・ファイルのサイズの合計です。ファイル・タイプとしてインデックスを指定した場合、この合計は、ボリューム上のすべてのインデックス・ファイルの合計を示します。ファイル・タイプとしてデータを指定した場合、この合計は、ボリューム上のすべてのデータ・ファイルの合計を示します。

  たとえば、Essbase ファイルのためにボリューム E 上で最大 12GB、ボリューム F 上で最大 16GB、ボリューム G 上で最大 16GB を使用するとします。インデックス・ファイルとデータ・ファイルのサイズがボリューム E 上で 12GB に達し、ディスクにさらにデータを書き込む必要がある場合は、Essbase によってボリューム F 上にファイルが作成されます。
- 任意のボリューム上の個々のインデックス・ファイルまたはデータ・ファイルのサイズが 2GB に達した場合

  前述した例(ボリューム E では 12GB、ボリューム F では 16GB、ボリューム G では 16GB)で、ボリューム E および F が最大容量に達し、Essbase がボリューム G を使用しているとします。ボリューム G 上で、Essbase によって `ess00001.ind` が作成されてデフォルトの 2GB の制限までデータが書き込まれ、`ess00001.pag` が作成されて 1GB までデータが書き込まれています。ボリューム G では 16GB 内の 3GB が使用されています。`ess00001.ind` が 2GB の最大ファイル・サイズに達しているため、次回、Essbase がディスクにインデックス・ファイルを書き込むときにストレージ・スペースが必要になった場合は、`ess00002.ind` が作成されます。`ess00002.ind` が 2GB の制限に達すると、Essbase は `ess00003.ind` を作成します。Essbase は、データ・ファイルについても同じ手順を実行します。

Essbase はファイルに、ess00001.xxx から始まる連続的な名前を付けます。ここで、xxx はインデックス・ファイルの場合は ind、データ・ファイルの場合 pag であり、最大 ess65535.xxx まで続きます。この命名規則はボリュームごとに適用されるため、上の例では、E、F および G の各ボリュームに、ess00001.pag および ess00001.ind という名前のファイルがあります。

ディスク・ボリュームを指定する際は、次のガイドラインに留意してください:

- ディスク・ボリュームは、各ボリュームを使用する順序で指定してください。Essbase がインストールされているボリュームをボリュームの 1 つとして指定する必要はありません。1 つのボリュームにインストールし、他のボリュームにデータを保管できます。
- あるボリュームがボリュームの容量に達すると、Essbase は次のボリュームに移動します。
- 指定されたすべてのボリュームが最大容量に達すると、Essbase では進行中のデータベース操作を停止し、エラー・メッセージを発行して、致命的なエラーの処理を実行します。1155 ページの「致命的なエラーの処理の理解」を参照してください。これらのイベントが発生した場合は、データベースをシャットダウンし、より多くのディスク・スペースを割り当ててから、データベースを再起動してください。
- Essbase に対しボリュームにファイルを保管するのを停止するよう指示できます。Essbase は、必要に応じてそのボリュームに引き続きアクセスできますが、そのボリュームにこれ以上インデックスやデータの情報が保管されることはなくなります。ボリュームへの情報の保管を停止するには、削除するボリューム定義を選択し、「削除」をクリックします。
- ディスク・ボリュームは、データベースごとに設定します。同じボリューム上のスペースを複数のデータベースで使用できるため、慎重にスペースを割り当ててください。たとえば、データベース 1 のためにボリューム A 上の 7GB を、データベース 2 のためにボリューム A 上の 7GB を指定した場合は、Essbase ファイルのためにボリューム A 上の 14GB を割り当てたことになります。

Windows では、Universal/Uniform Naming Convention (UNC)を使用して、ディスク・ボリュームをネットワーク・リソースの場所(共有ディレクトリなど)として指定できます。構文:

```
\\ComputerName\SharedFolder\Resource
```

Administration Services を使用したディスク・ボリュームの指定

➤ Administration Services を使用してディスク・ボリュームを指定するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「ディスク・ボリュームの設定」を参照してください。

MaxL を使用したディスク・ボリュームの指定

➤ 新しいボリュームの割当て、ボリュームの削除、またはボリューム設定の変更を行うには、『Oracle Essbase テクニカル・リファレンス』の **alter database (ディスク・ボリューム)**に関する項を参照してください。

ESSCMD を使用したディスク・ボリュームの指定

➤ 新しいボリュームを割り当てるには、『Oracle Essbaseテクニカル・リファレンス』の「ESSCMD SETDBSTATEITEM 23」を参照してください。

追加する新しいディスク・ボリュームの番号を、コマンドラインで入力しなかった場合には、入力を求めるプロンプトが表示されます。

さらに、コマンドラインで次の値を入力しなかった場合、新しいボリュームごとに、それらの入力を求めるプロンプトが表示されます。

- ボリューム名(各ボリューム)
- ボリューム・サイズ(ボリューム上で使用する最大スペース)

  デフォルト値は無制限であり、最小設定は 8MB です。

ESSCMD を使用する場合は、ボリューム・サイズをバイト(B)、キロバイト(K)、メガバイト(M)、ギガバイト(G)またはテラバイト(T)単位で指定できます。ESSCMD には、最小値、最大値、現在の値、無制限を示す 0 が表示されます。

- ファイル・タイプ

  インデックス・ファイル、データ・ファイルまたはその両方を指定できます。デフォルトは、インデックス+データ(同じボリューム上のインデックス・ファイルとデータ・ファイル)を示す 3 です。
- ファイル・サイズ(Essbase によって新規ファイルが作成される前に、ファイル・タイプで指定された各ファイルの最大サイズ)

  デフォルト値は 2GB であり、最小設定は 8MB です。

次の例では、ボリューム E 上で最大 10GB を割り当て、2GB の最大ファイル・サイズを設定し、さらにデータ・ファイルを E にのみ保管するように指定しています:

```
SETDBSTATEITEM 23 "SAMPLE" "BASIC" "1" "E" "10G" "2" "2G"
```

➤ 割当てボリュームの設定を変更するには、ESSCMD で SETDBSTATEITEM 24 と入力し、プロンプトに従うか、コマンドラインで必要な値を入力します。

コマンドラインで指定しない場合、ESSCMD から次の値を求めるプロンプトが表示されます:

- ボリューム番号

  現在定義されているディスク・ボリュームのリストや、各ボリュームに割り当てられている番号を表示するには、ESSCMD の GETDBSTATE コマンドを使用します。
- ボリューム名
- ボリューム・サイズ
- ファイル・タイプ
- ファイル・サイズ

次の例では、ボリューム C 上で最大 20GB を割り当て、2GB の最大ファイル・サイズを設定しています:

```
SETDBSTATEITEM 24 "SAMPLE" "BASIC" "1" "C" "20G" "3" "2G"
```

➤ Essbase によってボリュームにこれ以上ファイルが保管されないようにするには、ESSCMD で SETDBSTATEITEM 25 と入力してプロンプトに従うか、またはコマンドラインで必要な値を入力します。Essbase は、割当て解除されたボリューム上のファイルに引き続きアクセスしますが、そのボリュームに新しいファイルを書き込むことはなくなります。

削除するボリューム定義の値をコマンドラインで指定しなかった場合は、ESSCMD により値の入力を求めるプロンプトが表示されます。現在定義されているディスク・ボリュームのリストや、各ボリュームに割り当てられている番号を表示するには、ESSCMD の GETDBSTATE コマンドを使用します。

次の例では、4 番目として指定されているボリュームの割当てを解除します:

```
SETDBSTATEITEM 25 "SAMPLE" "BASIC" "4"
```

注： アプリケーションまたはデータベースを削除した場合、Essbase では、アプリケーションやデータベースを含む、ディスク・ボリューム上にあるディレクトリは削除されません。コンピュータのオペレーティング・システムでは、ディスク上にフォルダおよびファイル・ラベルが表示されたままになります。ただし、ディスク・ボリューム上で削除したアプリケーションまたはデータベースと同じ名前を再使用することはできます。

この構文の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

UNIX では、volume_name はマウントされる UNIX ファイル・システムです。Essbase のインストール・ディレクトリ(ESSBASEPATH)への完全修飾パス名を入力する必要があります。Essbase によって、そのパスに app ディレクトリが自動的に追加されるので、app ディレクトリを指定する必要はありません。

次に例を示します:

```
/vol2/EssbaseServer 10M
```

ボリューム・サイズは、そのボリュームに割り当てられた最大スペース(KB)です。デフォルト値は無制限であり、Essbase によってそのボリューム上の使用可能なすべてのスペースが使用されます。

ストレージを制御するためのボリュームの指定例の確認

Essbase ファイルのために、ボリューム E 上で最大 20GB、ボリューム F 上で最大 25GB、ボリューム G 上で最大 25GB を使用するとします。2GB のデフォルトのファイル・サイズ制限を使用しています。データをロードすると、Essbase によってボリューム E 上に最大 20GB が保管されます。データベースのサイズが 20GB を超えると、Essbase によってボリューム F 上に次の 25GB が保管され、以降も同様に処理されます。

表 154 は各ディスク・ボリュームに関する情報の例を示しています。

表 154　ディスク・ボリューム情報

ディスク・ボリューム	パーティション・サイズ	ファイル・タイプ	ファイル・サイズ
E	20971520K	インデックス+データ	20971520K
F	26214400K	インデックス+データ	20971520K
G	26214400K	インデックス+データ	20971520K

データ圧縮

Essbase では、ディスク上に保管されているデータ・ブロックを圧縮するかどうか、さらにはどの圧縮方式を使用するかを選択できます。Essbase では、データ圧縮が使用可能になっていると、データ・ブロックがディスクに書き込まれるときにそれらのデータ・ブロックが圧縮されます。圧縮データ・ブロックがデータ・キャッシュにスワップインされるときに、Essbase によって、そのブロック(空のセルを含む)が完全に展開されます。

一般には、データ圧縮によってストレージの使用が最適化されます。圧縮率の統計を確認することによって、圧縮効率をチェックできます。859 ページの「圧縮率の確認」を参照してください。

Essbase には、データ圧縮のためのいくつかのオプションが用意されています:

- ビットマップ圧縮(デフォルト)

Essbase により、非欠落値のみが保管され、ビットマップ方式が使用されます。

- ランレングス符号化(RLE)

  Essbase によって、ゼロおよび#MISSING 値を含む、反復的な、連続する値が圧縮されます。

- ZLIB 圧縮

  Essbase により、圧縮されている実際のデータに基づいてデータ辞書が構築されます。

- インデックス値ペア圧縮

  Essbase では、ブロックの密度が 3%未満の場合にこの圧縮が適用されます。

- 圧縮しない

  Essbase で、データ・ブロックがディスクに書き込まれるときにそれらのデータ・ブロックが圧縮されません。

Essbase では、データ・ブロックがディスクに書き込まれるときに圧縮されるため、同じデータ・ファイル内にビットマップ、RLE および非圧縮データ・ブロックが共存できます。次のルールに注意してください:

- どの圧縮方式を使用した場合でも、Essbase では圧縮したデータ・ブロックは、データ・キャッシュに取り込まれるときにフル・サイズに展開されます。
- Essbase によりディスク上にブロックが保管されるとき、Essbase では、データ・キャッシュに取り込まれたときに圧縮されていたかどうかにかかわらず各ブロックが同様に処理されます。いずれの場合も、Essbase では、指定された圧縮タイプ(「圧縮しない」が指定されている場合の圧縮なしを含む)に従ってブロックを圧縮します。
- 圧縮が使用不可になっている場合は、Essbase によって、完全に展開されたブロックがディスクに書き込まれます。

ブロックの密度が非常に高く(90%以上)、連続した繰返しデータ値がほとんどない場合は、データ圧縮を使用不可にした方がよいこともあります。このような状況では、圧縮を使用可能にするとリソースが不必要に消費されます。

ビットマップ・データ圧縮

データ・ブロックが圧縮されていない場合、Essbase では、どの非欠落セル(データ値が存在するセル)の保管にも 8 バイトが使用されます。また、ブロックが圧縮されているかどうかにかかわらず、Essbase はブロックごとに 72 バイトのブロック・ヘッダーを使用します。

ビットマップ圧縮では、Essbase はビットマップを使用してデータ・セルを表し、ビットマップ、ブロック・ヘッダーおよびその他の制御情報のみを保管します。ビットマップでは、セル値が欠落しているか("0")、欠落していないか("1")にかかわらず、データ・ブロック内のセルごとに 1 ビット("1"または"0"と表記)を使用します。したがって、ビットマップ方式ではデータ圧縮のために固定のオーバーヘッドが発生します。

ビットマップ圧縮を使用している場合は、Essbase によって非欠落値のみが保管され、繰返し値や 0 は圧縮されません(855 ページの「RLE データ圧縮」で説明されている RLE 圧縮と対照的)。Essbase では、データ・キャッシュにデータ・ブロックが配置されるとき、欠落している値がビットマップを使用して再作成され、データ・ブロックが完全に展開されます。

次に示す例は圧縮されていないデータ・ブロックの一部で、8 つのセル(1 番目の行がセル 1 から 4、2 番目の行がセル 5 から 8)があります。1 番目の行のデータ値は 25、#MISSING、#MISSING および#MISSING です。2 番目の行のデータ値は #MISSING、16、7 および#MISSING です。

```
   Uncompressed data block

25     #MISSING  #MISSING  #MISSING
#MISSING  16     7      #MISSING
```

データ・ブロックがメモリー内で完全に展開される場合、Essbase は 64 バイト(8 バイト* 8 セル)を使用します。データが圧縮されずにディスク上に保管される場合、Essbase は 24 バイト(8 バイト*データのある 3 セル - つまりセル 1、6 および 7)を使用します。

データが圧縮されてディスク上に保管される場合、Essbase は 1 バイト(1 ビット* 8 セル、8 ビット= 1 バイト)を使用してビットマップを保管します。次に示すのは、前述した圧縮されていないデータ・ブロックのビットマップの例です:

```
   Bitmap of compressed data block

1      0      0      0
0      1      1      0
```

ほとんどの場合、ビットマップ圧縮ではディスク・スペースをより効率的に節約できます。ただし、これはデータの構成に大きく依存しています。

RLE データ圧縮

ランレングス符号化(RLE)圧縮方式を使用している場合は、Essbase によって、連続した繰返し値、つまり、連続して 3 回以上繰り返される任意の値(0 を含む)が圧縮されます。Essbase によって、各繰返し値と、連続して繰り返される回数が追跡されます。

RLE 圧縮では、Essbase は 3 つ以上の連続した繰返しセルのセットに対する合計 24 バイトについて、8 バイト、プラス 16 バイトの繰返し係数を使用します。繰り返されていない、または 2 回だけ繰り返されている値の場合、Essbase は各値に対して 8 バイトを使用します。

次に示すデータ・ブロックの例では、Essbase は、1 番目の行の 3 つの連続した #MISSING 値(セル 2、3 および 4)と、2 番目の行の一番左側の#MISSING 値(セル 5)を繰返し値として想定します。2 番目の行の一番右側の#MISSING 値(セル 8)は繰

返し値ではありません。この値は、(それぞれ 16 および 7 の)データ値を持つセル 6 および 7 によって、セル 5 の#MISSING 値から離されているためです。

```
   Data values in data block

25     #MISSING  #MISSING  #MISSING
#MISSING  16     7      #MISSING
```

データ・ブロックがメモリー内で完全に展開される場合、Essbase は 64 バイト(8 バイト* 8 セル)を使用します。データが圧縮されずにディスク上に保管される場合、Essbase は 24 バイト(8 バイト*データのある 3 セル - つまりセル 1、6 および 7)を使用します。

データが圧縮されてディスク上に保管される場合、Essbase は 56 バイトを使用します:

- セル 1 - 8 バイト
- セル 2 から 5 - 24 バイト(8 バイト+ 16 バイト)
- セル 6 - 8 バイト
- セル 7 - 8 バイト
- セル 8 - 8 バイト

また、ブロックが圧縮されているかどうかにかかわらず、Essbase はブロックごとに 72 バイトのブロック・ヘッダーを使用します。

ZLIB 圧縮

この方法は、PNG、Zip、gzip などのパッケージで使用されます。計算およびデータ・ロードは、バッファ I/O と ZLIB 圧縮を使用するより、直接 I/O と ZLIB 圧縮を使用した方が高速です。データ・ストレージが最大の制約要因である場合は ZLIB を使用しますが、状況によっては、データ・ロードがビットマップ圧縮より最大 10%低速になる可能性があることに注意してください。ただし、ZLIB 圧縮を使用すると、データベースのサイズは一般に大幅に小さくなります。

ビットマップ圧縮では、欠けている値を追跡するためにアルゴリズムが使用され、他のタイプのデータとのやり取りがありませんが、ZLIB 圧縮では、圧縮されている実際のデータ(欠落しているすべての値を含む)に基づいてデータ辞書が構築されます。そのため、極端に高密度のデータの場合は、ZLIB 圧縮によってビットマップ圧縮より高い圧縮率が得られます。ただし、ZLIB アルゴリズムの効果は圧縮されている実際のデータに(ビットのレベルで)依存するため、どのような場合に ZLIB 圧縮によってビットマップ圧縮より高い圧縮率が得られるかに関する一般的なガイドラインを密度のみに基づいて示すことはできません。他の圧縮方式とは異なり、節約できるストレージ・スペースは、欠けているセルの数や、等しい値を持つ連続したセルの数にはほとんど、またはまったく関係しません。一般に、ビットマップ圧縮または RLE 圧縮と比較して、ZLIB の圧縮率はデータの密度または異種性が高ければ高いほど向上します。ただし、状況によっては、ZLIB によってビットマップ圧縮または RLE 圧縮を使用した場合より優れた結果が得られない

可能性があります。代表的なデータ・サンプルを使用してテストすることをお薦めします。

ZLIB で得られる可能性のあるストレージ削減を見積るには、通常の圧縮手法(ビットマップまたは RLE)を使用して実際のデータの少量のサンプリングを含む小さなデータベースを作成し、Essbase サーバーをシャット・ダウンします。作成されたデータ・ファイルのサイズに注目してください。次に、サンプル・データベース内のデータを消去し、圧縮設定を ZLIB に変更して、同じサンプル・データを再ロードしてから、Essbase サーバーを再度シャット・ダウンします。ここでは、使用されているストレージの違いに注目します。また、小さなサンプル・データベースを使用して、計算またはデータ・ロード速度の変化を見積ることもできます。

インデックス値ペア圧縮

インデックス値ペアは、ブロックがきわめて疎の状態にある、ブロック・サイズの大きいデータベースの圧縮に対応しています。この圧縮アルゴリズムをユーザーが選択することはできませんが、該当する場合は常に、データベースによって自動的に使用されます。ただし、ユーザーは、Administration Services を使用して、圧縮タイプ(なし、ビットマップ、RLE、ZLIB)を選択する必要があります。

表 155 は、ユーザーが選択できる使用可能な圧縮タイプと、Essbase によって評価および適用される圧縮タイプを示しています。

表 155　インデックス値ペア圧縮

選択された圧縮タイプ	評価された圧縮タイプ
なし	なし
ビットマップ	ビットマップ、インデックス値ペア
RLE	RLE、ビットマップ、インデックス値ペア
ZLIB	ZLIB

たとえば、ユーザーが RLE を選択すると、Essbase で各ブロックを確認して、RLE、ビットマップまたはインデックス値ペアのどの圧縮タイプで最も高い圧縮率が得られるかが評価されます。ユーザーが ZLIB を選択した場合、適用される圧縮タイプは ZLIB のみになります。

使用する圧縮タイプの決定

圧縮の設定は、ビットマップ(デフォルト)、RLE、ZLIB および圧縮なしの 4 つから選択できます。

ほとんどの場合、設定の選択について心配する必要はありません。通常は、ビットマップ圧縮により、高速なパフォーマンスと小さいデータ・ファイルという最適な組合せが得られます。ただし、これはデータの構成に大きく依存しています。

データ圧縮は、CPU を集中的に使用します。圧縮設定を選択する場合は、計算のコストと、I/O のコストおよびディスク・スペースのコストの間のトレードオフを考慮してください。

一般に、特定のブロックに関して繰返しの非欠落データ・セルが多数あり、その値が同じである場合は、ビットマップの設定より RLE の設定を使用した方がデータベースの圧縮率は高くなります。RLE 圧縮を使用すると、ビットマップ圧縮を使用した場合より計算のコストは高くなります。ただし、RLE 圧縮を使用してデータベースが大幅に圧縮される場合は、I/O コストの低下によってパフォーマンスが向上する可能性があります。

ZLIB 圧縮を使用した場合、通常はデータベースが圧縮されますが、常にそうなるとは限りません。ZLIB 圧縮を使用すると、CPU 処理が大幅に増加します。ほとんどのデータベースでは、この追加の処理によって、ブロック・サイズ削減の利点が打ち消されます。ただし、ZLIB 圧縮を使用してデータベースが大幅に圧縮される場合は、I/O コストの低下によってパフォーマンスが向上する可能性があります。

「なし」の圧縮設定では、ビットマップ圧縮と比較して、データベースのディスク使用量は削減されません。実際、ビットマップ圧縮はきわめて高速であるため、圧縮なしではデータベースのパフォーマンス向上に関して違いが出ない場合があります。

データベースはそれぞれ固有であり、ここで説明した内容は圧縮タイプの一般的な特性であることに注意してください。デフォルトのビットマップ圧縮はほとんどのデータベースに適していますが、個々のデータベースにとっての最適な圧縮設定を決定するための最善の方法は、各設定を試してみることです。

データ圧縮の設定の変更

データ圧縮設定への変更は、Essbase によりデータ・ブロックがディスクに書き込まれるときにただちに有効になります。すでにディスク上にあるブロックに対して、Essbase により圧縮方式が変更されたり、圧縮が使用可能または使用不可になったりすることはありません。すでにディスク上にあるブロックのデータ圧縮設定を変更した場合、Essbase では、Essbase の次回のブロックへのアクセス、更新および保管時に新しい圧縮方式が使用されます。

➤ 現在の設定を表示または変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データ圧縮方式の選択	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	データ圧縮を使用可能または使用不可にする場合: SETDBSTATE または: SETDBSTATEITEM 14 データ圧縮のタイプを設定する場合: SETDBSTATEITEM 15	『Oracle Essbase テクニカル・リファレンス』

SETDBSTATEITEM の使用例

➤ データ圧縮を使用可能または使用不可にするには、ESSCMD で SETDBSTATEITEM 14 と入力し、プロンプトに従うか、またはコマンド・ラインで必要な値を入力します。

コマンドラインで指定しない場合、ESSCMD から次の値を求めるプロンプトが表示されます:

- ディスク上のデータ圧縮-Y (はい、デフォルト)または N (いいえ)を入力します。
- データ圧縮のタイプ。1 (ランレングス符号化)または 2 (ビットマップ、デフォルト)を入力します。

➤ データ圧縮のタイプを指定するには、ESSCMD で SETDBSTATEITEM 15 と入力してプロンプトに従うか、またはコマンド・ラインで必要な値を指定します。ESSCMD により、「1」(ランレングス符号化)または「2」(ビットマップ、デフォルト)の値を入力するよう求められます。

次の例では、ビットマップ圧縮が使用可能になります:

```
SETDBSTATEITEM 14 "SAMPLE" "BASIC" "Y" "2"
```

この構文の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

圧縮率の確認

圧縮率は、有効になっている圧縮タイプには関係なく、圧縮されたブロック・サイズ(オーバーヘッドを含む)と圧縮されていないブロック・サイズの比率を表します。オーバーヘッドとは、圧縮/展開を管理するメカニズムに必要な領域のことです。

➤ 圧縮率を確認するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	圧縮率の確認	Oracle Essbase Administration Services Online Help
ESSCMD	GETDBSTATS	『Oracle Essbase テクニカル・リファレンス』

注： この数値が大きいほど、圧縮率は高くなります。圧縮率は、ブロックによって大きく変動することがあります。

データ・ブロック・サイズ

データ・ブロック・サイズは、密次元の特定の組合せにおけるデータ量によって決まります。たとえば、データベース内の 1 つ以上の次元の密または疎構成を変更すると、データ・ブロック・サイズが変わります。データ・ブロック・サイズは、8n バイトです。ここで、n は、密次元のその組合せに対して存在するセルの数です。

注： 最適なサイズの範囲は、8KB から 100KB までです。

データ・ブロックのサイズの判定については、1164 ページの「展開済データ・ブロックのサイズ」を参照してください。

➤ データベースのブロック・サイズを表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データ・ブロック統計の確認	Oracle Essbase Administration Services Online Help
ESSCMD	GETDBSTATS	『Oracle Essbase テクニカル・リファレンス』

51 データの整合性の確保

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

トランザクションの理解

データベースが読取り/書込みモードにある場合、Essbase では、サーバーへのすべての更新要求(データ・ロード、計算、計算スクリプト内のステートメントなど)がトランザクションと見なされます。Essbase は、トランザクション制御ファイル(dbname.tct)内のトランザクションに関する情報を追跡します。

トランザクション制御ファイルには、各トランザクションのエントリが含まれており、各トランザクションの現在の状態(アクティブ、コミット済または異常終了)が追跡されています。

869 ページの「Essbase によるトランザクションの処理方法の理解」を参照してください。

分離レベルの理解

分離レベルによって、Essbase がディスクへのデータをコミットする方法が決定されます。データがコミットされると、そのデータはサーバー・メモリーから取得され、ディスク上のデータベースに書き込まれます。Essbase は、ディスクへのデータを自動的にコミットします。データ・ブロックをコミットするためにユーザーが実行する明示的なコマンドは存在しません。ただし、データベースの分離

レベルを設定すると、Essbase がデータ・ブロックを自動的にコミットする方法が定義されます。

Essbase は、コミット・アクセスとアンコミット・アクセス(デフォルト)という、トランザクションのための 2 つの分離レベルを提供しています。コミット・アクセスを使用して、データの整合性を最適化できます。

アクセス・タイプの説明については、863 ページの「コミット・アクセス」および866 ページの「アンコミット・アクセス」を参照してください。

注： 分離レベルにコミット・アクセスを設定すると、データベースの再構築に必要なメモリーおよび時間が増加する可能性があります。

注： グリッド API は常にコミット・アクセス・モードです。

データ・ロック

Essbase は、作成、更新または削除されるブロックに対して書込み(排他)ロックを発行し、アクセスはされるが、変更は許可されないブロックに対して読取り(共有)ロックを発行します。適切なロックを発行することによって、Essbase では、ある操作で変更されたデータが同時更新によって壊されることがないようにしています。

この項では、データベース・アーティファクトに対するロックではなく、データ・ブロックに対するロックについて説明します。アウトラインやその他のアーティファクトのロックおよびロック解除については、788 ページの「アーティファクトのロックおよびロック解除」を参照してください。

表 156 で、ロック・タイプについて説明します:

表 156　基本的なロック・タイプ

ロック	説明
書込み(排他)ロック	ロックされたデータ・ブロックが、他の任意のトランザクションによってアクセスされないようにします。スプレッドシートのロックして送信操作を含む、すべてのデータ・ブロック更新に使用されます。
読取り(共有)ロック	他のトランザクションがデータ・ブロックに対して読取り専用アクセスを行えるようにしますが、データ・ブロックの変更はできないようにします。

表 157 は、Essbase が様々なタイプの操作のために発行するロックを示しています。

表 157　高レベルの機能によるロック

操作のタイプ	発行されるロック
スプレッドシートの取得	各データ・ブロックに対する読取り(共有)ロック。
取得およびロック	影響を受けるすべてのブロックに対する書込み(排他)ロック。以降の送信コマンドによってデータがコミットされます。

操作のタイプ	発行されるロック
派生ブロックの計算	計算対象のブロックに対する書込みロック。 ブロックの計算時には、そのブロックの子が含まれているすべてのブロックに、読取りロックが適用されます。
データ・ロード	書込みロック
再構築	書込みロック

Essbase がロックを処理する方法は、コミット・アクセスまたはアンコミット・アクセスのどちらが使用可能になっているかによって異なります。

コミット・アクセス

コミット・アクセスでは、一度に 1 つのトランザクションのみがデータ・ブロックを更新できるため、高レベルのデータの一貫性が提供されます。コミット・アクセスの場合、Essbase は、トランザクションが完了してコミットされるまで、そのトランザクションに関連したすべてのデータ・ブロックに対する読取り/書込みロックを保持できるようにします。ただし、最後にコミットされたデータ値への読取り専用アクセスは引き続き許可できます。

Essbase には、データ・ブロックに対するロックの発行時期を決定するためのオプションが用意されています:

- プリイメージ・アクセス(デフォルトで使用可能)。プリイメージ・アクセスでは、同時トランザクションの処理中にロックされるデータ・ブロックへの読取り専用アクセスがユーザーに提供されます。ユーザーには、ロックされたデータ・ブロックの、最後にコミットされたデータ値が表示されます。
- 待機またはタイムアウト:
  - 無限待機(デフォルト)。トランザクションは、必要なロックされたブロックに対するロックを取得するまで待機します。
  - 即時アクセスまたは待機なし。必要なブロックが別のトランザクションによってロックされている場合は、Essbase にロック・タイムアウト・メッセージが表示され、そのトランザクションが中止されます。
  - ユーザー指定の秒数。トランザクションは、必要なロックされたブロックに対するロックを取得するためにその秒数だけ待機します。トランザクションがロックを取得する前に指定された時間が経過した場合は、Essbase にロック・タイムアウト・メッセージが表示され、そのトランザクションが中止されます。

プリイメージ・アクセスが使用可能であれば、ユーザーは、データ・ブロックへの読取り専用アクセスには制限されません。ロックされたブロックへの書込みアクセスが必要な場合は、待機またはタイムアウトの設定に応じて、トランザクションが書込みアクセスを待機するか、またはタイムアウトします。トランザクションは、別のトランザクションによってロックされていないデータ・ブロックへの書込みアクセスを即座に取得します。

プリイメージ・アクセスが使用不可であり、ロックされたブロックへの読取りまたは書込みアクセスが必要な場合は、待機またはタイムアウトの設定に応じて、トランザクションが書込みアクセスを待機するか、またはタイムアウトします。

コミット・アクセスでのメモリーの考慮事項

コミット・アクセスでは、メモリーに関して次の点を考慮する必要があります:

- Essbase では、トランザクションがコミットされるまで、冗長データが保持されます。冗長データを収容するために、ディスク・スペースがデータベースのサイズの 2 倍になるようにしてください。
- 多数のブロックを含むモデルでは、コミット・アクセスでのメモリーに関する問題が発生する場合があります。1 つの計算当たり各ロック(ブロックごとに 1 つのロック)に約 80 バイトのメモリーが使用され、各ロックはトランザクションが完了するまでメモリー内に保持されます。プロセス当たりのアドレス可能なメモリー・スペースには制限があるため、多数のブロックを含むモデルは最終的にこの制限に達する可能性があり、それによってトランザクションが終了します。このような場合は、アンコミット・アクセスを使用することを検討してください。

注: 分離レベルにコミット・アクセスを設定すると、データベースの再構築に必要なメモリーおよび時間が増加する可能性があります。

コミット・アクセスでのロック

コミット・アクセスでは、読取りおよび書込みアクセスのために Essbase によってブロックがロックされます:

- 読取りアクセスの場合は、トランザクションが完了したかどうかにかかわらず、ロックは別のトランザクションから要求されるまで残ります。他のトランザクションはロックされたブロックを読み取ることができますが、変更はできません。
- 書込みアクセスを行った場合は、トランザクションによって変更対象の各ブロックがロックされ、そのトランザクションが完了するまで、ロックが保持されます。

表 158 は、複数のトランザクションが同じデータに対するロックの取得を競合しているときのコミット・アクセスでのロックの動作を示しています。この例では、トランザクション Tx1 が実行中であり、トランザクション Tx2 が同じデータへのアクセスを要求しています。(ロックされたブロックへのアクセスが、使用可能になっているオプションによって異なることに注意してください。オプションの説明については、863 ページの「コミット・アクセス」を参照してください。)

表 158　コミット・アクセスでのロックの動作

		Tx1 が読取りロックを保持; Tx2 が読取りロックを要求	Tx1 が読取りロックを保持; Tx2 が書込みロックを要求	Tx1 が書込みロックを保持; Tx2 が読取りロックを要求	Tx1 が書込みロックを保持; Tx2 が書込みロックを要求
プリイメージ・アクセス使用可能	待機(タイムアウト)期間を指定(無限待機または一定秒数待機)	Tx2 は、読取りロックを取得します。	Tx2 は、Tx1 が読取りロックを解放するまで待機します。	Tx2 は、プリイメージ・アクセスを取得します。	Tx2 は、Tx1 が書込みロックを解放するまで待機します。
	待機なし(タイムアウト)期間を指定(即時タイムアウト)	Tx2 は、読取りロックを取得します。	Essbase はタイムアウト・メッセージを発行し、トランザクションを中止します。	Tx2 は、プリイメージ・アクセスを取得します。	Essbase はタイムアウト・メッセージを発行し、Tx2 トランザクションを中止します。
プリイメージ・アクセスなし	待機(タイムアウト)期間を指定(無限待機または一定秒数待機)	Tx2 は、読取りロックを取得します。	Tx2 は、Tx1 が読取りロックを解放するまで待機します。	Tx2 は、Tx1 が書込みロックを解放するまで待機します。	Tx2 は、Tx1 が書込みロックを解放するまで待機します。
	待機なし(タイムアウト)期間を指定(即時タイムアウト)	Tx2 は、読取りロックを取得します。	Essbase はタイムアウト・メッセージを発行し、Tx2 トランザクションを中止します。	Essbase はタイムアウト・メッセージを発行し、Tx2 トランザクションを中止します。	Essbase はタイムアウト・メッセージを発行し、Tx2 トランザクションを中止します。

並行性パラメータの設定情報については、870 ページの「データの整合性設定の指定」を参照してください。

コミット・アクセスでの並行性

コミット・アクセスでは、2 つのトランザクションが同じブロックをロックしたり、同じブロックにアクセスしようと待機したりしているとデッドロックが発生して、どちらのトランザクションも完了できなくなる状態になる場合があります。

たとえば、トランザクション Tx1 で最初にデータ・ブロック B1 を、次にデータ・ブロック B2 を更新する必要がある場合は、まず B1 をロックした後、B2 をロックしようとします。一方、トランザクション Tx2 で最初にデータ・ブロック B2 を、次にブロック B1 を更新する必要がある場合、Tx2 はまず B2 をロックした後、B1 をロックしようとします。Tx1 は B1 をロックした後に B2 を待機しており、Tx2 は B2 をロックした後に B1 を待機しています。

Essbase のトランザクションは、ロックを取得しようと待機している間、定期的にデッドロック検出を実行します。デッドロックが検出された場合は、Essbase によってエラー・メッセージが発行され、そのトランザクションは失敗します。

別のユーザーによってロックされているブロックを更新しようとすると、Essbase は次のように動作します:

- 無限待機が設定されている場合、トランザクションは、必要なロックを取得するまで待機します。
- 待機が 0 (即時)に設定されており、必要なブロックがただちに使用可能にならない場合は、Essbase によってエラー・メッセージが表示され、そのトランザクションは失敗します。
- 待機がユーザー指定の秒数に設定されており、その時間が経過した場合は、Essbase によってエラー・メッセージが表示され、そのトランザクションは中止されます。
- 要求がタイム・アウトした場合は、操作を再び試みてください。

並行性オプションの設定方法については、870 ページの「データの整合性設定の指定」を参照してください。

コミット・アクセスでのロールバック

コミット・アクセスでは、サーバーがクラッシュした場合、サーバーが停止した時点で実行されていたトランザクションによるすべてのデータベース更新が Essbase によってロール・バックされ、中止されたトランザクションによって実行された変更は確実に元に戻されます。

トランザクションが致命的でないエラーのために中止された場合は、そのトランザクションによって実行されたすべての変更がロール・バックされます。

873 ページの「クラッシュしたデータベースのリカバリ」を参照してください。

アンコミット・アクセス

アンコミット・アクセス(デフォルトで使用可能)では、Essbase カーネルは、トランザクションがブロックごとに読取り/書込みロックを保持できるようにします。Essbase は、更新されたブロックを解放しますが、トランザクションが完了するか、または指定された制限(「同期ポイント」)に達するまでそのブロックをコミットしません。この制限は、下に示す方法で設定できます。

アンコミット・アクセスでは、最後にコミットされた時点のデータへの読取り専用アクセスが Essbase によって許可されるため、同じデータ・ブロックにユーザーが同時にアクセスすると予期しない結果が発生する場合があります。

アンコミット・アクセスでは、次に示す同期ポイント・パラメータを指定することによって、Essbase によって明示的なコミット操作が実行されるタイミングを制御できます:

- 「ブロックのコミット」(同期ポイントが発生するまでに変更されるブロックの数)。デフォルトは 3,000 です。

  「ブロックのコミット」を 0 に設定した場合、トランザクションの終了時に同期ポイントが発生します。
- 「行のコミット」(同期ポイントが発生するまでのデータ・ロードの行数)。デフォルトは 0 です。これは、データ・ロードの最後に同期ポイントが発生することを示します。

- 「ブロックのコミット」または「行のコミット」のどちらかが 0 以外の値になっている場合は、最初のしきい値に達した時点で同期ポイントが発生します。たとえば、「ブロックのコミット」は 10 だが、「行のコミット」が 0 のときにデータをロードした場合は、10 ブロックが更新された後に同期ポイントが発生します。「ブロックのコミット」が 5 で、「行のコミット」が 5 のときにデータをロードした場合は、5 行のロードと 5 ブロックの更新のどちらか早い方の後に同期ポイントが発生します。

操作中にユーザー定義のしきい値を超えると、Essbase により、その時点まで処理されたデータをコミットするための同期ポイントが発行されます。Essbase では、操作を完了するために必要な数だけの同期ポイントが実行されます。

注： Essbase では、並列計算の使用の実行可能性を分析中に「ブロックのコミット」と「行のコミット」の値を分析します。これらのパラメータが低すぎる値に設定されていることが Essbase によって検出されると、値は自動的に増加します。

同期ポイント・パラメータを指定する方法については、870 ページの「データの整合性設定の指定」を参照してください。

注意 Essbase では、トランザクション・セマンティクスを適用するために冗長データが保持されます。特に「ブロックのコミット」と「行のコミット」の両方を 0 に設定する場合は、冗長データを収容するために、ディスク・スペースがデータベースのサイズの 2 倍になるようにしてください。

アンコミット・アクセスでのメモリーの考慮事項

データ・キャッシュが小さすぎるために「ブロックのコミット」や「行のコミット」の設定で指定されたブロック数を保持できない場合は、トランザクションがコミットされる前であっても、キャッシュがいっぱいになるとすぐにブロックがディスクに書き込まれます。

アンコミット・アクセスでのロック

アンコミット・アクセスでは、Essbase でブロックの更新が終了するまで、書込みアクセスのために Essbase によってブロックがロックされます。コミット・アクセスでは、トランザクションが完了するまで、Essbase によってロックが保持されます。

表 159 は、多数のトランザクションが同じデータに対するロックの取得を競合しているときのアンコミット・アクセスでのロックの動作を示しています。この例では、トランザクション Tx1 が実行中であり、トランザクション Tx2 が同じデータへのアクセスを要求しています。

表 159　アンコミット・アクセスでのロックの動作

Tx2 がロック要求を発行したときのステータス	Tx1 が、読取りロックを保持している場合	Tx1 が、書込みロックを保持している場合
読取りロック	Tx2 は、読取りロックを取得します	Tx2 は、読取りロックを取得します
書込みロック	Tx2 は、書込みロックを取得します	Tx2 は、Tx1 がロックを解放するまで待機します

アンコミット・アクセスでの並行性

トランザクションが終了するまですべてのブロックがロックされる場合、アンコミット・アクセスでは、コミット・アクセスに比べ、より頻繁にブロックが解放されます。

アンコミット・アクセスでのロールバック

アンコミット・アクセスでは、サーバーがクラッシュした場合、最後の正常なコミットの時点からのすべてのデータベース更新が Essbase によってロール・バックされます。中止されたトランザクションの一部である更新がコミットされている可能性があります。トランザクションが、ユーザーが期待している方法で更新をコミットしたかどうかは、重複したトランザクションによってデータが更新され、コミットされた順序によって異なります。

トランザクションが致命的でないエラーのために中止された場合は、そのトランザクションが中止される前にトランザクションによって処理が終了したデータのみが Essbase によってコミットされます。

873 ページの「クラッシュしたデータベースのリカバリ」を参照してください。

並列計算とアンコミット・アクセス

Essbase で並列計算を使用している場合は、Essbase によってコミットしきい値が確認されます。

コミット・アクセスとアンコミット・アクセスの比較

分離レベルを選択するときには、次の問題を考慮してください:

- データベースのパフォーマンス。

  アンコミット・アクセスでは常に、コミット・アクセスより優れたデータベースのパフォーマンスが得られます。アンコミット・アクセスを使用している場合は、トランザクションの処理中に保持されるロックが Essbase によって作成されず、短期間の書込みロックに基づいてデータがコミットされます。

- データの一貫性。

  コミット・アクセスでは、アンコミット・アクセスより高いレベルのデータの一貫性が提供されます。データベースからの取得の一貫性も優れています。また、分離レベルがコミット・アクセスに設定されている場合は、一度に 1

つのトランザクションのみがデータ・ブロックを更新できます。複数のトランザクションがデータベースを同時に更新しようとするデータベースでは、この要素が重要です。

- データの並行性。

  アンコミット・アクセスでは、コミット・アクセスより優れたデータの並行性が提供されます。ブロックは、コミット・アクセス中よりも頻繁に解放されます。コミット・アクセスでは、デッドロックが発生する可能性があります。

- データベースのロールバック。

  アクティブなトランザクション中にサーバーのクラッシュやその他のサーバーの中断が発生した場合は、サーバーが再起動されたときに、Essbase カーネルによってトランザクションがロール・バックされます。コミット・アクセスでは、ロールバックによって、データベースはトランザクションの開始前の状態に戻ります。アンコミット・アクセスでは、ロールバックによって、コミットされるデータとコミットされないデータが発生する可能性があります。

  875 ページの「サーバーの中断が発生した場合に予測される結果」を参照してください。

Essbase によるトランザクションの処理方法の理解

Essbase ではトランザクションを開始から終了まで追跡し、必要に応じてメモリーへのデータ・ブロックのスワップ・インやスワップ・アウトを行ったり、トランザクションの完了時にデータ・ブロックをコミットしたりします。次のリストは、Essbase でトランザクションを処理する方法を示しています。これらのリスト・アイテムはすべて、コミットおよびアンコミット・アクセスに適用されます(861 ページの「分離レベルの理解」を参照)。

1. ユーザーまたはバッチ・プログラムによって操作が開始されます。
2. OLAP エンジンが Essbase カーネルに対しトランザクションを開始するように通知します。
3. Essbase カーネルがトランザクションを開始します。
4. OLAP エンジンが Essbase カーネルにデータを要求します。
5. Essbase カーネルにより、要求されたデータが特定されます。OLAP エンジンに、データと関連付けられた制御情報が渡されます。スプレッドシートを使用している場合は、シート上にこのデータが表示されます。
6. スプレッドシートを使用している場合は、データを変更するときに、ユーザーがデータを送信します。
7. Essbase カーネルは、トランザクションとトランザクション制御テーブル内のエントリを関連付けます。
8. OLAP エンジン側で操作が完了すると、OLAP エンジンは更新について Essbase カーネルに通知し、Essbase カーネルがそれに応じて内部のデータ構造を更新します。

9. 操作を完了するために必要な回数だけ手順 4 から手順 8 を繰り返します。
10. トランザクションが終了します。トランザクション処理中に Essbase でエラーが検出された場合は、そのトランザクションが中止されます。エラーが発生しない場合は、Essbase によってそのトランザクションがコミットされます。コミット・アクセスとアンコミット・アクセスでのコミット動作の違いについては、861 ページの「分離レベルの理解」を参照してください。
11. Essbase は、クライアントにトランザクションが完了したことを通知するメッセージ(たとえば「計算の合計経過時間...」)を発行します。

アンコミット・アクセスでは、アクティブな複数のトランザクションが同じデータにアクセスしている場合に、コミットされていないデータにアクセスしてしまうことがあります。アンコミット・アクセスでは、トランザクションの結果は予測できません。

アンコミット・アクセスでは、コミットしきい値が定義されていると、Essbase で 1 つのデータベース操作を複数の同期ポイントに分割する必要がある場合があります。コミットしきい値については、866 ページの「アンコミット・アクセス」を参照してください。

データの整合性設定の指定

Administration Services、MaxL または ESSCMD を使用して、分離レベル、同期ポイント・パラメータおよび並行性パラメータを指定できます。分離レベル設定への変更は、次回、アクティブなトランザクションが存在しなくなったときに有効になります。どちらの設定を選択するかの判断については、863 ページの「コミット・アクセス」および 866 ページの「アンコミット・アクセス」を参照してください。

➤ データの整合性設定を指定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**データの整合性オプションの設定**	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM 18	『Oracle Essbase テクニカル・リファレンス』

ESSCMD による分離レベル設定の指定例

➤ ESSCMD で分離レベル設定を指定するには、ESSCMD で SETDBSTATEITEM 18 と入力し、プロンプトに従うか、コマンドラインに必要な値を入力します。

1 (コミット・アクセス)または 2 (アンコミット・アクセス、デフォルト)を選択します。指定した値に応じて、ESSCMD から他のパラメータを入力するよう求められます(または、コマンドラインで値を指定できます)。

1 (コミット・アクセス)を選択した場合、次の情報の入力を求めるプロンプトが表示されます:

- プリイメージ・アクセス- Y (はい)または N (いいえ、デフォルト)。プリイメージ・アクセスでは、トランザクションの処理中にロックされるデータ・ブロックへの読取り専用アクセスがユーザーに提供されます。ユーザーには、ロックされたデータ・ブロックの、最後にコミットされたデータ値が表示されます。
- 待機(「データベース設定」ダイアログ・ボックス)またはタイムアウト(ESSCMD): -1、0 または n。
  - -1 は無限待機です。
  - 0 は即時アクセス(つまり、待機なし)です。
  - n は、ユーザーが指定する秒数です。

2 (アンコミット・アクセス)を選択した場合は、ESSCMD から次の値を入力するよう求められます。これらのオプションの説明については、866 ページの「アンコミット・アクセス」を参照してください。

- 内部コミットの発生までに変更が加えられるブロックの数
- 内部コミットの発生までに行われるデータ・ロードの行数

また、SETDBSTATEITEM で 19 から 22 のパラメータを指定することによって、分離レベル・パラメータ(プリイメージ・アクセスなど)を指定できます。パラメータを指定しないで SETDBSTATEITEM を入力すると、ESSCMD によって、各パラメータの番号とそれぞれの説明が記載されたリストが表示されます。

分離レベルを設定するための SETDBSTATEITEM の使用例を次に示します。この例では、コミット・アクセスおよびプリイメージ・アクセスを使用可能にして、無限の待機時間を指定します。

```
SETDBSTATEITEM 18 "SAMPLE" "BASIC" "1" "Y" "-1"
```

『Oracle Essbase テクニカル・リファレンス』を参照してください。

MaxL による分離レベル設定の指定例

分離レベル設定を指定するには、次の MaxL ステートメントを使用します:

```
alter database
appname
.
dbname
 enable committed_mode
```

『Oracle Essbase テクニカル・リファレンス』を参照してください。

冗長なデータの収容

データの整合性を保証するために、Essbase カーネルでは、冗長な(重複した)情報が一時的に保持されます。冗長な情報を収容するために、ディスク・スペースがデータベースのサイズの 2 倍になるようにしてください。

Essbase では、重要な制御情報を保管するために、dbname.esm という名前のファイルが保持されます。

注意 dbname.tct ファイル、dbname.esm ファイル、インデックス・ファイルおよびデータ・ファイルには、データ・リカバリのために重要な情報が含まれています。これらのファイルを決して変更したり削除したりしないでください。

構造およびデータの整合性の検査

データベースの整合性を検証し、データベースの破損がないかどうかを確認するには、次のアクションを実行します:

- 密な再構築を実行します。密な再構築では、データベース内のすべてのブロックが再作成されるので、これにより各ブロックのインデックス・ノードやセルが確認されます。
- データベースからすべてのレベルのデータをエクスポートします。データベース全体のエクスポートでは、データベース全体にわたるブロックおよびすべてのデータ値がアクセスされます。
- ESSCMD VALIDATE コマンドを使用して、構造およびデータの整合性を確認します。872 ページの「VALIDATE を使用した整合性の検査」を参照してください。

これらのいずれかの確認中にエラーが発生した場合は、データベースをバックアップから復元します。『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

VALIDATE を使用した整合性の検査

VALIDATE コマンドによって、多くの構造およびデータの整合性が検査されます:

- インデックス内の空きスペース情報に関する、構造の整合性の確認。
- インデックス・ページのデータ・ブロック・キーと対応するデータ・ブロックのデータ・ブロック・キーが比較されます。
- Essbase のインデックスには、すべてのデータ・ブロックのエントリが含まれています。VALIDATE では、すべての読取り操作について、インデックス・ページ内のインデックス・キーと対応するデータ・ブロック内のインデックス・キーが自動的に比較され、ブロック内の他のヘッダー情報が確認されます。不一致が検出されると、VALIDATE によりエラー・メッセージが表示され、データベース全体を確認するまで処理が続行されます。
- 増分再構築によって再構築が延期されたデータ・ブロックの再構築。

- データベース内のすべてのブロックで、各値が有効な浮動小数点値であるかどうかの検査。
- LRO カタログの構造の整合性が確認されます。

注： VALIDATE コマンドを発行するときは、データベースを読取り専用モードにしておくことをお薦めします。

Essbase で不一致が検出されると、VALIDATE エラー・ログにエラー・メッセージが記録されます。エラー・ログのファイル名を指定できます。指定しなかった場合は、Essbase からこの情報を入力するよう求められます。VALIDATE ユーティリティは、データベース全体を確認し終わるまで実行されます。

ESSCMD の VALIDATE コマンドで、これらの構造の整合性の検査を実行できます。

インデックスの空きスペースの検証中に、VALIDATE コマンドによってインデックス内の空きスペース情報の構造の整合性が確認されます。整合性エラーがあった場合は、それらのエラーが Essbase によって VALIDATE ログに記録されます。エラー・ログは、VALIDATE コマンドで指定したファイルに保持されます。

VALIDATE でインデックスの空きスペース情報に関連した整合性エラーが検出された場合は、データベースを再構築する必要があります。再構築を行うには、次の 3 つの方法があります:

- 最新のシステム・バックアップからデータベースを復元します
- データベースからデータをエクスポートし、空のデータベースを作成して、エクスポートされたデータを新しいデータベースにロードすることによって、データを復元します。
- データベースを再構築します

第 56 章「データベースの再構築の最適化」および『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

VALIDATE を使用しない場合でも、読取り操作が実行される場合は常に Essbase によって一定の妥当性の検査が自動的に実行され、インデックスがデータと同期されていることが確認されます。

すべての読取り操作について、Essbase によってインデックス・ページ内のデータ・ブロック・キーと対応するデータ・ブロック内のデータ・ブロック・キーが比較され、ブロック内の他のヘッダー情報が確認されます。

Essbase で不一致が検出されると、「無効なブロック・ヘッダー」というエラー・メッセージが表示されます。

クラッシュしたデータベースのリカバリ

クラッシュなどのサーバーの中断の後、Essbase によってデータベースがリカバリされ、中断の発生時にアクティブであったすべてのトランザクションがロール・

バックされます。リカバリ時間は、インデックスのサイズによって異なります。インデックスが大きいほど、時間がかかります。

また、Essbase によって、空きフラグメント(データ・ブロック内の未使用のアドレス可能な単位)もリカバリおよび統合されます。ただし、空きスペースのリカバリはデータベース・リカバリにおいて最も時間のかかる処理であるため、デフォルトでは後回しにされます。空きスペースのリカバリは、デフォルト設定を変更していないかぎり明示的にトリガーする必要があります。空きスペースのリカバリを遅らせることのメリットとデメリットについては、874 ページの「空きスペースのリカバリ」を参照してください。

Essbase では、サーバーの中断の後、サーバーが起動されるとすぐにデータがリカバリされます。リカバリのフェーズは次のとおりです:

1. トランザクション・リカバリ処理により、サーバー中断時にアクティブであったすべてのトランザクションがロール・バックされます。
2. インデックス・ファイル・リカバリ処理により、ファイルが、以前にコミットされたときのサイズに切り捨てられます。
3. データ空きスペース・リカバリ処理により、データ空きスペース・テーブルが再構築されます。このフェーズの所要期間は、インデックスのサイズによって決まります。

注： 空きスペースのリカバリは、デフォルト設定を変更していない場合、トリガーするまで行われません。

メディア障害(欠陥のあるディスク、ディスク障害またはヘッド・クラッシュ)が発生した場合は、バックアップからデータを復元する必要があります。『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

注意 `essxxxxx.ind`、`essxxxxx.pag`、`dbname.ind`、`dbname.esm`、`dbname.tct` の各ファイルは移動、コピー、変更または削除しないでください。データが破損する場合があります。

Essbase カーネルは、致命的なエラーの処理を使用することにより、発生したエラーに応じて適切なメッセージを表示したり、サーバーをシャット・ダウンしたりします。致命的なエラーの処理の動作については、1155 ページの「致命的なエラーの処理の理解」を参照してください。

クラッシュの後にトランザクションがロール・バックされる方法については、868 ページの「コミット・アクセスとアンコミット・アクセスの比較」を参照してください。

空きスペースのリカバリ

データベース・リカバリは、直前にクラッシュしたか、または異常終了したアプリケーションをロードしたときに実行されます。空きスペースのリカバリは、データベース・リカバリの中で最もコストの高い部分であるため、Essbase によって自

動的には実行されません。空きスペースのリカバリを明示的にトリガーするか、またはEssbaseによって空きスペースが自動的にリカバリされるようにデフォルト設定を変更する必要があります。

空きスペースをリカバリするかどうかにかかわらず、すべてのデータベース機能が正常に実行されます。空きスペースをリカバリした場合は、データ・ファイル内の空きとしてマークされたディスク・スペースを再利用できます。空きスペースのリカバリには時間がかかるため、より適切な時期まで遅らせることもできます。

ただし、データ・ファイル内の空きスペースを利用したり、データベースが破損していないことを確認したりするためには、できるだけ早く空きスペースのリカバリを実行する必要があります。また、データベースが繰り返しクラッシュしていて、空きスペースのリカバリを実行していない場合は、データ・ファイルが不必要に大きくなっている可能性があります。

空きスペースのリカバリをトリガーするには、MaxLのalter databaseコマンドを使用します。例:

```
alter database DBS-NAME recover freespace
```

『テクニカル・リファレンス』を参照してください。

空きスペースのリカバリに対するデフォルトの動作を変更するには、DELAYEDRECOVERY構成設定をFALSEに変更します。『テクニカル・リファレンス』の「設定の構成」を参照してください。

空きスペースのリカバリに関する情報を取得するには、GETDBSTATSコマンドを使用します。GETDBSTATSでは、空きスペースのリカバリに関する次の情報が提供されます:

```
   Free Space is Recoverable         : true/false
Estimated Bytes of Recoverable Free Space :
   nnn
```

注: 空きスペースのリカバリが可能な場合、ブロック・カウンタは概数であり、既存のブロック数とは一致しないことがあります。

サーバーの中断が発生した場合に予測される結果

表 160 は、Essbaseサーバーの中断のタイプとその結果を示しています:

表 160 Essbaseのリカバリ処理

中断	結果
• サーバーの停電 • オペレーティング・システムのクラッシュ • [Ctrl]+[C]によるサーバーの停止	Essbaseサーバーが停止します。サーバーを再起動すると、Essbaseによってデータベースがリカバリされます。

中断	結果
• システム制限のために操作を完了できない • メモリー不足 • ディスク・スペースの不足 • [Ctrl]+[C]によるアプリケーションの停止	Essbase により致命的なエラーの処理が実行されます。より多くのメモリーまたはディスク・スペースを割り当て、サーバーを再起動することが必要な場合があります。
サーバーのクラッシュ	Essbase 例外マネージャによって、タイプ.xcp のエラー・ログが作成されます。サーバーが停止します。サーバーを再起動すると、Essbase によってデータベースがリカバリされます。

表 161 は、トランザクション中にサーバーの中断が発生した場合に実行する必要のある手順を示しています。Essbase が中断からリカバリする方法は、トランザクションの分離レベル設定(コミット・アクセスまたはアンコミット・アクセス)によって異なります。866 ページの「コミット・アクセスでのロールバック」および 868 ページの「アンコミット・アクセスでのロールバック」を参照してください。

表 161　サーバー要求のリカバリ手順

要求のタイプ	推奨手順
ロック (スプレッドシートの更新のため)	ロック・コマンドを再度発行します。
送信 (スプレッドシートの更新)	Essbase によってエラーが発行された場合は、最後の送信操作を繰り返します。 スプレッドシートが失われたか、または存在しないとき、SSAUDIT のスプレッドシート・ロギングを使用している場合は、dbname.atx ファイルを再ロードします。877 ページの「スプレッドシート更新ロギングの使用方法」を参照してください。
計算	サーバー・ログおよびアプリケーション・ログを確認して、計算がどこで中断しているかを確認します。817 ページの「Essbase サーバー・ログおよびアプリケーション・ログの表示」を参照してください。計算をやり直すかどうかを決定します。最後の計算を繰り返します。
データ・ロード	次のいずれかのアクションを行います: • 最後のデータ・ロードを繰り返します。第 20 章「データ・ロードまたは次元構築の実行およびデバッグ」を参照してください。 • エラー・ログをロードします。833 ページの「次元構築およびデータ・ロードのエラー・ログのロード」を参照してください。
算術データ・ロード(データベース内の値に対する加算または減算)	データベースがコミット・アクセスに設定されている場合は、データを再ロードします。(トランザクションはロール・バックされています。) データベースがアンコミット・アクセスに設定されている場合は、一部のデータがロードされているため、すべてのデータを再ロードすると、2 回ロードされたデータ値が原因で結果が不正確になります。そのため、次のアクションを実行します: • データベースの消去。 • ロード前の状態へのデータベースの復元。 • ロードの再実行。

要求のタイプ	推奨手順
再構築	再構築が完了していません。.pan、.inn および.otn の一時的な再構築ファイルを削除します。再構築を実行した最後の操作を繰り返します。

注： UPDATECALC パラメータが FALSE に設定されている場合は、計算中に中断が発生すると、Essbase によってデータベース全体が再計算されます。(このパラメータのデフォルト値は TRUE です。)

スプレッドシート更新ロギングの使用方法

データ損失に対する追加の保護およびスプレッドシート監査情報のために、Essbase には、スプレッドシート更新ロギングが用意されています。この機能は、サーバー上の essbase.cfg ファイル内の SSAUDIT または SSAUDITR パラメータで使用可能にできます。SSAUDIT は、サーバー上のすべてのデータベースに対して、または個別のデータベースに対して指定できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

通常の状況では、Essbase によってリカバリが処理されます。ただし、スプレッドシート更新ログを手動でロードすることもできます。たとえば、最新のバックアップから復元したが、そのバックアップ以降に加えられた変更を失いたくない場合や、メディア障害が発生した場合は、更新ログからトランザクションをリカバリできます。これには、Essbase サーバー・ウィンドウから、Essbase のコマンドライン機能である ESSCMD を使用します。

次の一連の ESSCMD コマンドで、更新ログがロードされます:

```
   LOGIN
   hostnode username password

SELECT
   appname dbname

LOADDATA 3
   filepath
   :
   appname
   .ATX
EXIT
```

更新ログのロードを簡略化するには、1195 ページの「バッチ・プロセスにおけるスクリプト・ファイルおよびバッチ・ファイルの使用」で説明されているバッチ・ファイルを準備します。

SSAUDIT または SSAUDITR が指定されている場合は、Essbase によって、スプレッドシート更新トランザクションが時間の経過順にログに記録されます。Essbase では、次の 2 つのファイルが使用されます:

- dbname.atx には、スプレッドシート更新トランザクションが、データ・ロードの入力ソースとして使用できる 1 つの単位として保管されます。

- dbname.alg には、各トランザクションの履歴情報(ユーザー名、日付、タイムスタンプなど)と、.atx ファイルのトランザクション行数が含まれています。

両ファイルはサーバーに保管されます。

スプレッドシート更新ログはかなり大きくなることがあり、SSAUDITR を使用している場合でも、Essbase ではデータがバックアップされた後でしかログが消去されません。スプレッドシート更新を頻繁に行う場合は、ログを手動で定期的に削除することを検討してください。

スプレッドシート・ロギングが使用可能になっている場合は、シャットダウン後にデータベースを起動すると、Essbase によって次のメッセージがデータベース・ログに書き込まれます:

```
   Starting Spreadsheet Log
volumename\app\
   appname
   \
   dbname
   \
   dbname
   .atx for database
   dbname
```

メッセージの例は次のとおりです:

```
   Starting Spreadsheet Log Hyperion\products\Essbase\EssbaseServer\app\app1\sample
\sample.atx for database Sample
```

スプレッドシート更新ロギングが正常に記録されるようにするため、アプリケーションを停止して再起動を行う前に、次のいずれかの処理を実行するようにしてください:

- 再構築が実行される任意の操作。第 56 章「データベースの再構築の最適化」を参照してください。
- 次のいずれかの ESSCMD コマンドの実行:
  - CREATEAPP
  - CREATEDB
  - COPYDB
  - RENAMEDB

Essbase では、スプレッドシート・ロギングを使用可能にすると、更新が実行された場合はその更新が必ず記録されます。なんらかの理由で、Essbase で更新ログに書き込めない場合は、Essbase によってトランザクションが停止され、エラー・メッセージが発行されます。

SSAUDIT および SSAUDITR は、essbase.cfg ファイル内でのみ指定できます。

ハイブリッド分析の問題に関する考慮事項

ハイブリッド分析によりリレーショナル・データベースを多次元データベースと統合できるため、下位レベルのメンバーとそれに関連付けられたデータはリレーショナル・データベース内に残し、上位レベルのメンバーとそれに関連付けられたデータを Essbase データベース内に置くことができます。このオプションによって、データの一貫性と整合性に関連した追加の問題が発生します。

データがすべての場所で正しいことを確認するための方法については、641 ページの「ハイブリッド分析でのデータの一貫性の管理」を参照してください。

52 MaxLデータ定義言語の使用

この章の内容

『Oracle Essbase テクニカル・リファレンス』の「MaxL DDL」の項も参照してください。

MaxL DDL 言語

MaxL データ定義言語は、複雑な引数を含む一連のコマンドではなく、ステートメントを使用して Essbase への管理要求を行うためのインタフェースです。

MaxL を使用すると、Essbase データベースに対する管理操作を自動化できます。MaxL スクリプトは、カスタマイズおよび再利用が可能になるように変数を使用して記述できます。

Essbase で MaxL ステートメントを受け取って解析できるようにするには、MaxL シェル(essmsh)、Administration Services またはカスタマイズされた Perl プログラム(Perl プログラムにステートメントを埋め込むことができるようにする MaxL Perl モジュールを使用)のいずれかを使用して、これらのステートメントを Essbase サーバーに「渡す」必要があります。

ステートメントの概要

MaxL DDL スクリプトおよび対話的なセッションはすべて、ログインと一連のステートメントで構成されています。ステートメントは、それぞれがセミコロンで終了し、文法的なシーケンスに従ったキーワードや変数から成ります。ステートメントは英語の文に似ています。たとえば、次のようになります:

```
create or replace user
user-name
 identified by
password
;
```

MaxL ステートメントは、実行する操作のタイプを示す、create や alter などの動詞で始まります。次に、操作対象の Essbase 要素を示す、database や user などのアーティファクトを指定します。ステートメントの残りの部分では、文法的に正しいシーケンスに従ったステートメントの部品(トークン)を使用して、実行するアクションに関する詳細情報を示します。

文法要件の全体的な概要、および MaxL ステートメントの動詞/アーティファクトの構造については、『Oracle Essbase テクニカル・リファレンス』の MaxL DDL の項を参照してください。

ステートメントの構成要素

MaxL パーサーは、ステートメントの構成要素であるトークンの順序付けられた表現を認識し、これを必要とします。トークンとは、MaxL によって単一の読取り可能な単位として認識される、スペースで区切られた有効な文字のシーケンスのことです。トークンは次のいずれかです:

- キーワード
- 名前
- 文字列
- 数字

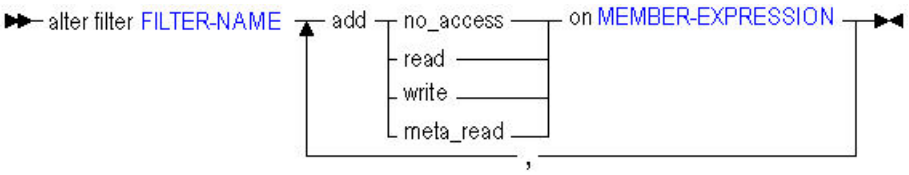
キーワード

キーワードは、MaxL の語彙を構成している予約語です。これには、動詞やアーティファクトのほか、ステートメントを構成するために必要なその他の単語が含まれます。MaxL でのキーワードは、データには無関係です。逆に言えば、その他の MaxL トークン(名前など)はすべて定義する必要があります。

MaxL パーサーでは、『Oracle Essbase テクニカル・リファレンス』の MaxL のトピックに図示されているように、MaxL キーワードやその他のトークンが文法的に正しい順序で配置されていることが要求されます。

図 150 は、『Oracle Essbase テクニカル・リファレンス』にあるサンプルの構文図を示しています。この図にある小文字の単語のみがキーワードを表しています。他の要素は、名前のプレースホルダか、またはユーザーによって指定される値です。

図 150　MaxL 構文図の例: Alter Filter



注： キーワードでは大文字と小文字が区別されません。キーワードに小文字を使用するのは、文書作成上の規則です。『Oracle Essbase テクニカル・リファレンス』の MaxL 線路図の解釈方法に関する項を参照してください。

名前

MaxL での名前は、データベースや、ユーザー、アプリケーション、フィルタなどのデータベース・アーティファクトを一意に識別するために使用されます。

名前のルール

一重引用符で囲まれている場合を除き、MaxL の名前は、英字またはアンダースコアで始まる文字列である必要があります。引用符で囲まれていない名前には、英字、数字およびアンダースコアのみを含めることができます。

名前を一重引用符で囲む場合は、名前に、スペースと次に示す任意の特殊文字を含めることができます:

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

注： MaxL キーワードでもある任意の名前は、一重引用符で囲む必要があります。キーワードのリストは、『Oracle Essbase テクニカル・リファレンス』を参照してください。

例:

次のアプリケーション名は、一重引用符で囲む必要はありません:

```
   Orange
Orange22
_Orange
```

次のアプリケーション名は、一重引用符で囲む必要があります:

```
   Orange County(because the name contains a space)
22Orange (because the name begins with a number)
variable (because the name is a MaxL keyword)
```

名前のタイプ

一部の Essbase アーティファクトはシングル名を持っていますが、ダブルやトリプルと呼ばれる、ネームスペースのネストを表す複合名が必要なものもあります。

シングル名は、システム全体にわたるコンテキストで意味を持つ場合があります。この名前が指すアーティファクトは、Essbase に対してグローバルである可能性があるか、または特定のアプリケーションまたはデータベース・コンテキストを必要としません。たとえば、シングル名を持つアプリケーションは、別のアプリケーションまたはデータベースのコンテキストで考慮される必要がない場合があります。

ダブルはピリオドで接続された 2 つの名前であり、トリプルは 2 つのピリオドで接続された 3 つの名前です。ダブルとトリプルは、名前付きエンティティの継承されたネームスペースを示します。たとえば、データベースは通常、2 つの名前を使用して識別されます。最初の名前でそのデータベースが存在するアプリケーションを識別し、2 番目の名前がデータベース名になります。たとえば、次のようになります:

```
   Sample.Basic
```

データベース・アーティファクト(フィルタなど)は通常、トリプルの名前を使用して識別されます。最初の 2 つの名前でアプリケーションとデータベースを識別し、3 番目の名前がアーティファクト名になります。そのため、フィルタ名は次のようになります:

```
   sample.basic.filter3
```

表 162 は、最も一般的なアーティファクトに必要な名前のタイプと、ステートメントで使用される名前の例を示しています。

表 162　アーティファクトの名前の要件

アーティファクト	名前	例
ユーザー	シングル	create user **Fiona** identified by 'password';
グループ	シングル	alter user Fiona add to group **Managers** ;
ホスト	シングル	drop replicated partition Samppart.Company from Sampeast.East at **EastHost** ;
アプリケーション	シングル	create application **'&New App'** ;
データベース	ダブル	display database **'&New App'.testdb** ;
計算	トリプル	drop calculation **Sample.basic.'alloc.csc'** ;
フィルタ	トリプル	display filter row **sample.basic.filter1** ;
関数(ローカル)	ダブル	drop function **sample.'@COVARIANCE'** ;
関数(グローバル)	シングル	create function **'@JSUM'** as 'CalcFnc.sum';

アーティファクト	名前	例
ロケーション別名	トリプル	drop location alias **Main.Sales.EasternDB** ;
役割	シングル	grant **designer** on database Sample.basic to Fiona;
代替変数	シングル	alter system add variable **Current_month** ; alter system set variable **Current_month** July;
ディスク・ボリューム	シングル(定義する場合)、トリプル(表示する場合)	alter database AP.main1 add disk volume **G** ; alter database AP.main1 set disk volume **G** partition_size 200mb; display disk volume **sample.basic.C** ;

文字列

文字列は、MaxL ステートメント内で、コメント、メンバー式、計算スクリプトおよびファイル参照のテキストを表すために使用されます。文字列は、任意の有効な文字で始めることができます。名前と同様に、空白または特殊文字を含む文字列は一重引用符で囲む必要があります。

有効な特殊文字のリストについては、883 ページの「名前のルール」を参照してください。

表 163 は、文字列であるステートメント要素の例を示しています:

表 163　MaxL ステートメント内の文字列の例

文字列のタイプ	例
パスワード	create user Fiona identified by **sunflower** ;

文字列のタイプ	例
コメント	alter group Financial comment **'Reports due July 31'** ;
メンバー式	create filter sample.basic.filt1 read on 'Sales, **@ATTRIBUTE(Bottle)** '; create or replace replicated partition sampeast.east area **'@IDESC(East), @IDESC(Qtr1)'** to samppart.company mapped globally '("Eastern Region")' to '(East)';
計算の本文	execute calculation **'"Variance"=@VAR(Actual, Budget);** **"Variance %"=@VARPER(Actual, Budget);'** on Sample.basic;
ファイル参照	spool on to **'/homes/fiona/out.txt'** ;

数値

数値は、MaxL ステートメント内で、Essbase 内の特定のデータベース設定を変更するために使用します。たとえば、キャッシュやバッファのサイズを変更したり、ユーザーがパスワードの変更を求められるまでの経過日数などのシステム全体にわたる間隔を設定することができます。数値の設定を変更する場合、正の整数、正の実数および 0 を使用できます。小数点および科学記号を使用できます。

例:

```
   1000
2.2
645e-2
```

サイズの設定の場合は、数値の後に単位を付ける必要があります。数値と単位の間のスペースはオプションです。単位では大文字と小文字が区別されず、B/b (バイト)、KB/kb (キロバイト)、MB/mb (メガバイト)、GB/gb (ギガバイト)および TB/tb (テラバイト)を含めることができます。単位を指定しない場合は、バイトと見なされます。

例:

```
   1000 b
5.5GB
645e-2 mb
145 KB
2,000e-3TB
```

ステートメントの分析

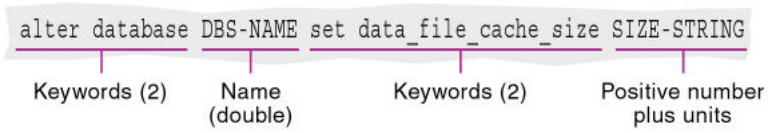
サブトピック

- データベースの変更
- 権限の付与
- 計算スクリプトの作成

この項では、MaxL ステートメントとその構成要素をテンプレート形式で図示することによって、ステートメントについて説明してきた内容を確認できるようにしています。次の各図の例に示すように、図では小文字の単語がキーワードであり、大文字の単語は適切な値で置き換えられます。

データベースの変更

図 151　データ・ファイルのキャッシュ・サイズを変更する MaxL ステートメント

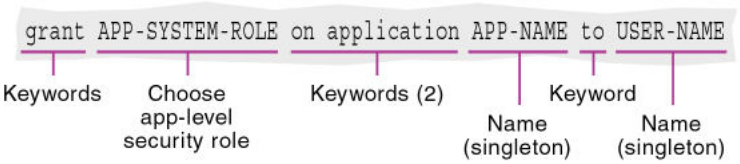


例:

```
   alter database Sample.Basic set data_file_cache_size 32768KB;
```

権限の付与

図 152　ユーザーにアプリケーション権限を付与する MaxL ステートメント

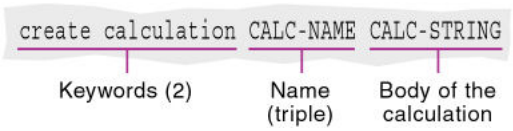


例:

```
grant designer on application Sample to Fiona;
```

計算スクリプトの作成

図 153　保管される計算を作成する MaxL ステートメント



例:

```
    create calculation sample.basic.varcalc
'"Variance"=@VAR(Actual, Budget);
"Variance %"=@VARPER(Actual, Budget);'
;
```

MaxL シェル

サブトピック

- MaxL シェルの開始
- Essbase へのログイン
- コマンド・シェルの機能
- MaxL シェルの停止

この項では、MaxL シェルのほとんどの機能を使用して操作を開始する方法について説明します。『Oracle Essbase テクニカル・リファレンス』も参照してください。

この項では、Administration Services の MaxL スクリプト・エディタの使用については説明していません。Oracle Essbase Administration Services Online Help を参照してください。

MaxL シェルの開始

MaxL シェルを起動して入力を取得する方法には、次のものがあります:

- キーワードを使用した対話式入力
- MaxL スクリプト・ファイルからの入力(コマンドラインで指定されたファイルからステートメントが読み取られる)
- 別のプログラムの出力から MaxL シェルにパイプ接続された標準入力からの入力

MaxL シェルではまた、起動時にコマンドライン引数を受け入れられます。この機能は、任意の名前またはパスワードを表す位置パラメータに使用できます。

対話式入力の場合のシェルの起動

➤ MaxL ステートメントをコマンドラインで対話式に入力するには、オペレーティング・システムのプロンプトでシェルを呼び出します。

例:

```
essmsh
```

➤ 呼出し時、ログイン後に MaxL ステートメントを対話式に入力するには、-l フラグを使用します。

例:

```
essmsh -l Admin password
...
49 - User logged in: [Admin].
```

➤ MaxL ステートメントを対話式に入力し、対話式セッションで使用する変数を表すコマンドライン引数を指定するには、-a フラグを使用します。

例:

```
essmsh -a Admin password Sample.Basic
...
login $1 $2;

49 - User logged in: [admin].

alter database $3.$4 enable aggregate_missing;

72 - Database altered: ['sample'.'basic'].
```

この例では、$1、$2、$3 および$4 は定位置パラメータ変数であり、呼出し時に essmsh の後に入力された引数が入力された順に表します。

ファイル入力の場合のシェルの起動

➤ MaxL スクリプト・ファイルからの入力を受け入れるように MaxL シェルを呼び出すには、essmsh の後に、現在のディレクトリ内の MaxL スクリプトの名

前を入力するか、または別のディレクトリにある MaxL スクリプトのフルパスとファイル名を入力します。

ファイル名のみを指定すると、MaxL シェルは、そのファイルが現在のディレクトリ(オペレーティング・システムのコマンド・プロンプトが、essmsh が呼び出された時点で格納されているディレクトリ)内にあると見なします。次の呼出しの例では、ファイル maxlscript.msh は C:\の中に存在する必要があります。

```
C:\> essmsh maxlscript.msh
```

MaxL スクリプトが現在のディレクトリ内に存在しない場合は、MaxL スクリプトへのパスを指定します。絶対パスまたは相対パスを使用できます。

例:

```
$ essmsh ../Oracle/Middleware/EPMSystem11R1/products/Essbase/EssbaseServer/
test.msh
```

注： MaxL スクリプトは、特定の拡張子、つまりどの拡張子も、必要ありません。このドキュメントでは.msh を使用します。

UNIX シェルでは、ファイル読取りエラーを防ぐために、パスを一重引用符で囲んでください。

Windows コマンド・プロンプトでは、スクリプトへのパスにスペースが含まれている場合、ファイル読取りエラーを防ぐために、パス全体とファイル名を二重引用符で囲む必要があります。

プログラム入力の場合のシェルの起動

➤ 別のプログラムまたはプロセスの標準出力からの入力を受け入れるように MaxL シェルを呼び出すには、-i フラグを使用します。

例:

```
program.sh | essmsh -i
```

シェル・スクリプト program.sh は、出力として MaxL ステートメントを生成する場合があります。シェル・スクリプト出力は essmsh -i にパイプされ、その出力が入力として使用されます。これにより、スクリプトの効果的な同時実行が可能になります。

Windows の場合の-i を使用した呼出し例

次の Windows バッチ・スクリプトは、その出力としてログイン・ステートメントと MaxL 表示ステートメントを生成します。-i フラグを使用して、その出力を、入力として essmsh(MaxL シェル)として使用できます。

```
   echo login admin password on localhost; display privilege user;|essmsh -i
```

管理者ユーザーがログインすると、すべてのユーザー権限が表示され、MaxL セッションが終了します。

UNIX の場合の-i を使用した呼出し例

次に示すシェル・スクリプトの部分では、**display application** ステートメントで、アプリケーション数としてゼロが戻されるかどうかがテストされ、システム上にアプリケーションが存在しないことが確認されます。

```
   if [ $(echo "display application;" |
essmsh -l admin password -i 2>&1 |
awk '/Records returned/ {print $7}' ) != "[0]." ]
then
 print "This test requires that there be no applications."
 print "Quitting."
 exit 2
fi
```

説明:

1. MaxL の構文が、UNIX echo コマンドの出力として、MaxL シェルの呼出しおよびログインにパイプされます。
2. MaxL セッションの結果が awk によってテストされ、空のシステムに **display application** を入力した場合に表示される、次のような MaxL ステータス・メッセージとのパターン照合が実行されます: `Records returned: [0]`。
3. Awk では、まず文字列`'Records returned: '`が照合され、次に`'[0].'`と等しいかどうかが確認されます。
4. $7(awk によって検出されたステータス文字列内の 5 番目のトークンを表す変数)が、`'[0].'`に等しい場合は、システム上にアプリケーションは存在しません。それ以外の場合は、$7 が`'[1].'`またはシステム上に存在するアプリケーション数に等しくなります。

呼出しオプションの詳細と例については、『Oracle Essbase テクニカル・リファレンス』を参照してください。呼出しに関する情報は、essmsh の「man ページ」にあります。man ページを表示するには、オペレーティング・システムのコマンド・プロンプトで `essmsh -h | more` と入力してください。

Essbase へのログイン

MaxL 言語インタプリタでは、MaxL ステートメントの解析を始める前に、Essbase セッションに接続する必要があります。Essbase への接続を確立するには、MaxL シェルを使用します。

➤ コマンド・シェルを開始した後で Essbase にログオンするには、シェルの login 文法を使用します。

例:

```
essmsh

MAXL>login Fiona identified by sunflower on hostname;
```

ホスト名を指定しない場合は、`localhost` が使用されます。

➤ シェルを呼び出すときにログオンするには、`-l` オプションを使用します。呼出し時に `localhost` 以外のサーバーにログオンするには、`-s` オプションを使用します。メッセージのレベルを設定するには、`-m` を使用します。

例:

```
essmsh -l fiona sunflower -s myHost -m error
```

注: シェルを終了しなくても、ログアウトおよびユーザーの変更を行うことができます。

ユーザーが MaxL シェル・セッションを介して Essbase にログインし、ユーザーのログイン中に Essbase を再起動すると、同じ MaxL シェル・セッションを介した次回のログインは 4 秒遅れます。

MaxL シェル呼出しオプションの詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

コマンド・シェルの機能

MaxL シェルには、コマンドライン引数の処理、環境変数の処理、MaxL スクリプトのネストおよびシェルのエスケープが組み込まれています。これらの機能は、高度に自動化された Essbase の本番環境を作り出すために必要な柔軟性を備えています。

『Oracle Essbase テクニカル・リファレンス』を参照してください。

MaxL スクリプトのネスト

DBA の場合、単一のスクリプトから多数の操作を実行するのではなく、自動化した個々のタスクをいくつかの MaxL スクリプトに保存する場合があります。ある MaxL スクリプトを別の MaxL スクリプトから参照する方法がわかっている場合、個々のピースをつなぎ合わせるのは簡単です。

➤ 現在の MaxL セッション内で他の MaxL スクリプトを参照または追加するには、次の MaxL シェル構文を使用します:

```
msh <scriptfile>;
```

注： msh はシェル・コマンドであるため、開始セッションに制限されます。したがって、新しいログイン・ステートメントを含む MaxL スクリプトは参照しないでください。

ファイルへの出力のスプール

出力をファイルにスプールして、MaxL セッションおよび関連メッセージのすべてまたはその一部のログ・ファイルを作成できます。

➤ MaxL セッションを記録するには:

1 **Essbase にログオンします。たとえば、**

```
login fiona sunflower;
```

2 **「spool on to <filename>」を使用して、出力のスプールを開始します。例:**

```
spool on to 'C:\\output\\display.txt';
```

3 **記録する MaxL ステートメントをすべて入力します。**

4 **「spool off」を使用して、出力のスプールを停止します。**

MaxL ステートメントとその出力は、spool コマンドを対話式セッションまたはスクリプトのいずれかで発行したときに、出力ファイルに記録されます。ただし、MaxL シェルのコマンドと出力は、対話式セッション中にスプールする場合のみ記録されます。MaxL シェルのコマンドと出力は、スクリプト・セッションから作成されたログ・ファイルでは無視されます。さらに、オペレーティング・システム・コマンドを含めた可能性のある出力は、対話式とスクリプトの両方のセッションのログ・ファイルで無視されます。

オペレーティング・システムのコマンドの追加

オペレーティング・システムのコマンドは、MaxL セッションから直接発行できます。オペレーティング・システムの出力は、MaxL シェル出力の一部になります。

オペレーティング・システムは、コマンドの実行を終了したときに、制御を essmsh に戻します。

- MaxL セッションからオペレーティング・システムにエスケープするには、**shell** を使用します。

たとえば、MaxL スクリプトから UNIX の **date** コマンドを実行するには、次を使用します:

```
shell date;
```

- MaxL セッションから ESSCMD にエスケープするには、一重引用符を使用します:

例:

```
shell esscmd ‘../scripts/test.scr’;
```

変数を使用した MaxL スクリプトのカスタマイズ

MaxL シェルでは、コンピュータの名前、ユーザー名またはパスワードなどの、変更される可能性のあるデータや頻繁に参照されるデータのプレースホルダとして、変数を使用できます。変数は、MaxL スクリプトで、対話式セッション中に使用できます。MaxL スクリプトで変数を使用すると、ユーザー、データベースまたはホストごとにカスタマイズしたスクリプトを作成する必要がなくなります。変数は、環境変数(たとえば、Essbase インストール・ディレクトリを参照先とする $ESSBASEPATH)、定位置パラメータ(たとえば、$1、$2、など)、およびローカルで定義したシェル変数である場合があります。変数は、参照するときは常に$(ドル符号)で開始します。

MaxL シェルでの変数の使用の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

MaxL シェルの停止

MaxL セッションからログアウトしたり、シェルを終了せずに別のユーザーとしてログオンできます。MaxL スクリプトの末尾には、ログアウト・ステートメントを指定する必要があります。MaxL スクリプト・ファイルの末尾で終了したり、または別のプログラムの出力からのストリーム指向の入力を使用してセッションの後に終了する必要はありません。

- MaxL シェルを終了しないでログアウトするには、次のコマンドを入力します:

```
logout;
```

➤ 対話式モードの使用後に MaxL シェルを終了するには、次のコマンドを入力します:

```
exit;
```

MaxL Perl モジュール

MaxL Perl モジュール(`Essbase.pm`)を使用すると、MaxL 言語を Perl プログラムに埋め込むことができ、これにより、シェルよりも詳細にプログラムを制御できるようになります。

`Essbase.pm` は `Perlmod` ディレクトリ内にあり、これにより Perl プログラマは、Perl スクリプトで MaxL ステートメントを折り返せます。MaxL によるデータベース管理は、Perl プログラムと同じくらい効率的で柔軟なものになります。

Essbase データベースを管理するときには、MaxL を含む Perl により、次の機能やその他のプログラム機能を活用できます:

- 条件付テスト
- プロセス間通信
- メッセージ処理
- 電子メール通知
- Web スクリプト

`Essbase.pm` には、Perl を使用して MaxL ステートメントを渡すことができるようにするメソッドが含まれています:

- **connect** ()は、Essbase への接続を確立します
- **do** ()は、カッコで囲まれた MaxL ステートメントを実行するように Perl に指示します
- **pop_msg** ()は、戻された MaxL メッセージのスタック内を移動します
- **fetch_col** ()、**fetch_desc** ()および **fetch_row** ()は、MaxL 表示出力テーブルから情報を取得します
- **disconnect** ()は、Essbase への接続を終了します

Perl メソッドを Essbase で使用できるようにするには、Perl プログラムに `Essbase.pm` への参照を組み込みます。各 Perl スクリプトの一番上に次の行を配置します:

```
use Essbase;
```

Perl は、UNIX シェルやその他のプログラミング言語の知識がある場合は特に、学習は難しくありません。Perl をダウンロードして、より詳しく学習するには、包括的な Perl アーカイブ・ネットワークの Web サイト(http://www.cpan.org/)にアクセスしてください。

『Oracle Essbase テクニカル・リファレンス』の MaxL DDL に関する項、および Essbase インストールの PERLMOD ディレクトリ内の README ファイルを参照してください。

第 IX 部
Essbaseの最適化

Essbase の最適化の内容：

53 パフォーマンスの監視

この章の内容

関連項目:

- 800 ページの「Essbase ログの使用」
- 『Oracle Essbase エラー・メッセージ・リファレンス』

診断情報の取得

Essbase には、Essbase サーバー、アプリケーション、およびデータベース・レベルのパフォーマンス情報ダイアログ・ボックスがあります。次のタスクを実行する前には、パフォーマンス情報を確認してください:

- データの移行準備
- ユーザーの追加
- パフォーマンスの問題の分析
- その他の管理タスクの実行

Essbase では、情報がスナップショット・ベースで表示されます。最新情報を表示するには、「リフレッシュ」ボタンをクリックします。「リフレッシュ」ボタンがウィンドウまたはダイアログ・ボックスに表示されている場合は、それによってウィンドウまたはダイアログ・ボックスのすべてのタブが更新されます。

Essbase サーバー、アプリケーションおよびアウトライン・ログの詳細は、801 ページの「Essbase サーバー・ログおよびアプリケーション・ログ」を参照してください。

次の項では、Essbase サーバー、アプリケーションおよびデータベースの情報にアクセスするための詳細な手順を示します。これらの手順は、通常はパフォーマンスやその他の問題の診断に使用されます。

Essbase サーバー情報の表示

Essbase サーバーのライセンス、構成、オペレーティング・システム、ディスク・ドライブおよび Essbase サーバーのアプリケーションに関する情報を表示できます。

➤ Essbase サーバー情報を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	Essbase サーバーの監視	Oracle Essbase Administration Services Online Help
MaxL	**display application**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	GETAPPSTATE	『Oracle Essbase テクニカル・リファレンス』

アプリケーション情報の表示

アプリケーション情報を表示することにより、アプリケーション内で動作中のデータベースを確認できます。また、アクセス、セキュリティおよび起動に関する情報を確認できます。

➤ アプリケーション情報を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アプリケーションの監視	Oracle Essbase Administration Services Online Help
MaxL	**display application**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	GETAPPSTATE	『Oracle Essbase テクニカル・リファレンス』

データベース情報の表示

データベース・ストレージ、データベース統計およびロック競合に関する情報を表示できます。この情報は、パフォーマンスに影響を及ぼすアクティビティや操作の識別に役立つことがあります。

➤ データベース情報を表示するには、次のツールを使用します:

ツール	指示	参照
Administration Services	データベースの監視	Oracle Essbase Administration Services Online Help
MaxL	**display database**	『Oracle Essbase テクニカル・リファレンス』

ツール	指示	参照
ESSCMD	• 一般情報: GETDBINFO • データベース・ストレージ情報: GETDBSTATE • 通貨情報: GETCRDBINFO • データベース統計: GETDBSTATS • ランタイム情報: GETDBINFO	『Oracle Essbase テクニカル・リファレンス』

アプリケーションおよびデータベースのステータスの監視

使用権限のあるアプリケーションおよびデータベースの、開始/停止ステータスを表示できます。

➤ アプリケーション/データベースのステータスを表示するには、次のツールを使用します:

ツール	指示	参考資料
Administration Services	アプリケーション/データベース・ステータスの表示	Oracle Essbase Administration Services Online Help
ESSCMD	GETAPPINFO GETDBINFO	『Oracle Essbase テクニカル・リファレンス』

ユーザー・セッションおよび要求の監視

Essbase サーバー、アプリケーションまたはデータベースのアクティブなユーザー・セッションを監視できます。管理者またはアプリケーション・マネージャ権限を持っている場合は、ユーザー・セッションを切断したり、セッション中に行われた要求を終了できます。687 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの切断および要求の終了」を参照してください。

➤ ユーザー・セッションおよび要求を監視するには、次のツールを使用します:

ツール	指示	参考資料
Administration Services	アクティブなユーザー・セッションの表示	Oracle Essbase Administration Services Online Help
MaxL	**display session alter system**	『Oracle Essbase テクニカル・リファレンス』

オペレーティング・システムからのアプリケーションの監視

ロードされている各アプリケーションは、オペレーティング・システムのオープンなタスクまたはプロセスです。オペレーティング・システムを使用して、次のようにアプリケーションのタスクまたはプロセスを表示できます:

- Windows プラットフォームの場合、アプリケーションはアプリケーション・サーバーのウィンドウに表示されます。このウィンドウで、アプリケーションのアクティビティをその発生時に確認できます。エージェントがアプリケーションを起動すると、タスクバーに新しいアイコンが表示されます。このアイコンをダブルクリックすると、アプリケーション・サーバーのウィンドウが表示されます。
- UNIX プラットフォームでは、たいていの場合アプリケーション・サーバーはバックグラウンド・プロセスです。アプリケーションが起動すると、ESSBASE コマンドによって ESSSVR プロセスが開始されます。アクティビティを表示するには、`tail -f log` コマンドを使用して、すべてのメッセージをファイルに送ります。ここで、`log` は、ユーザーが指定するファイルの名前です。
- また、Administration Services を使用して、Essbase サーバーまたはアプリケーションのスナップショットを表示することもできます。ログの表示、フィルタ処理、検索および分析の詳細は、Oracle Essbase Administration Services Online Help を参照してください。サーバーおよびアプリケーション・ログの詳細は、801 ページの「Essbase サーバー・ログおよびアプリケーション・ログ」を参照してください。

54 Essbaseのパフォーマンスの向上

この章の内容

最適化に影響する基本的な設計問題の認識

次のリストのコンテンツ(いずれも他のボリュームに掲載)を参照して、最適化に影響を及ぼす基本設計の問題を特定します:

- 基本的な概念、およびそれらの最適なパフォーマンスとの関係の導入については、第 3 章「多次元データベースの理解」を参照してください。
- 計画段階でデータベースの設計を検討する方法、およびこの分析が最適化にもたらす効果については、77 ページの「分析とプランニング」を参照してください。
- 基本的なデータベース・アウトラインの問題がパフォーマンスにどのように影響するかを理解するには、91 ページの「パフォーマンスを最適化するためのアウトラインの設計」を参照してください。

データベースのリセットによるパフォーマンスの向上

データベースを定期的にリセットし、後で再ロードできます。データベースを頻繁に再ロードしても、データベースをリセットしないかぎり、メインのデータベース・ファイル(.pag)が大きくなる可能性があります。

➤ データベースをリセットするには、次のツールを使用します:

ツール	トピック	場所
MaxL	**alter database appname.dbname reset**	『Oracle Essbase テクニカル・リファレンス』

ツール	トピック	場所
ESSCMD	RESETDB	『Oracle Essbase テクニカル・リファレンス』

データベース設定を使用する、パフォーマンス最大化のためのカスタマイズ

Administration Services、ESSCMD、または MaxL のデータベース・レベルでのデータベース設定を使用して、最大のパフォーマンスを得るために Essbase をカスタマイズできます。

注： データベースを移行する場合の移行後のデフォルト設定の詳細は、Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

次の項では、パフォーマンス設定をリストし、その調整方法について説明します。

データベース・キャッシュの設定

表 164 に、データベース・キャッシュ設定の説明と、Administration Services、MaxL および ESSCMD でのこの設定の場所の一覧を示します:

表 164　データベース・キャッシュの設定

設定	トピック	場所
インデックス・キャッシュ・サイズ	917 ページの「インデックス・キャッシュ・サイズの設定」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「キャッシュ」タブ • MaxL: **alter database appname.dbname set index_cache_size n** • ESSCMD: SETDBSTATEITEM 12
データ・ファイル・キャッシュ・サイズ	918 ページの「データ・ファイル・キャッシュ・サイズの設定」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「キャッシュ」タブ • MaxL: **alter database appname.dbname set data_file_cache_size n** • ESSCMD: SETDBSTATEITEM 27
データ・キャッシュ・サイズ	919 ページの「データ・キャッシュ・サイズの設定」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「キャッシュ」タブ • MaxL: **alter database appname.dbname set data_cache_size n** • ESSCMD: SETDBSTATEITEM 5
インデックス・ページ・サイズ	固定サイズ	該当なし

設定	トピック	場所
キャッシュ・メモリーのロック	916 ページの「キャッシュ・メモリーのロックを使用するかどうかの決定」	● Administration Services: 「データベース・プロパティ」ウィンドウ、「キャッシュ」タブ ● MaxL: **alter database appname.dbname enable cache_pinning** ● ESSCMD: SETDBSTATEITEM 26

Oracle Essbase Administration Services Online Help の「データベース・プロパティの設定」を参照してください。

データベース・ディスク・ボリュームの設定

表 165 に、データベース・ディスク・ボリュームの設定の説明と、Administration Services、MaxL および ESSCMD でのこれらの設定の場所の一覧を示します:

表 165　データベース・ディスク・ボリュームの設定

設定	トピック	場所
ボリューム名	848 ページの「ディスク・ボリュームの指定」	● Administration Services: 「データベース・プロパティ」ウィンドウ、「ストレージ」タブ ● MaxL: **alter database appname.dbname set disk volume** ● ESSCMD: SETDBSTATEITEM 23 SETDBSTATEITEM 24
パーティション・サイズ	848 ページの「ディスク・ボリュームの指定」	● Administration Services: 「データベース・プロパティ」ウィンドウ、「ストレージ」タブ ● MaxL: **alter database appname.dbname set disk volume** ● ESSCMD: SETDBSTATEITEM 23 SETDBSTATEITEM 24
ファイル・タイプ	848 ページの「ディスク・ボリュームの指定」	● Administration Services: 「データベース・プロパティ」ウィンドウ、「ストレージ」タブ ● MaxL: **alter database appname.dbname set disk volume** ● ESSCMD: SETDBSTATEITEM 23
最大ファイル・サイズ	848 ページの「ディスク・ボリュームの指定」	● Administration Services: 「データベース・プロパティ」ウィンドウ、「ストレージ」タブ ● MaxL: **alter database appname.dbname set disk volume** ● ESSCMD: SETDBSTATEITEM 23

Oracle Essbase Administration Services Online Help の「ディスク・ボリュームの設定」を参照してください。

データベース・トランザクション・コントロールの設定

表 166 に、データベース・トランザクション・コントロールの設定の説明と、Administration Services、MaxL および ESSCMD でのこれらの設定の場所の一覧を示します:

表 166　データベース・トランザクション・コントロールの設定

設定	トピック	場所
分離レベル	861 ページの「分離レベルの理解」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「トランザクション」タブ • MaxL: **alter database appname.dbname enable committed_mode** • ESSCMD: SETDBSTATEITEM 18
ブロックのコミット	861 ページの「分離レベルの理解」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「トランザクション」タブ • MaxL: **alter database appname.dbname enable committed_mode** および **alter database appname.dbname set implicit_commit after n blocks** • ESSCMD: SETDBSTATEITEM 21
行をコミット	861 ページの「分離レベルの理解」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「トランザクション」タブ • MaxL: **alter database appname.dbname enable committed_mode** および **alter database appname.dbname set implicit_commit after n rows** • ESSCMD: SETDBSTATEITEM 22
ロックされたデータ・ブロックへの書込みアクセスの待機	861 ページの「分離レベルの理解」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「トランザクション」タブ • MaxL: **alter database appname.dbname set lock_timeout** • ESSCMD: SETDBSTATEITEM 20
プリイメージ・アクセス	861 ページの「分離レベルの理解」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「トランザクション」タブ • MaxL: **alter database appname.dbname enable pre_image_access** • ESSCMD: SETDBSTATEITEM 19

Oracle Essbase Administration Services Online Help の「データの整合性オプションの設定」を参照してください。

その他のデータベース設定

表 167 に、その他のデータベース設定の説明と、Administration Services、MaxL および ESSCMD でのこれらの設定の場所の一覧を示します:

表 167　その他のデータベース設定

設定	トピック	場所
取得バッファ・サイズ	993 ページの「取得バッファのサイズの設定」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「全般」タブ • MaxL: **alter database appname.dbname set retrieve_buffer_size n** • ESSCMD SETDBSTATEITEM 16
取得ソート・バッファ・サイズ	994 ページの「取得ソート・バッファのサイズの設定」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「全般」タブ • MaxL: **alter database appname.dbname set retrieve_sort_ buffer_size n** • ESSCMD: SETDBSTATEITEM 17
データ圧縮	853 ページの「データ圧縮」	• Administration Services: 「データベース・プロパティ」ウィンドウ、「ストレージ」タブ • MaxL: **alter database appname.dbname enable compression** および **alter database appname.dbname set compression type** • ESSCMD: SETDBSTATEITEM 14 SETDBSTATEITEM 15
トリガー定義のための最大メモリー	117 ページの「トリガー定義の理解」	MaxL: **create or replace trigger**、**alter trigger display trigger** および **drop trigger**

Oracle Essbase Administration Services Online Help の「データベース・プロパティの設定」を参照してください。

断片化の削除および測定

断片化は、Essbase によってディスク上の新しい場所にデータ・ブロックが書き込まれ、そのデータ・ブロックの以前の場所に未使用のスペースが残ったときに発生します。ブロックにはデータ・ロードや計算によるデータが追加されるため、ブロックのサイズは大きくなります;そのため、ブロックをデータ・ファイルの末尾に書き込む必要があります。

Essbase カーネルは、未使用スペースが再使用される可能性が高くなるように、隣接する断片をより大きな断片の中にマージします。

断片化を完全に解消できない場合もあります。断片化は次のような状況で発生する可能性があります:

- ユーザーが常時データの更新を行う読取り/書込みデータベース
- 24 時間体制で計算を実行するデータベース
- 密メンバーを頻繁に更新および再計算するデータベース
- 適切に設計されていないデータ・ロード
- 多数の動的計算および保管メンバーを含むデータベース
- 「ブロックをコミット」設定が 0 で、アンコミット・アクセスの分離レベルを使用するデータベース

パフォーマンスが低下した場合、データベースの断片が多すぎるか確認し、多すぎる場合には、断片化を削減するための処置を取ることができます。

断片化の測定

平均断片化指数の統計または平均クラスタ率を使用して、断片化を測定できます。

平均断片化指数の使用:

ESSCMD では、GETDBSTATS コマンドを実行したときに戻される平均断片化指数を調べます。表 168 の情報を使用して、断片化のレベルがパフォーマンスの問題を引き起こす可能性があるかどうかを評価します:

表 168　平均断片化指数を使用した、断片化しきい値の測定

データベースのサイズ	断片化指数のしきい値
小(200MB 未満)	60%以上
中(2GB 未満)	40%以上
大(2GB を超える)	30%以上

指数が前述の範囲の上限を超えている場合、断片化の削減によってパフォーマンスを改善できる可能性があります。この場合、次の条件も考慮します:

- データベース上で他の書込みトランザクションが実行されていないときのほうが、報告される断片化指数の値はより正確になります。
- 直接入出力のアクセス・モードを使用する 50MB 未満のデータベースの場合、断片化指数は高くなる傾向があります。空きスペースは 8MB のチャンクで作成され、そのすべてが即時使用されるわけではないため、断片化指数が高くても、必ずしも断片化を削除する必要があるとは限りません。

平均クラスタ率の使用:

平均クラスタ率のデータベース統計は、データ・ファイル(.pag)の断片化レベルを示します。最大値は 1 で、断片化なしを示します。

➤ データベースの平均クラスタ率を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	断片化統計の表示	Oracle Essbase Administration Services Online Help
ESSCMD	GETDBSTATS	『Oracle Essbase テクニカル・リファレンス』

断片化の防止または削除

断片化は、次の方法で防止および削除できます:

- 断片化を防止するには、疎次元メンバーに基づいてロードをソートすることで、データ・ロードを最適化します。疎メンバーのグループ化によるデータの最適化の詳細は、946 ページの「疎メンバーの組合せのグループ化」を参照してください。
- 断片化を削除するには、データベースのエクスポートを実行し、CLEARDATA でデータベース内のすべてのデータを削除してから、エクスポート・ファイルを再ロードします。『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。
- 断片化を削除するには、強制的にデータベースの密な再構築を実行します。934 ページの「データベースの再構築のタイプ」を参照してください。

Windows 4 GB Tuning (4GT)の使用

Essbase では、Microsoft 4 GB Tuning (4GT)がサポートされています。4GT により、きわめて大規模なデータベースを持つユーザーが、より大きなアドレス・スペースを活用してパフォーマンスを向上させることができます。

Essbase インストールに次の特徴がある場合は、Windows 4GT 機能を使用可能にすると役立つことがあります:

- Essbase が直接入出力を使用するように構成されている場合。バッファ入出力用に構成された Essbase インストールで 4GT 機能を使用可能にすることは、お薦めしません。
- インデックス・キャッシュとデータ・キャッシュのサイズは正しく設定されていて、Essbase のパフォーマンスはデータ・ファイル・キャッシュを増やすことによって一貫して向上していますが、前の 2GB のアドレス可能度の制限により、さらに増やすには限界がある場合。キャッシュ値の設定の詳細は、第 55 章「Essbase キャッシュの最適化」を参照してください。

Microsoft Windows 4GT 機能の詳細は、http://www.microsoft.com を参照してください。

64 ビットの Essbase の実装

64 ビットのプロセスは、32 ビットのプロセスに比べてメモリーのアドレス可能度が大幅に増加しているため、64 ビット版の Essbase では 32 ビットの Essbase より大きなアウトラインやキャッシュ・サイズを処理できます。これをサポートするコンピュータ環境に 64 ビットの Essbase を実装することで、既存のアプリケーションのパフォーマンスを向上させることができ、非常に多くのアプリケーションを維持できます。64 ビットの Essbase のプラットフォーム・サポートの詳細は、Oracle Hyperion Enterprise Performance Management System 動作保証マトリックス(http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html)を参照してください。

64 ビットの Essbase では、キャッシュ・サイズを既存の 32 ビットの制限より大きく設定できます。Essbase クライアントでは、データ・キャッシュ、データ・ファイル・キャッシュおよびインデックス・キャッシュに設定できる最大値は 4 GB です。MEMSCALINGFACTOR 構成設定を使用することで、データ・キャッシュおよびデータ・ファイル・キャッシュにより大きな値が使用可能になります。『Oracle Essbase テクニカル・リファレンス』を参照してください。

64 ビットの Essbase では、32 ビットの Essbase よりも最大スレッド設定が高くなります。表 169 に、それぞれの最大スレッド設定の一覧を示します。スレッド設定の変更の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。マルチスレッド処理については、758 ページの「マルチスレッド処理」を参照してください。

表 169　32 ビットおよび 64 ビットの Essbase の最大スレッド設定

設定	32 ビットの最大値	64 ビットの最大値
AGENTTHREADS	500	1024
SERVERTHREADS	500	1024
DLTHREADSPREPARE	16	32
DLTHREADSWRITE	16	32

64 ビットの Essbase では、32 ビットの Essbase よりデフォルトの取得バッファ設定が高くなります。表 170 に、それぞれのデフォルトの取得バッファ設定の一覧を示します。993 ページの「バッファのサイズの変更」を参照してください。

表 170　32 ビットおよび 64 ビットの Essbase のデフォルトの取得バッファ設定

設定	32 ビットのデフォルト値	64 ビットのデフォルト値
取得バッファ	10 KB	20 KB
取得ソート・バッファ	10 KB	20 KB

注： 内部データ構造のサイズ変更のため、64 ビットの Essbase では、32 ビットの Essbase より大きな取得ソート・バッファ・サイズが必要です。`"Sort buffer limit of [x] rows have been exceeded"`(x は、現在のバッファ・サイズに許可されている現在最大行数)というエラーが発生した場合は、取得ソート・バッファ・サイズを 2 だけ大きくします。

その他の最適化情報

906 ページの「データベース設定を使用する、パフォーマンス最大化のためのカスタマイズ」は一般的な情報を示すもので、様々な構成機能については考慮していません。パフォーマンスやサーバー、アプリケーション、またはその他の設定の詳細は、次の各章を参照してください:

- 特定の機能のパフォーマンスを最適化する方法の詳細は、次の各章を参照してください:
  - 第 57 章「データ・ロードの最適化」
  - 第 58 章「計算の最適化」
  - 第 26 章「高機能計算についての理解」
  - 第 59 章「レポートおよび他の取得機能の最適化」
- Essbase カーネルの詳細は、次の章を参照してください:
  - 第 49 章「データベース設定の管理」
  - 第 50 章「ストレージの割当てとデータの圧縮」
  - 第 51 章「データの整合性の確保」
  - 第 55 章「Essbase キャッシュの最適化」
  - 付録 D「ディスク要件およびメモリー要件の見積り」

55 Essbaseキャッシュの最適化

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

Essbase キャッシュの理解

Essbase では、次の 5 つのメモリー・キャッシュを使用して、メモリーの使用率を調整します:

表 171　Essbase キャッシュのリスト

キャッシュ	説明
インデックス・キャッシュ	インデックス・ページを保持するメモリー内のバッファ。メモリー内に一度にいくつのインデックス・ページが保持されるかは、キャッシュに割り当てられたメモリー量によって異なります。
データ・ファイル・キャッシュ	圧縮データ・ファイル(.pag ファイル)を保持するメモリー内のバッファ。データ・ロード、計算および取得操作を行うと、必要に応じて Essbase によってメモリーがデータ・ファイル・キャッシュに割当てられます。データ・ファイル・キャッシュは、直接入出力が有効な場合にのみ使用されます。
データ・キャッシュ	非圧縮データ・ブロックを保持するメモリー内のバッファです。データ・ロード、計算および取得操作を行うと、必要に応じて Essbase によってメモリーがデータ・キャッシュに割当てられます。
計算機のキャッシュ	計算操作の際に、Essbase でデータ・ブロックを作成および追跡するために使用されるメモリー内のバッファ。
動的計算キャッシュ	密次元の動的計算メンバーの計算(クエリーなど)に必要なすべてのブロックを Essbase に保管するために使用される、メモリー内のブロック。

Essbase では、それぞれのキャッシュにデフォルトのサイズが設定されています。各データベースの必要に応じてサイズを調整できます。適切なキャッシュ・サイズは、データベース・サイズ、ブロック・サイズ、インデックス・サイズおよびサーバーの使用可能メモリーなど、多くの要因の影響を受けます。キャッシュ・

サイズの設定は、データベースやサーバーの全体的なパフォーマンスに大きく影響する可能性があります。

次のトピックで、最適なパフォーマンスを得るためのキャッシュのサイズ設定に関する情報を示します。

キャッシュ・メモリーのロックを使用するかどうかの決定

キャッシュ・サイズを設定する前に、キャッシュ・メモリーのロックを使用可能にするか、または設定を使用不可のままにしておく(デフォルト)必要があります。

キャッシュ・メモリーのロックを設定することによって、インデックス・キャッシュ、データ・ファイル・キャッシュおよびデータ・キャッシュに使用されるメモリーを物理メモリーにロックして、Essbase カーネルがシステム RAM を優先的に使用できるようにするかどうかを制御します。

キャッシュ・メモリーのロックを使用するには、直接入出力を使用している必要があります(デフォルトの入出力アクセス・モードは、バッファ入出力です)。直接入出力では、バッファ入出力に比べて大きなインデックス・キャッシュ・サイズが必要です。第 49 章「データベース設定の管理」を参照してください。

ロックによって、Essbase サーバーで使用されるメモリーのスワップ時に、システム・メモリー・マネージャでキャッシュが使用するメモリーをスワップする必要がなくなるため、Essbase データベースのパフォーマンスが向上します。デフォルトでは、キャッシュ・メモリーのロックはオフです。

キャッシュ・メモリーのロックを使用可能にすると、Essbase カーネルがシステム RAM を優先的に使用できるようになります。キャッシュ・メモリーのロックを使用可能にする場合は、Essbase カーネル以外が使用できるように、システム RAM の少なくとも 3 分の 1 を残してください。Essbase カーネルにシステム RAM を優先的に使用させない場合は、キャッシュ・メモリーのロックを使用可能にしないでください。

Solaris で Essbase を実行している場合は、Essbase を起動してキャッシュ・メモリーのロックを使用可能にする前に、Bourne シェル・スクリプト `root.sh` を実行します。このスクリプトは、メモリーをロックできるように、サーバーをスーパーユーザー・モードで実行するように設定します。Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

➤ キャッシュ・メモリーのロックを使用可能にするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	キャッシュ・メモリーのロックの有効化	Oracle Essbase Administration Services Online Help
MaxL	**alter database enable cache_pinning**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM 26	『Oracle Essbase テクニカル・リファレンス』

キャッシュのサイズ設定

構成可能な各キャッシュで使用する必要のある設定は、データ配分と、データベースの密/疎の構成によって異なります。キャッシュを合算した最大合計サイズは、Essbase で必要なメモリーを考慮に入れた後の使用可能なメモリー量に等しくなければなりません。

各サイトのニーズは、特定のデータベースであっても異なる場合があります。Essbase では、各操作の複雑さとタイプに応じて、データ・ファイル・キャッシュとデータ・キャッシュに必要とされる量のメモリーが割り当てられます。最適なパフォーマンスを得るために十分なメモリーを見積もるには、この項の推奨値を使用します。

初めて Essbase を使用する場合、キャッシュ・サイズは、次の項で説明するデフォルト値に自動的に設定されます。これらのトピックを使用して、各キャッシュ・サイズの推奨事項を調べて理解してください。

注： キャッシュ・サイズの変更は、データベースを次に起動したときに、有効になります。

インデックス・キャッシュ・サイズの設定

インデックスは、ディスク上のインデックス・ファイルに保管されます。データベースがアクティブであるときは、最後にアクセスされたインデックス・ページがインデックス・キャッシュ内に保持されます。メモリー内に同時に保持できるインデックスの量は、インデックス・キャッシュに割り当てるメモリーの量によって変わります。

注： インデックス・ページのサイズは、8 K に固定されています。これは、データベースの移行を簡単にすると同時に、入出力のオーバーヘッドを削減するためです。

インデックス・キャッシュ・サイズの有効性は、計算の性質によって異なります。たとえば、データベース全体を再ロードおよび再計算する場合(毎月リフレッシュされるデータベースなど)、インデックス・キャッシュ・サイズが大きいことは有益ではありません。Essbase で、計算中に既存のブロックのインデックス・キャッシュが検索されるのではなく、ブロックが作成されるためです。

表 172 に、インデックス・キャッシュのデフォルト設定および推奨の設定を示します:

表 172　インデックス・キャッシュ・サイズの設定

最小値	最大値	デフォルト値	推奨値
1MB (1,048,576 バイト)	32 ビットの Essbase: 4GB 64 ビットの Essbase: 256TB	バッファ I/O: 1MB (1,048,576 バイト) 直接 I/O: 10MB	可能な場合は、すべての `essn.ind` ファイルの合計サイズ。それ以外の場合は、可能なかぎり大きなサイズ。このキャッシュ・サイズを大きくしてもパフォーマンスは向上しないため、インデックス・サイズより大きいサイズに設定しないでください。インデックス・サイズの合計を決定するには、1169 ページの「インデックス・ファイル」を参照してください。

データベースに対する入出力アクセス・モードの変更、または新規作成したすべてのデータベースのデフォルトの変更の詳細については、836 ページの「バッファ I/Oと直接 I/O の理解」を参照してください。

通常、直接入出力を使用している場合は、インデックス・キャッシュをシステム・リソースで許容されるかぎり大きくします。バッファ入出力を使用している場合は、インデックス・キャッシュを可能なかぎり小さくします。

キャッシュ設定のテストと微調整の詳細については、929 ページの「キャッシュ設定の微調整」を参照してください。

インデックス・キャッシュ・サイズの変更

➤ インデックス・キャッシュのサイズを設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	キャッシュ・サイズの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database set index_cache_size**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM 12 SETDBSTATE	『Oracle Essbase テクニカル・リファレンス』

データ・ファイル・キャッシュ・サイズの設定

直接入出力を使用している場合、データ・ファイル・キャッシュはデータ・ファイル(.pag ファイル)をメモリー内に保持します。直接入出力を使用していない場合、データ・ファイル・キャッシュは使用されません。メモリー内に同時に収容できるデータ・ファイル内のデータの量は、データ・ファイル・キャッシュに割り当てるメモリーの量によって変わります。

一般に、データ・キャッシュとデータ・ファイル・キャッシュのどちらに対してメモリーを割り当てるかを選択する必要がある場合、直接入出力を使用しているときは、データ・ファイル・キャッシュを選択します。

表 173 に、データ・ファイル・キャッシュのデフォルト設定および推奨の設定を示します:

表 173　データ・ファイル・キャッシュ・サイズの設定

最小値	デフォルト値	推奨値
直接入出力: 10,240 KB (10,485,760 バイト)	直接入出力: 32,768KB(33,554,432 バイト)	可能な場合は、すべての `essn.pag` ファイルの合計サイズ。それ以外の場合は、可能なかぎり大きなサイズ。 Essbase がバッファ入出力を使用するように設定されている場合、このキャッシュ設定は使用されません。

通常、直接入出力を使用している場合は、データ・ファイル・キャッシュをシステム・リソースで許容されるかぎり大きくします。バッファ入出力を使用している場合、データ・ファイル・キャッシュは使用されません。

キャッシュ設定のテストと微調整の詳細については、929 ページの「キャッシュ設定の微調整」を参照してください。

データ・ファイル・キャッシュ・サイズの変更

➤ データ・ファイル・キャッシュ・サイズを設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	キャッシュ・サイズの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database set data_file_cache_size**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM 27	『Oracle Essbase テクニカル・リファレンス』

データ・キャッシュ・サイズの設定

データ・ブロックは、物理ディスク上とメモリー内にあります。データ・キャッシュ内に同時に保持できるブロックの数は、データ・キャッシュに割り当てるメモリーの量によって変わります。

ブロックが要求されると、Essbase ではデータ・キャッシュからそのブロックが検索されます。Essbase でキャッシュ内でブロックが見つかると、ただちにそのブロックがアクセスされます。キャッシュ内でブロックが見つからなかった場合は、Essbase では適切なブロック番号のインデックスが検索され、そのブロックのインデックス項目を使用して、ディスク上の適切なデータ・ファイルからブロックが取得されます。要求されたブロックをデータ・キャッシュから取得するほうが速いため、パフォーマンスが向上します。

一般に、データ・キャッシュとデータ・ファイル・キャッシュのどちらに対してメモリーを割り当てるかを選択する必要がある場合、直接入出力を使用しているときは、データ・ファイル・キャッシュを選択します。

表 174 に、データ・キャッシュのデフォルト設定および推奨の設定を示します:

表 174　データ・キャッシュ・サイズの設定

最小値	デフォルト値	推奨値
3072 KB (3145728 バイト)	3072 KB (3,145,728 バイト)	0.125 *データ・ファイル・キャッシュ・サイズの値。次のいずれかの状況が存在する場合は、値を大きくします: ● 多数の同時ユーザーが異なるデータ・ブロックにアクセスする場合。 ● 疎な範囲に対する関数が計算スクリプトに含まれていて、その関数で対象メンバーすべてがメモリー内に存在する必要がある場合(たとえば@RANK および@RANGE の使用時)。 ● データ・ロードにおいて、DLTHREADSWRITE 設定で指定されたスレッドの数が多く、展開されているブロックのサイズが大きい場合。

同時計算の実行時には、データ・キャッシュを増やすことが必要な場合があります。たとえば、同時計算が共通のブロックを共有せず、子の数が最も大きい疎メ

ンバーがそのすべての子ブロックをデータベース内に移入している場合です。データ・キャッシュを同時計算で計算するには、次の式を使用します:

```
   (Size of big block in bytes) * max(Number of children for a Sparse member) *
(Number of concurrent batch calc processes) * 2
```

同時計算の実行時にその他の同時操作(データ・ロードやクエリーなど)を行う場合は、これらの要求に適応するため、データ・キャッシュをさらに増やします。

データ・キャッシュは、バッファ入出力と直接入出力のいずれを使用する場合でも、できるかぎり小さくしてください。

キャッシュ設定のテストと微調整の詳細については、929 ページの「キャッシュ設定の微調整」を参照してください。

注： 64 ビット・プラットフォームで Essbase を実行している場合、最適なデータ・キャッシュとデータ・ファイル・キャッシュの設定は 4 GB より大きくなることがあります。Essbase クライアントで 4 GB より大きい設定は指定できませんが、MEMSCALINGFACTOR 構成設定を使用することで、より大きな設定を使用可能にできます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

データ・キャッシュ・サイズの変更

➤ データ・キャッシュ・サイズを設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	キャッシュ・サイズの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database set data_cache_size**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM 5 SETDBSTATE	『Oracle Essbase テクニカル・リファレンス』

計算機キャッシュのサイズ設定

Essbase では、サイズが計算機キャッシュのサイズによって制御されるビットマップを作成して、計算中にデータ・ブロックを記録して追跡できます。ビットマップを使用して存在するブロックを判別することは、データベースを初めて計算する場合や、データが疎であるときにデータベースを計算する場合は特に、ディスクにアクセスして情報を取得するよりも短時間で済みます。

Essbase では、データベースに少なくとも 2 つの疎次元があり、さらに次のいずれかの条件に該当する場合に、計算機キャッシュのビットマップが使用されます:

- 少なくとも 1 つの疎次元全体を計算する。
- 計算スクリプト内で SET CACHE ALL コマンドを指定する。

計算機キャッシュの最適なサイズは、アウトライン内の疎次元の数と密度によって異なります。計算機キャッシュのビットマップの理解、計算機キャッシュのサイズ設定、および計算機キャッシュのサイズ(すなわち、ビットマップの可能なかぎり大きなサイズ)の変更を行うには、必要に応じて次のトピックを使用してください:

計算機キャッシュのビットマップの理解

計算機キャッシュの場合、データベース内の密次元は Essbase によって次の 2 つのグループに分けられます:

- ビットマップ次元: データベース・アウトラインの疎次元のうち、Essbase によって、ビットマップがいっぱいになるまでビットマップに対応付けられているもの。ビットマップに配置された疎次元のメンバーの組合せは、それぞれが 1 ビットのメモリーを占有します。ビットマップ内には、疎次元のすべてのメンバーの組合せをビットマップに配置できるだけの十分なスペースが必要です。
- アンカー次元: ビットマップに対応付けされていない、データベース・アウトライン内のその他の疎次元。

Essbase では、データベース・アウトライン内の最初の疎次元から開始して、可能なかぎり多くの疎次元がビットマップに対応付けられます。対応付けられる次元はビットマップ次元です。別の疎次元全体をビットマップに対応付けできなくなると、Essbase によってプロセスが停止されます。ビットマップのサイズは計算機キャッシュによって制御されるため、ビットマップに対応付けられる疎次元の数は、計算機キャッシュ(および疎次元の数とサイズ)によって変わります。

残りの疎次元はアンカー次元です。アンカー次元の場合、ブロックが存在するかどうかを判別するために Essbase でビットマップを使用できません。

アンカー次元かビットマップ次元かを識別するには、SET MSG DETAIL 計算コマンドを使用して、アプリケーション・ログ内にビットマップの情報を表示します。

ビットマップ内に可能なかぎり多くの次元を配置できるように、アウトライン内の疎次元は慎重に配列します。含まれているメンバーが最も少ない次元から始めて、メンバーが最も多い次元が最後になるようにします。このような順序によって、より多くのビットマップを次元に対応付けられるようになり、結果として計算のパフォーマンスが向上します。

注: アウトライン内の疎次元の順序は、クエリーのパフォーマンスにも影響を与えます。92 ページの「クエリーのパフォーマンスの最適化」を参照してください。

複数のアンカー次元が存在する場合、または計算機キャッシュが複数のビットマップをサポートできるだけの大きさでない場合は、Essbase では単一のビットマップが使用されます。また、1 つのアンカー次元が存在する場合は、複数のビットマップが使用されます。

単一のビットマップには次のようなプロパティがあります:

- 単一ビットマップは、子ブロックの追跡に使用されます。

- 単一ビットマップでは、最小限のメモリーしか使用されませんが、複数ビットマップより効率が悪くなります。

複数のビットマップには、次のようなプロパティがあります:

- 複数のビットマップが使用され、子ブロックの追跡に 1 つ、親ブロックの追跡に 1 つのビットマップが使用されます。
- 複数のビットマップは、単一のビットマップを使用する場合よりも多くのメモリーを使用しますが、より高速になります。初めてデータベースを計算するときは、パフォーマンスが特に大幅に向上します。
- 使用されるビットマップの数は、アンカー次元のメンバーが依存している親の数によって決まります。メンバーには、共有メンバーを持っている場合を除き、1 つの従属する親があります。たとえば、Sample.Basic データベースの製品次元について考えてみてください。メンバー Cola (100-10)には Colas (100)という親が 1 つあります。ただし、Diet Cola (100-20)には、Diet Drinks (Diet)と Colas (100)という 2 つの親があります。製品のメンバーで、3 つ以上の依存する親を持つものはありません。したがって、製品がアンカー次元の場合、依存する親の最大数は 2 です。

Essbase では、計算用に表 175 に示す 3 つのオプションのいずれかが選択されます:

表 175　計算機キャッシュ・オプション

オプション	メソッド	パフォーマンス評価
1	単一のアンカー次元、複数ビットマップ	1
2	単一のアンカー次元、単一ビットマップ	2
3	複数のアンカー次元、単一ビットマップ	3

Essbase では、計算機キャッシュのサイズを基準にして、データベース計算に最適なパフォーマンスの計算が選択されます。前述のオプションのいずれかで計算機キャッシュのサイズが小さすぎる場合、Essbase によって計算機キャッシュは使用されません。計算のパフォーマンスは大幅に低下する可能性があります。

並列計算を使用可能にすると、使用される計算機キャッシュ・オプションが変わる場合があります。964 ページの「計算機のキャッシュ」を参照してください。

注意　初めてデータベースを計算する場合、計算パフォーマンスに特に重要なものは計算機キャッシュのサイズです。可能であれば、計算機キャッシュを、Essbase で最適な計算機キャッシュ・オプションを使用するために十分な大きさにしてください。

計算機キャッシュ・サイズの計算

計算機キャッシュの最適なサイズは、システムで使用可能なメモリー、およびデータベースの種類と構成によって異なります。

次の式を使用して、Essbase に必要な計算機キャッシュ・サイズを計算して、表 175 の 3 つのオプションのそれぞれを選択できます:

```
Calculator cache = Bitmap size in bytes * Number of bitmaps
```

ここで、

```
Bitmap size in bytes = Max ((member combinations on the bitmap dimensions / 8), 4)
```

また、ここで、

```
Number of bitmaps = Maximum number of dependent parents in the anchoring dimension
+ 2 constant bitmaps
```

注： ビットマップの最小サイズは 4 バイトです。(ビットマップ次元上のメンバーの組合せ/8)が 4 バイトより小さい場合は、Essbase では 4 バイトのビットマップ・サイズが使用されます。

表 176 で示されている、S1 から S5 までの疎次元を持つサンプル・データベースについて考えてみます:

表 176　例: 5 つの疎次元を使用したデータベースの事例

疎次元	メンバー数	依存する親
S1	20	適用なし
S2	20	適用なし
S3	50	適用なし
S4	50	適用なし
S5	200	3

次のサンプル計算では、このサンプルのデータベースを使用します:

- 923 ページの「オプション 1: 単一のアンカー次元、複数ビットマップ」
- 924 ページの「オプション 2: 単一のアンカー次元、単一ビットマップ」
- 925 ページの「オプション 3: 複数のアンカー次元、単一ビットマップ」

オプション 1: 単一のアンカー次元、複数ビットマップ

このサンプル計算では、データベースに関して次の状況を仮定します(表 175 より):

- ビットマップ次元: S1、S2、S3、S4
- アンカー次元: S5
- アンカー次元内の依存する親: 3

次の計算を実行します:

```
   Bitmap size in bytes = (S1 * S2 * S3 * S4) / 8
          = (20 * 20 * 50 * 50) / 8
          =
   125,000 bytes


Number of bitmaps = Maximum number of dependent parents in the anchoring dimension
          = +
          = 2 constant bitmaps
          = 3 + 2
          =
   5


Calculator cache = Bitmap size * Number of bitmaps
          = 125,000 * 5
          =
   625,000 bytes
```

Essbase では、1 つのアンカー次元を持つこのデータベースで複数のビットマップを使用するには、計算機キャッシュが 625,000 バイトである必要があります。

オプション 2: 単一のアンカー次元、単一ビットマップ

このサンプル計算では、データベースに関して次の状況を仮定します(表 175 より):

- ビットマップ次元: S1、S2、S3、S4
- アンカー次元: S5
- アンカー次元内の依存する親: 該当なし

次の計算を実行します:

```
   Bitmap size in bytes = (S1 * S2 * S3 * S4) / 8
          = (20 * 20 * 50 * 50) / 8
          =
   125,000 bytes


Number of bitmaps = Single bitmap
          =
   1


Calculator cache = Bitmap size * Number of bitmaps
          = 125,000 * 1
          =
   125,000 bytes
```

Essbase では、1 つのアンカー次元が存在するこのデータベースで単一のビットマップを使用するには、計算機キャッシュが 125,000 バイトである必要があります。

オプション 3: 複数のアンカー次元、単一ビットマップ

このサンプル計算では、データベースに関して次の状況を仮定します(表 175 より):

- ビットマップ次元: S1、S2、S3
- アンカー次元: S4、S5
- アンカー次元内の依存する親: 該当なし

次の計算を実行します:

次の計算を実行します:

```
Bitmap size in bytes = (S1 * S2 * S3) / 8
       = (20 * 20 * 50) / 8
       =
2,500 bytes


Number of bitmaps = Single bitmap
       =
1


Calculator cache = Bitmap size * Number of bitmaps
       = 2,500 * 1
       =
2,500 bytes
```

Essbase では、複数のアンカー次元が存在するこのデータベースで単一のビットマップを使用するには、計算機キャッシュが 2,500 バイトである必要があります。

データベースの計算機キャッシュ・サイズの選択

表 177 は、指定された計算機キャッシュ・サイズに基づいて Essbase が使用する計算機キャッシュ・オプションを示します:

表 177　計算機キャッシュ・サイズに基づく計算機キャッシュ・オプションの選択

指定した最小サイズ	選択されたオプション
625,000 バイト	オプション 1 (最適なパフォーマンス)
125,000 バイト	オプション 2
2,500 バイト [1]	オプション 3

[1]2,500 バイト未満の計算機キャッシュ・サイズを指定すると、Essbase では計算中に計算機キャッシュは使用されません。計算のパフォーマンスは大幅に低下する可能性があります。

計算スクリプトで SET MSG SUMMARY コマンドを使用することで、データベースに対して Essbase で使用できる計算機キャッシュ・オプションを確認できます。空のデータベースに対して次の計算スクリプトを実行します:

```
   SET MSG SUMMARY;
CALC ALL;
```

Essbase の ESSCMD ウィンドウまたはアプリケーション・ログに、計算機キャッシュ設定が表示されます。956 ページの「SET MSG SUMMARY と SET MSG DETAIL」を参照してください。

指定できる計算機キャッシュの最大サイズは、200,000,000 バイトです。デフォルトは 200,000 バイトです。選択する計算機キャッシュ・サイズは、使用可能なメモリーとデータベースの構成によって変わります。

注： 一般に、集約に基づくデータベース計算が多くなり、式計算に基づく計算が少なくなるほど、計算機キャッシュ、インデックス・キャッシュ、データ・ファイル・キャッシュおよびデータ・キャッシュのサイズが、パフォーマンスに与える影響は大きくなります。

初めてデータベースを計算する場合の、計算機キャッシュのサイズ設定

初めてデータベースを計算する場合、計算機キャッシュのサイズは特に重要です。可能であれば、計算機キャッシュを、Essbase で最適な計算機キャッシュ・オプションを使用するために十分な大きさにしてください。922 ページの「計算機キャッシュ・サイズの計算」を参照してください。

計算スクリプトによる計算機キャッシュの変更

デフォルトの計算機キャッシュのサイズを使用するか、計算スクリプト内で計算機キャッシュのサイズを設定できます。サイズを計算スクリプトから設定すると、その設定は計算スクリプトの実行中のみ使用されます。『Oracle Essbase テクニカル・リファレンス』で計算スクリプト SET CACHE コマンドおよび CALCCACHE 構成設定を参照してください。

動的計算キャッシュのサイズ設定

Essbase では、開いているデータベースごとに、別個の動的計算キャッシュが使用されます。`essbase.cfg` ファイル内の DYNCALCCACHEMAXSIZE 設定により、サーバー上の各動的計算キャッシュの最大サイズが指定されます。デフォルトでは、最大サイズは 20 MB です。DYNCALCACHEMAXSIZE 設定で指定された最大メモリー領域が割り当てられるまで、Essbase によってデータ・ブロックに対して動的計算キャッシュ内の領域が割り当てられます。927 ページの「動的計算キャッシュ・サイズの変更」を参照してください。

動的計算キャッシュの使用状況の確認

データベースごとに、Essbase によって各データ取得のアプリケーション・ログに 2 つのメッセージが書き込まれます:

```
   [Thu Oct 17 11:37:17 2007]Local/Sample///Info(1007125)
The number of Dynamic Calc Non-Store Members = [7 6 0 0 2 ]


   [Thu Oct 17 11:37:17 2007]Local/Sample///Info(1007126)
The number of Dynamic Calc Store Members = [0 0 0 0 0 ]
```

最初のメッセージは、取得に要した合計時間を示します。動的計算キャッシュを使用している場合、2 つ目のメッセージでは、データの計算機キャッシュ内で計算されたブロックの数(DCC = n)、および一般のメモリーで計算されたブロックの数(non-DCC = n)が表示されます。

動的計算キャッシュ・サイズの変更

動的計算キャッシュに関連する構成ファイル設定は、5 つあります。これらの動的計算キャッシュ設定の最適な値は、サーバー・マシン上のメモリー、サーバー・マシン上のすべてのデータベースの構成、およびユーザー・クエリーの指定によって異なります。

次の各構成設定の説明に、使用しているシステム用の値の決定方法に対する推奨事項を示します。サイト独自の要件に適合させるには、設定のテストや調整が必要な場合もあります。

- **DYNCALCCACHEMAXSIZE:** この設定では、サーバー上の動的計算キャッシュそれぞれに Essbase で割当てが可能な最大サイズを指定します。

  推奨する設定値= C * S * U

  - C は、`essbase.cfg` ファイル内の適切な CALCLOCKBLOCK 設定の値です(SET LOCKBLOCK コマンドにより、使用する CALCLOCKBLOCK 設定が指定されます)。
  - S は、マシン上のすべてのデータベース内の拡張ブロックの最大サイズです。拡張ブロックのサイズを計算するには、各密次元内のメンバー(動的計算メンバーと動的時系列メンバーを含む)の数を掛け合わせてブロック内のセルの数を求め、そのセルの数に各メンバー・セルのサイズ(8 バイト)を乗算します。

    例として、Sample.Basic 内の密次元内のメンバー・カウントについて考えます(ラベルのみのメンバーはカウントされません):

    - 19 (年、保管済メンバー 12、動的計算メンバー 7、HTD および QTD を含む)
    - 14 (メジャー、保管済メンバー 8、動的計算メンバー 6)
    - 4 (シナリオ、保管済メンバー 2、動的計算メンバー 2)

S = 19 * 14 * 4 セル(8 バイト/セル) =ブロック当たり 8512 バイト

この数値は、論理ブロック・サイズとしてアプリケーション・ログに出力されます。

- U は、最大の同時ユーザー数を抱えるデータベース上で予想される同時ユーザーの最大数です。

DYNCALCACHEMAXSIZE に値 0 (ゼロ)を割り当てることで、Essbase で動的計算キャッシュを使用しないよう指定します。

デフォルトでは、この値の最大サイズは 20MB (20,971,520 バイト)です。

- DYNCALCCACHEWAITFORBLK: 動的計算キャッシュに対して割り当てられた領域すべてが Essbase で使用されている場合は、この設定により、キャッシュ内のスペースが使用可能になるまで待機するか、動的計算キャッシュ以外のメモリーでただちにブロックを書き込んで計算するかが Essbase に指定されます。動的計算キャッシュが小さすぎる場合は、キューに複数のスレッドがあり、それぞれのスレッドがそのデータ・ブロックの計算を待機している可能性があります。

  推奨する設定値= FALSE (デフォルトの値)。

  TRUE に設定する前に、次の方法を試してください:

  - サーバー・コンピュータに物理メモリーを追加します。
  - DYNCALCCACHEMAXSIZE の値を増やしてテストし、これ以上動的計算キャッシュに使用できるメモリーがなくなるまで繰り返します。
- DYNCALCCACHEBLKTIMEOUT: 動的計算キャッシュ用の空き領域ができるまで Essbase が待機する場合の、待機時間の長さを設定します。

  推奨する設定値= WT / B

  - WT は、1 回のクエリーで待てる最長の時間で、たとえば 5 秒などにします。
  - B は、最大のクエリーでアクセスされる論理ブロックの合計数です。

    B の値を決定するには、アプリケーション・ログのメッセージを調べて、クエリーの Dyn.Calc.Cache の「Big Block Allocs」の最大数を確認します。例を 457 ページの「動的計算キャッシュの使用状況の確認」に示します。
- DYNCALCCACHEBLKRELEASE: Essbase で指定された待機時間が経過しても、動的計算キャッシュ内の領域が使用可能にならない場合、ただちに動的計算キャッシュの外にブロックを記述して計算するか、またはブロックをスワップ・アウトした後、動的計算キャッシュの圧縮ブロック・バッファで一時的に圧縮して、動的計算キャッシュ内に領域を作成するかを Essbase に指定します。

  推奨する設定値= FALSE (デフォルトの値)。

  メモリーの空き領域が極端に不足している場合にのみ、TRUE に設定してください。
- DYNCALCCACHECOMPRBLKBUFSIZE: Essbase で指定された待機時間が経過して、DYNCALCCACHEBLKRELEASE 設定が TRUE である場合の、動的計算キャッシュの圧縮ブロック・バッファのサイズを決定します。

推奨する設定値=(C * S) / 2

- C は、essbase.cfg ファイル内の現在の CALCLOCKBLOCK 設定の値です。SET LOCKBLOCK コマンドにより、現在の CALCLOCKBLOCK 構成設定が指定されます。
- S は、マシン上のすべてのデータベース内の拡張ブロックの最大サイズです。S は、DYNCALCCACHEMAXSIZE 設定の説明のとおりに計算します。

注： essbase.cfg ファイル内のパラメータを変更した後に、新しい値を使用するには、Essbase サーバーを停止して再起動する必要があります。

具体的な動的計算キャッシュの設定の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

キャッシュ設定の微調整

一般的なデータ、ユーザー・アクセス、および標準的環境(サーバーやネットワークなどを含む)のあるサイトでデータベースを使用した後で、Essbase のパフォーマンスを確認してください。テストなしで最適なキャッシュ・サイズを予測することは困難です。キャッシュ設定は調整が必要になる場合があります。

キャッシュ設定の理解

インデックス・キャッシュとデータ・ファイル・キャッシュ・サイズ(直接入出力を使用している場合)は、Essbase の最も重要なキャッシュ設定です。一般に、これらのキャッシュを大きくすれば、スワッピングのアクティビティの発生は減少します。ただし、キャッシュ・サイズを大きく設定すれば常にパフォーマンスが向上するとは限りません。キャッシュ・サイズの考慮事項を理解するには、この項全体をお読みください。

インデックス・キャッシュ

大きなインデックス・キャッシュの利点は、ある時点で頭打ちになります。インデックス・キャッシュのサイズがインデックスのサイズ(すべてのボリューム上のすべてのインデックス・ファイルを含む)と等しくなるかそれを超えると、パフォーマンスは向上しなくなります。ただし、将来のインデックスの増大を計上するため、インデックス・キャッシュのサイズを現在のインデックス・サイズより大きく設定できます。インデックス・キャッシュはインデックス・ページで埋められているため、ストレージの最適な使用のためには、インデックス・キャッシュのサイズをインデックス・ページのサイズ(8 KB)の倍数に設定します。インデックス・サイズの見積りの例については、1169 ページの「インデックス・ファイル」を参照してください。

データ・ファイル・キャッシュ

可能であれば、データ・ファイル・キャッシュは、保管されているデータのサイズ(すべての ess*.pag ファイルの合計サイズ)に等しいサイズに設定します。そうしない場合は、データ・ファイル・キャッシュを可能なかぎり大きくします。保管されているデータの将来の増大を計上する場合は、データ・ファイル・キャッシュ・サイズを保管されているデータの現在のサイズより大きく設定できます。

注： データ・ファイル・キャッシュは、直接入出力を利用している場合にのみ使用されます。

データ・キャッシュ

データ・キャッシュは、データ・ファイル・キャッシュの約 0.125 倍に設定します。ただし、計算によっては、より大きなデータ・キャッシュ・サイズが必要になる場合があります。多数の同時ユーザーが様々なデータ・ブロックにアクセスする場合は、このキャッシュをより大きくする必要があります。

データ・キャッシュとデータ・ファイル・キャッシュのどちらに対してメモリーを割り当てるのかを選択する必要がある場合で、直接入出力を使用しているときは、データ・ファイル・キャッシュを選択するのが一般的です。以前のバージョンの Essbase からアップグレードする場合は、Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

キャッシュのヒット率の確認

すべてのキャッシュに「ヒット率」があります: これは、要求された情報がキャッシュ内で使用可能である回数の割合です。インデックス・キャッシュ、データ・キャッシュおよびデータ・ファイル・キャッシュのヒット率を確認することで、キャッシュ・サイズを大きくすべきかどうかを決定できます。

➤ キャッシュのヒット率を確認するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「キャッシュのヒット率の確認」を参照してください。

- キャッシュのヒット率は、要求された情報がキャッシュ内にすでに存在したことのある回数の割合を示します。ヒット率が高いということは、データがキャッシュ内にある回数がより多いことを示します。この場合、要求されたデータを次のプロセスのためにディスクから取得する必要がないため、パフォーマンスが向上します。ヒット率 1.0 は、データが要求されるときは常にキャッシュ内に存在することを示します。これは、キャッシュ設定により最大のパフォーマンスです。
- 「インデックス・キャッシュ・ヒット率」には、Essbase カーネルが、ディスクから別のインデックス・ページを取得することなく、インデックス・キャッシュ内でインデックス情報を見つけられた成功率が示されます。
- 「データ・ファイル・キャッシュ・ヒット率」には、Essbase カーネルが、ディスクからデータ・ファイルを取得することなく、データ・ファイル・キャッシュ内でデータ・ファイル・ページを見つけられた成功率が示されます。

- 「データ・キャッシュ・ヒット率」には、Essbase が、データ・ファイル・キャッシュからデータ・ブロックを取得することなく、データ・キャッシュ内でデータ・ブロックを特定できた成功率が示されます。
- メモリーの割当てを確認します。必要な場合は、メモリーの量を一度に少しずつ追加してください。これは、少量ずつ増やすことで、大量のメモリーを増やしたときと同じ効果が得られることがあるためです。通常は、大きなメモリーを割り当てても、ヒット率はごくわずかしか向上しません。

パフォーマンスの確認

データベースのキャッシュ統計は、**performance statistics** 文法で **query database** MaxL ステートメントを使用することで確認できます。第 53 章「パフォーマンスの監視」を参照してください。

テスト計算の実行

計算は Essbase データベースでプロセッサに最も負荷がかかる操作であるため、テスト計算を実行して、様々なキャッシュ・サイズが Essbase サーバーのメモリー使用に与える影響を調べる必要があります。

56 データベースの再構築の最適化

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 91 ページの「パフォーマンスを最適化するためのアウトラインの設計」
- 128 ページの「次元およびメンバーの位置付け」
- 171 ページの「設計の代替アプローチの使用」
- 172 ページの「アウトラインのパフォーマンスの最適化」
- 179 ページの「計算と取得のパフォーマンスの最適化」

データベースの再構築

業務形態の変化にあわせて、Essbase のデータベース・アウトラインを変更し、新しい製品ラインの導入、新しいシナリオの情報の追加、新しい期間の反映などを行う必要があります。データベース・アウトラインの変更によって、データ・ストレージの配置が影響を受け、Essbase でのデータベースの再構築が必要になることもあります。

データベースの再構築を必要とする変更は非常に時間がかかるため(再構築の前にデータを破棄しないかぎり)、このような変更を決定するときは、それらがパフォーマンスにどのように影響するかに基づいて、十分に検討してください。この項では、再構築がパフォーマンスにどのような影響を与えるかを理解するために必要な情報を示し、データベースの再構築に関連して実行できるタスクについて説明します:

注： データの消去およびそれに伴う一部の再構築を回避する方法については、『Oracle Essbase テクニカル・リファレンス』の「CLEARDATA」および「CLEARBLOCK」、または Oracle Essbase Administration Services Online Help の「データの消去」を参照してください。

データベースの再構築のタイプ

サブトピック

- 暗黙的な再構築
- 明示的な再構築

この項では、データベースの再構築を起動するための 2 つの方法について説明します。

暗黙的な再構築

アウトライン・エディタまたは次元構築を使用してアウトラインを変更した後、Essbase でデータベース・ファイルの暗黙的な再構築が開始されます。実行される再構築のタイプは、アウトラインに加えた変更のタイプによって異なります:

- **密な再構築:** 密次元のメンバーが移動、削除または追加されると、Essbase でデータ・ファイル内のブロックが再構築され、新しいデータ・ファイルが作成されます。Essbase でデータ・ブロックが再構築されると、インデックス・エントリが新しいデータ・ブロックを指すように、自動的にインデックスが再生成されます。空のブロックは削除されません。再構築されたすべてのブロックは、Essbase でダーティとしてマークされるため、密な再構築の後はデータベースを再計算する必要があります。密な再構築は最も時間のかかる再構築であり、大きなデータベースに対して実行すると時間がかかる可能性があります。
- **疎な再構築:** 疎次元のメンバーが移動、削除または追加されると、Essbase でインデックスが再構築され、新しいインデックス・ファイルが作成されます。インデックスの再構築は比較的高速です。所要時間はインデックスのサイズによって変わります。
- **アウトラインのみの再構築:** 変更がデータベース・アウトラインのみに影響する場合、Essbase ではインデックス・ファイルやデータ・ファイルの再構築は行われません。データベース・アウトラインのみに影響する変更の例として、メンバー名の変更、別名の作成および動的計算式の変更があります。

増分再構築を使用する場合、Essbase では密な再構築は延期されます。データベース・アウトラインを頻繁に変更する場合は、増分再構築の使用可能化を検討してください。増分再構築の詳細は、938 ページの「増分再構築とパフォーマンス」を参照してください。

注: データベース・アウトラインを変更する方法(アウトライン・エディタの使用または次元構築の使用)は、再構築には影響を与えません。再構築が行われる場合、どのタイプの再構築が行われるかに影響するのは、情報の変更のタイプのみです。アウトラインの変更と行われる再構築のタイプの詳細は、941 ページの「アウトラインの変更のクイック・リファレンス」を参照してください。

明示的な再構築

データベースの再構築を手動で開始する場合は、明示的な再構築を実行します。明示的な再構築は、強制的にデータベースの完全な再構築を行います。完全な再構築は、密な再構築に加え、空のブロックの削除で構成されます。

➤ 全体の再構築を開始するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	手動によるデータベースの再構築	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』

データベースの再構築に影響する条件

高機能計算、名前変更および式の変更は、次のようにデータベースの再構築に影響を与えます:

- データベースで高機能計算を使用している場合には、データ・ブロックが再構築されるたびに、再構築されたブロックはすべてダーティとしてマークされます。ブロックをダーティとしてマークすると、次のデフォルトの高機能計算は必然的にフル計算となります。
- 名前または式を変更した場合、Essbase では影響を受けるブロックはダーティとしてマークされません。したがって、メンバーまたはデータベースを再計算するには、フル計算以外の方法を使用する必要があります。

再構築に関する情報については、表 178 を使用してください:

表 178　データベースの再構築に関連する項目

トピック	関連情報
高機能計算	441 ページの「データベースの再構築」
疎次元および密次元	• 62 ページの「疎次元および密次元」、63 ページの「密次元と疎次元の選択」 • 65 ページの「密次元と疎次元の選択のシナリオ」
属性次元	171 ページの「属性次元の設計」
次元構築	第 17 章「データ・ロードおよび次元構築の理解」
アウトライン・エディタ	第 7 章「データベース・アウトラインの作成および変更」

再構築時に使用される一時ファイル

Essbase でデータ・ブロックとインデックスの両方が再構築されるとき、表 179 に示すファイルが使用されます:

表 179　データベースの再構築時に使用されるファイル

ファイル	説明
ess xxxxx .pag	Essbase データ・ファイル
ess xxxxx .ind	Essbase インデックス・ファイル
dbname .esm	Essbase カーネル・ファイル(データベースのリカバリに使用される制御情報を含む)
dbname .tct	トランザクション制御表
dbname .ind	データおよびインデックスのフリー・フラグメント用のフリー・フラグメント・ファイル
dbname .otl	データベースのすべてのメタデータと、データの保管方法が定義されているアウトライン・ファイル

密な再構築

➤ 密な再構築を実行するとき、次のことが行われます:

1　次のファイルのコピーである一時ファイルが作成されます:

```
      ess
      xxxxx
      .ind
ess
      xxxxx
      .pag

      dbname
      .otl

      dbname
      .esm

      dbname
      .tct
```

それぞれの一時ファイルは、ファイル拡張子の最後の文字が n または u に置き換えられます。一時ファイルの名前は次のとおりです:

```
      ess
      xxxxx
      .inn
ess
      xxxxx
      .pan

      dbname
      .otn

      dbname
```

```
.esn

dbname
.tcu
```

2 手順 1 でコピーされたデータベース・ファイルからブロックが読み取られ、メモリー内でブロックが再構築され、新しい一時ファイルにブロックが保管されます。この手順に最も時間がかかります。

3 手順 1 でコピーしたデータベース・ファイルが削除されます

4 手順 1 で作成した一時ファイルの名前が、正しいファイル名に変更されます。

疎な再構築

Essbase で疎な再構築(インデックスのみの再構築)が行われるとき、次のファイルが使用されます:

- ess xxxxx .ind
- dbname .otl
- dbname .esm

➤ 疎な再構築を実行するとき、Essbase では次のことが行われます:

1 dbname.esm ファイル名が dbname.esr に変更されます。

2 essxxxxx.ind ファイル名が essxxxxx.inm に変更されます。

3 インデックス・ファイル(essxxxxx.ind)が作成されて、再構築操作によって変更されるインデックス情報が保管されます。

4 手順 1 で作成した dbname.esr および essxxxxx.inm が削除されます。

再構築操作の最適化

データベース・アウトラインを頻繁に変更する場合は、アウトラインとそれに加える変更のタイプを分析します。疎次元または属性次元に対する変更は、インデックスのみが変更されるため、比較的速く行われます。密次元に対する変更は、データ・ブロックが再構築されるため、比較的時間がかかります。

再構築操作のタイプを、速度の速い順に示します:

- アウトラインのみ(インデックス・ファイルまたはデータ・ファイルは含まれない)
- 疎(インデックス・ファイルのみ)
- 密(インデックス・ファイルおよびデータ・ファイル)。メンバーとその他の操作の追加、削除または移動によって発生するもの(941 ページの「アウトラインの変更のクイック・リファレンス」を参照)
- 密(インデックス・ファイルおよびデータ・ファイル)。密次元を疎次元に、または疎次元を密次元に変更することで発生するもの

パフォーマンスを向上するためのアクション

サブトピック

いくつかのアクションで、データベースの再構築に関連するパフォーマンスが改善されます:

- 頻繁に変更する次元がある場合は、その次元を疎にします。
- 必要なデータベースの再構築を Essbase でいつ実行するかを制御するには、増分再構築を使用します。
- マルチプロセッサ・システムでは並列再構築を検討します。
- 変更したアウトラインを保存するときには、必要な再構築の量を削減するオプションを選択します。

注： 分離レベルをコミット・アクセスに設定すると、データベースの再構築に必要なメモリーおよび時間が増加する可能性があります。データベースの再構築の前に、分離レベルをアンコミット・アクセスに設定することを検討してください。分離レベル設定の詳細は、第 51 章「データの整合性の確保」を参照してください。

増分再構築とパフォーマンス

データベース・アウトラインを頻繁に変更する場合は、増分再構築を使用可能にすることを検討してください。増分再構築を使用可能にすると、データベース・アウトラインまたは次元への変更によって構造的な変化が発生しないように、Essbase で再構築が延期されます。インデックスと、必要に応じて、次にブロックがアクセスされるときに影響を受けるブロックが Essbase で再構築されます。

増分再構築の理解

増分再構築が有効な場合は、特定のアウトライン操作のために後続のアクセスが必要になるまで、インデックス・ファイル(essxxxxx.ind)およびデータ・ファイル(essxxxxx.pag)への変更が延期されます。データ量の多いデータベースでは、この延期によってかなりの時間を節約できます。Essbase が増分再構築を適用できるアウトライン操作のリストは、941 ページの「アウトラインの変更のクイック・リファレンス」を参照してください。

次の変更は、増分再構築の設定に優先します。つまり、増分再構築が使用可能であっても、これらの変更では再構築が即時に行われます:

- 属性以外の次元の追加または削除。
- 疎次元での保管されているメンバーの削除。

- 疎から密、あるいは密から疎への次元定義の変更。
- データベース内で LRO を使用している場合、そのデータベースでは増分再構築が自動的に使用不可になります。増分再構築の使用不可は、サーバー上の他のデータベースには影響を与えません。
- また、疎次元へのメンバーの追加および疎次元の変更でも、場合によっては即時の再構築が発生することがあります。941 ページの「アウトラインの変更のクイック・リファレンス」を参照してください。

増分再構築が使用可能かどうかにかかわらず、アウトラインがすでに再構築されている(密な再構築が保留中である)場合は、共有メンバーを追加すると Essbase で密な再構築が実行されます。

注： 再構築操作を行った後は、必ずデータベースの再計算を実行してください。

増分再構築の使用

次のデータベースに対して、増分再構築を使用可能にできます:

- アプリケーション内の 1 つのデータベース
- アプリケーション内のすべてのデータベース
- すべてのアプリケーション内のすべてのデータベース

増分再構築を使用可能にするには、`essbase.cfg` ファイル内の INCRESTRUC 構成設定を使用します。『Oracle Essbase テクニカル・リファレンス』を参照してください。

Essbase では、アウトラインの変更は、内部ファイル、`dbname.ocl` に記録されます。Essbase によってこのファイルに対し、密な再構築が行われたとき、またはデータベースが消去またはリセットされたときに、消去されます。ファイル `dbname.ocl` はきわめて大きくなる可能性があります。このファイルを消去するには、ESSCMD で VALIDATE を発行します。これにより、再構築が延期されたブロックが Essbase で再構築されます。VALIDATE を発行するときは、データベースが読取り専用モード(データベースのバックアップに使用される)ではないことを確認します。872 ページの「VALIDATE を使用した整合性の検査」を参照してください。

並列再構築

デフォルトでは、ブロック・ストレージの再構築は連続して実行されます。ブロックが最初から最後まで再度番号付けされ、再形成されるため、処理に時間がかかります。並列再構築では、使用可能なプロセッサ・コアを使用するために、ブロックの再構築作業を複数の同時スレッド全体に分割することで、再構築の時間が短縮されます。計算は再構築とは別に実行されるため、各ブロックは他のブロックから独立して再構築されます。

ブロックは n のグループに分割されます。n は、再構築スレッドの数です。この分割は、レベルのキーの数が n 以上になるまで、インデックス BTree の大きさを

走査することにより実行されます。キーの数が n を超えるレベルがない場合には、再構築スレッドの数もそれに応じて減少します。

使用する再構築スレッドの数は、RESTRUCTURETHREADS 構成設定を使用して essbase.cfg に定義されます。RESTRUCTURETHREADS が定義されていない場合、デフォルトは 1 スレッドです。詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

変更したアウトラインを保存する場合のオプション

データベースの再構築をトリガーする(アウトライン・エディタを使用して)アウトラインの変更を保管するとき、Essbase でダイアログ・ボックスが表示されます。「データベースの再構築」ダイアログ・ボックスでは、再構築中にデータ値がどのように処理されるかを定義します。たとえば、再構築中にすべてのデータを保持、レベル 0 または入力データのみを保持、またはすべてのデータを破棄できます。Oracle Essbase Administration Services Online Help の「アウトラインの保管」を参照してください。

データベースにデータが含まれている場合は、サーバー上にデータベースのバックアップ・コピーを作成するために十分な空きディスク・スペースが必要です。バックアップにより、再構築プロセス中の異常終了によってデータベースが破壊されることがないようにします。

Essbase で再構築不要のメッセージが表示されることがありますが、その場合でもインデックスのみの再構築は実行されます。これは、疎次元に変更を加えた場合に発生する可能性が高くなります。再構築操作の取消しを試みた場合、Essbase で取消し不可のメッセージが表示されることがあります。このようなメッセージが表示された場合は、Essbase ですでに最終的なクリーンアップが実行中のため取り消せません。

アウトライン変更ログ

アウトライン変更ログをアクティブにすると、アウトラインに影響するすべてのアクティビティ(メンバー名の変更やメンバーの移動など)が Essbase に記録されます。アウトラインに加える変更が多くなるほど、Essbase でログに加える必要がある更新が多くなるため、パフォーマンスは低下します。

デフォルトでは、アウトライン変更は Essbase のログに記録されません。アウトラインのロギングによりパフォーマンスが低下しているかどうかを確認するには、essbase.cfg ファイルの OUTLINECHANGELOG が TRUE になっているかどうかを調べてください。822 ページの「アウトライン変更ログとその使用」を参照してください。

Essbase パーティショニング・オプション

パーティショニングを使用すると、Essbase でアウトラインの変更が追跡されて、パーティション間でデータベース・アウトラインを同期できるようになります。

アウトラインの変更を追跡すると、構造的な変更が多い場合は特に、再構築の速度が低下します。

パーティショニングを使用しているときに Essbase でデータの再構築を行う場合、次の手順を実行して、パーティション間でデータを確実に同期化させます:

1. パーティションを検証します。

   261 ページの「パーティションの検証」を参照してください。

   注： パーティションを検証するには、データベース・マネージャ以上の権限が必要です。

2. パーティションのアウトラインを同期化します。

   263 ページの「アウトラインの同期化」を参照してください。

アウトラインの変更のクイック・リファレンス

この項の表に、計算および再構築(増分再構築を含む)に影響を与えるアウトラインの変更をすべて示します。

注： パーティショニングを使用している場合、再構築は接続先のデータベースにのみ影響を与えます。

表 180　アクション: メンバーの削除、追加または移動

アクション	計算および標準的な再構築への影響	増分再構築が適用されるか? (使用可能な場合)
疎次元のメンバーの削除	関係の変更を反映するために、データを再計算する必要があります。 Essbase では、削除したメンバーによって表されるブロックに対するすべてのポインタは、インデックス・ファイルから削除されます。ブロックはこれ以上参照されなくなるため、空きスペースになります。再構築は行われません。	通常のメンバーには適用されません。増分再構築を優先して、Essbase でインデックスが再構築されます。 ラベルのみメンバーの場合、増分再構築が適用され、再構築は延期されます。
属性次元のメンバーの削除	なし	いいえ
密次元のメンバーの削除	関係の変更を反映するために、データを再計算する必要があります。 データ・ファイルは、変更されたブロック・サイズを反映するよう Essbase で再構築されます。Essbase でインデックスが再構築されます。	はい。データおよびインデックス・ファイルへの変更は延期されます。
疎次元または密次元における共有メンバーの削除	データを再計算する必要があります。データは、依然として元のメンバー名に関連付けられています。しかし、共有メンバーの親が子のデータに依存する場合があるので、再計算が必要です。 再構築は行われません。	いいえ

アクション	計算および標準的な再構築への影響	増分再構築が適用されるか?(使用可能な場合)
疎次元へのメンバーの追加	新しいメンバーのデータをロードまたは計算し、新しい値を取得する必要があります。 Essbase でインデックスが再構築されます。	はい。データおよびインデックス・ファイルへの変更は延期されます。
密次元へのメンバーの追加	新しいメンバーのデータをロードまたは計算し、新しい値を取得する必要があります。データを再計算する必要があります。 データ・ファイルは、変更されたブロック・サイズを反映するよう Essbase で再構築されます。Essbase でインデックスが再構築されます。	はい。データおよびインデックス・ファイルへの変更は延期されます。
属性次元へのメンバーの追加	なし	いいえ
疎次元または密次元に対する共有メンバーの追加	データを再計算する必要があります。新しい共有メンバーは、その親への集計に影響を及ぼします。 再構築は行われません。	いいえ
疎次元内の通常のメンバーの移動	集計の変更を反映するために、データを再計算する必要があります。 Essbase でインデックス・ファイルが再構築されます。	いいえ。インデックス・ファイルは、増分再構築を優先して Essbase で再構築されます。
密次元内の通常のメンバーの移動	集計の変更を反映するために、データを再計算する必要があります。 Essbase でインデックス・ファイルとデータ・ファイルが再構築されます。	はい。データおよびインデックス・ファイルへの変更は延期されます。
属性次元メンバーの移動	なし	いいえ

表 181 アクション: その他のメンバー関連の変更

アクション	計算および標準的な再構築への影響	増分再構築が適用されるか?(使用可能な場合)
メンバーの別名の変更またはメンバーへの別名の追加	なし	いいえ
メンバー名の変更	なし	いいえ
メンバー式の変更	式の変更を反映するために、データを再計算する必要があります。 再構築は行われません。	いいえ

表 182 アクション: 動的計算関連の変更

アクション	計算および標準的な再構築への影響	増分再構築が適用されるか?(使用可能な場合)
動的計算および保管としての動的計算メンバーの定義	密次元メンバーの場合: Essbase で、インデックス・ファイルとデータ・ファイルが両方とも再構築されます。 疎次元メンバーの場合: 再構築は行われません。	はい。データおよびインデックス・ファイルへの変更は延期されます。
動的計算としての動的計算および保管メンバーの定義	なし	いいえ
動的計算および保管としての通常の密次元メンバーの定義	なし	いいえ
動的計算としての通常の密次元メンバーの定義	Essbase で、インデックス・ファイルとデータ・ファイルが両方とも再構築されます。	再構築は延期されます。
通常のメンバーとしての疎次元の動的計算および保管メンバーまたは動的計算メンバーの定義	再構築は行われません	いいえ
動的計算または動的計算および保管としての疎次元の通常メンバーの定義	Essbase でインデックス・ファイルとデータ・ファイルが再構築されます。	はい。データおよびインデックス・ファイルへの変更は延期されます。
通常のメンバーとしての密次元の動的計算および保管メンバーの定義	再構築は行われません	いいえ
通常のメンバーとしての密次元の動的再計算メンバーの定義	Essbase でインデックス・ファイルとデータ・ファイルが再構築されます。	はい。データおよびインデックス・ファイルへの変更は延期されます。
動的計算メンバーとしての密次元の通常メンバーの定義	Essbase でインデックス・ファイルとデータ・ファイルが再構築されます。	はい。データおよびインデックス・ファイルへの変更は延期されます。
疎次元の動的計算メンバーの追加、削除、または移動	Essbase でインデックス・ファイルが再構築されます。	メンバーの追加または削除の場合、データおよびインデックス・ファイルへの変更は延期されます。 メンバーの移動の場合、増分再構築を優先して、Essbase でインデックス・ファイルが再構築されます。
疎次元の動的計算および保管メンバーの追加、削除、または移動	Essbase でインデックス・ファイルが再構築されます。	メンバーの追加の場合、データおよびインデックス・ファイルへの変更は延期されます。 メンバーの移動または削除の場合、Essbase でインデックス・ファイルが再構築されます(増分再構築が優先されます)。
密次元の動的計算および保管メンバーの追加、削除、または移動	Essbase でインデックス・ファイルとデータ・ファイルが再構築されます。	いいえ

アクション	計算および標準的な再構築への影響	増分再構築が適用されるか?(使用可能な場合)
密次元の動的計算メンバーの追加、削除、または移動	再構築は行われません。	いいえ

表 183 アクション: プロパティとその他の変更

アクション	計算および標準的な再構築への影響	増分再構築が適用されるか?(使用可能な場合)
密-疎のプロパティの変更	データを再計算する必要があります。 Essbase で、インデックス・ファイルとデータ・ファイルが両方とも再構築されます。	増分再構築を優先して、Essbase でインデックス・ファイルとデータ・ファイルが再構築されます。
ラベルのみプロパティの変更	データを再計算する必要があります。 Essbase でインデックス・ファイルとデータ・ファイルが再構築されます。	はい。データおよびインデックス・ファイルへの変更は延期されます。
共有メンバーのプロパティの変更	データを再計算して、変更した子のデータ値を反映させる必要があります。 Essbase で、インデックス・ファイルとデータ・ファイルが両方とも再構築されます。	はい。データおよびインデックス・ファイルへの変更は延期されます。
プロパティ(密-疎、ラベルまたは共有を除いた他のプロパティ)の変更	タイム・バランスの期首から期末への変更などの集計プロパティの変更を反映するため、データを再計算することが必要な場合があります。	いいえ
2 つの疎次元の順序の変更	計算またはデータ・ロードには影響がありません。 Essbase でインデックスが再構築されます。	インデックスは、増分再構築を優先して Essbase で再構築されます。
次元の順序の変更	データを再計算する必要があります。 Essbase で、インデックス・ファイルとデータ・ファイルが両方とも再構築されます。	Essbase でインデックス・ファイルとデータ・ファイルが再構築されます(増分再構築が優先されます)。
属性次元の順序の変更	なし	いいえ
別名テーブルの作成、削除、消去、名前変更、またはコピー	なし	いいえ
別名テーブルのインポートまたはメンバーの別名の設定	なし	いいえ
大文字と小文字の区別の設定の変更	なし	いいえ
レベルおよび世代の名前付け	なし	いいえ
UDA の作成、変更、または削除	なし	いいえ

57 データ・ロードの最適化

この章の内容

この章の情報の一部は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

データ・ロードの理解

この項は、集約ストレージ・データベースには適用されません。

大きなデータ・ソースを Essbase データベースにロードするには、数時間かかる場合があります。データ・ロードの処理時間は、次のようなアクションにかかる時間を最小限に抑えることによって短縮できます:

- データ・ソースの読取りと解析
- データベースへの読取りと書込み

データ・ロードのパフォーマンスを最適化するには、データベース構造の点から考えます。Essbase では、データはブロックごとにロードされます。疎次元メンバーの一意の組合せごとに、1 つのデータ・ブロックには密次元のすべての組合せのデータが含まれています。これは、データが含まれているセルが少なくとも 1 つあることを前提としています。ブロックのある場所により速くアクセスするため、Essbase ではインデックスが使用されます。インデックス内の各エントリは、1 つのデータ・ブロックに対応しています。62 ページの「疎次元および密次元」、63 ページの「密次元と疎次元の選択」、および 65 ページの「密次元と疎次元の選択のシナリオ」を参照してください。

Essbase でデータ・ソースがロードされるとき、Essbase ではデータが次の 3 つの主な段階で処理されます:

- **入力:** Essbase でデータ・ソースの一部が読み取られます。

- **準備:** データをブロックに振り分けるための準備として、Essbase でデータが整理されます。
- **書込み:** Essbase により、データがメモリーのブロックに振り分けられ、次にブロックがディスクに書き込まれます。このとき、ディスク上の正しいブロックは、疎の交差に基づくポインタで構成されるインデックスを使用して検索されます。

すべてのデータがロードされるまで、このプロセスが繰り返されます。各段階で 1 つ以上の処理スレッドを使用することで、Essbase では複数のプロセスを並列処理できます。950 ページの「並列データ・ロードの使用」を参照してください。

この章の例は、274 ページの「データ・ソース」に記載されているコンテンツが理解されていることを前提としています。

疎メンバーの組合せのグループ化

この項は、集約ストレージ・データベースには適用されません。

パフォーマンスを向上する最も効果的な戦略は、データベースへの書込みや読取りの間に Essbase で実行する必要のあるディスク入出力の数を最小限に抑えることです。Essbase ではデータがブロックごとにロードされるため、ソース・データを物理的なブロック編成に対応するように編成すると、Essbase で実行する必要のある物理的なディスク入出力の数が減ります。

データ・ソースを整理して、疎次元の同じ一意の組合せを持つレコードをグループ化します。この配列はデータベース内のブロックに対応します。

この章の例では、この戦略に従ってデータを編成する方法を示します。これらの例では、表 184 で示す、Sample.Basic データベースのサブセットを使用しています:

表 184　例の次元と値

疎、非属性次元	密次元
シナリオ(予算、実績)	メジャー(売上高、Margin、COG、利益)
製品(コーラ、ルート・ビア)	年(1 月、2 月)
市場(フロリダ、オハイオ)	

注： 属性次元にはデータはロードされないので、属性次元は疎次元ですが、この項での説明とは無関係です。

次のようなデータ・ソースについて考えてみます。疎次元メンバーの組合せによってグループ化されていないため、このデータは最適化のためのソートは行われていません。Essbase での各レコードの読取りのたびに、疎次元の異なるメンバーの処理が必要になります。

```
   Jan
Actual   Cola     Ohio    Sales   25
```

```
Budget   "Root Beer"  Florida  Sales   28
Actual   "Root Beer"  Ohio     Sales   18
Budget   Cola        Florida  Sales   30
```

Essbase では、1 つではなく 4 つのブロックにアクセスするため、このデータ・ロードには時間がかかります。

同じ Sample.Basic データベースの最適に編成されたデータ・ソースでは、疎次元メンバーの一意の組合せによって「Actual」->「Cola」->「Ohio」の順にソートされた異なるレコードが示されます。Essbase で 1 つのブロックにアクセスするのみで、これらのレコードがロードされます。

```
   Actual   Cola   Ohio   Jan  Sales   25
Actual   Cola   Ohio   Jan  Margin  18
Actual   Cola   Ohio   Jan  COGS    20
Actual   Cola   Ohio   Jan  Profit   5
```

1 つのレコードにつき多数のセルをロードするデータ・ソースを使用できます。レコードが、一意の疎次元メンバーの組合せでグループ化されていることを確認します。次に、複数の値を指定するレコードの次元が密次元であるようにレコードを並べます。

次のデータ・ソースの例では、ヘッダー・レコードを使用して、密次元であるメジャー次元のメンバーを特定しています。データは、最初に密次元である年のメンバーごとにソートされた後、他の次元のメンバーごとに階層的にグループ化されています。各レコードには、メジャー次元の複数の値が示されています。

```
                  Sales  Margin  COG  Profit
Jan Actual  Cola     Ohio    25    18   20    5
Jan Actual  Cola     Florida  30    19   20    10
Jan Actual  "Root Beer"  Ohio    18    12   10    8
Jan Actual  "Root Beer"  Florida  28    18   20    8
```

この例では、必要な行が見出しと最初のデータ行の 2 行であることに注目してください。前の例では、同じデータに対して 4 行が必要です。

ロードの前にソース・ファイル内のデータを整理する方法の詳細については、281 ページの「ルール・ファイルを必要としないデータ・ソース」を参照してください。

データ・ソースの最小化

データ・ソースを可能なかぎり小さくします。データ・ソース内の、Essbase で読み取るフィールドが少ないほど、データの読取りとロードに必要な時間が少なくてすみます。

データを範囲別にグループ化します。データ・ソース内の冗長性をなくすと、データ値のロード前に Essbase で読み取る必要のあるフィールドの数が減ります。

次の例のデータ・ソースは、範囲別に編成されていません。これには、フィールドの不要な繰返しが含まれています。値はすべて「Profit」の値です。「Profit」

は、該当するデータのグループの最初のみに含めるだけですみます。この例には、データ値を適切にロードするために Essbase で読み取る必要のある、33 個のフィールドが含まれています。

```
   Profit
Jan   "New York"  Cola      4
Jan   "New York"  "Diet Cola"  3
Jan   Ohio      Cola      8
Jan   Ohio      "Diet Cola"  7
Feb   "New York"  Cola      6
Feb   "New York"  "Diet Cola"  8
Feb   Ohio      Cola      7
Feb   Ohio      "Diet Cola"  9
```

次の例では、同じデータが、メンバーを範囲別にグループ化することによって最適化されています。冗長性をなくすことによって、この例では、データ値を適切にロードするために Essbase で読み取る必要のあるフィールドが、23 個のみになっています。

```
   Profit
Jan  "New York"  Cola      4
         "Diet Cola"  3
   Ohio      Cola      8
         "Diet Cola"  7
Feb  "New York"  Cola      6
         "Diet Cola"  8
   Ohio      Cola      7
         "Diet Cola"  9
```

Essbase では、最初の値 4 が「Jan」->「New York」->「Cola」に、次の値 3 が「Jan」->「New York」->「Diet Cola」にといった具合で割り当てられます。

効率的にソートされてはいますが、密次元ごとにソートされてグループ化されたデータ・ソースは、ロード・プロセスの速度を遅くする可能性のある多数の繰返しを示しています。このデータは、データを範囲別にグループ化することで、さらに最適化できます。下のような最適化されたデータ・ソースでは、冗長なフィールドがなくなり、処理時間が削減されます。

```
                 Sales  Margin  COG  Profit
Jan Actual  Cola     Ohio    25   18   20    5
            Florida  30    19   20    10
      "Root Beer"  Ohio    18    12   10    8
            Florida  28    18   20    8
```

282 ページの「メンバー・フィールド範囲のフォーマット」を参照してください。

ソース・フィールドの最小化

データ・ソース内のフィールドを小さくすることで、Essbase での読取りとロードをより高速にできます。

データ・ソース内のフィールドを可能なかぎり小さくするには、次のタスクを実行します:

- データ・ソース内の余分な空白を削除します。たとえば、空白のかわりにタブを使用します。
- コンピュータが生成した数値を、必要な精度に丸めます。たとえば、データ値の小数点以下の桁数が 9 桁で、関心があるのは 2 桁の場合は、その数値を丸めて 2 桁にします。
- #MISSING のかわりに#MI を使用します。

アウトラインと同じ順序でのデータの配置

この項は、集約ストレージ・データベースには適用されません。

インデックスは、アウトライン内の疎次元と同じ順序で編成されています。データ・ソース内の疎のデータの組合せをグループ化して、データ・ソースをさらに最適化するには、疎次元がアウトラインと同じ順序になるようにデータを整理します。

Essbase では、データ・ロードまたはその他の操作による要求に応じて、インデックスの一部がメモリーにページ・インされたりページ・アウトされたりします。ソース・データを整理して、インデックス内のエントリの順序を一致させれば、インデックスのページングが少なくてすむため、データ・ロードの速度が上がります。ページングが少なくなれば、結果として入出力操作が少なくなります。

Essbase では、インデックス・キャッシュ・サイズを使用して、メモリーにページ・インできるインデックスの量が決定されます。インデックス・キャッシュのサイズを調整することで、データ・ロードのパフォーマンスが向上する場合もあります。

注： インデックス・キャッシュのサイズが、メモリー内のすべてのインデックスを含めるほど大きい場合は、データをアウトラインと同じ順序に配列してもデータ・ロードの速度には影響しません。

917 ページの「インデックス・キャッシュ・サイズの設定」を参照してください。

Essbase サーバーからのロード

Essbase サーバーからのデータ・ソースのロードは、クライアント・コンピュータからのロードよりも高速です。サーバーからデータ・ソースをロードするには、サーバーにデータ・ソースを移動してから、ロードを開始します。

サーバーからデータをロードすると、ネットワーク上で、クライアント・コンピュータからサーバー・コンピュータにデータを転送する必要がないため、パフォーマンスが向上します。

並列データ・ロードの使用

サブトピック

- 並列データ・ロードの理解
- 並列データ・ロードの有効化
- 並列データ・ロードの最適化

次のトピックでは、並列データ・ロードと、それによってサイトのパフォーマンスを向上させる方法について説明します。

並列データ・ロードの理解

並列データ・ロードは、複数のデータ・ファイルを Essbase データベースに同時にロードすることを意味します。大規模なデータ・セット(2GB のファイル 10 個など)を処理する場合、データ・ソースを同時にロードすることで、複数のプロセッサと高パフォーマンスのストレージ・サブシステムを備えた最新サーバーの CPU リソースや I/O チャネルを十分に活用できます。

並列データ・ロードは、`essbase.cfg` の設定である DLTHREADSPREPARE および DLTHREADSWRITE を使用して行われる単一ファイルのデータ・ロード・パイプライン最適化とは異なります。最適化された単一パイプライン・データ・ロードは、各ステージが 1 つ以上のスレッドを使用して同時に実行されるという意味において並列であり、一度にロードされるデータ・ファイルは 1 つのみです。

並列データ・ロードでは、サーバー側の複数の並列パイプラインとクライアント側の複数のスレッドを使用して、複数のデータ・ファイルが同時にロードされるため、データ・ロードが最新サーバーの性能に対して真に最適化されます。

並列データ・ロードの有効化

➤ 並列データ・ロードを有効化するには:

1. **使用するすべてのデータ・ソース・ファイルが一致するよう、ワイルドカード文字(*および/または?)を使用して、複数のファイルをデータ・ソースとして指定します。『Oracle Essbase テクニカル・リファレンス』で、MaxL import data ステートメントを参照してください。**
2. **必要な場合は、並列データ・ロードによって生成されるスレッド数を制御します。**

   950 ページの「並列データ・ロードの最適化」を参照してください。

並列データ・ロードの最適化

並列データ・ロードには、使用されるクライアント・スレッドまたはサーバー・パイプラインの数を制限するスロットルがあります。データ・ファイルの指定が数百ファイルと一致する場合は、その数のスレッドまたはパイプラインが生成されないようにするために、スロットル制御が重要です。データ・ロード要求によって生成されるスレッドまたはパイプラインの数を制御するには、**import data** MaxL ステートメントで **max_threads** 文法を使用して制限を設定します

一般的に、CPU コアを超えるパイプラインは使用しません。また、システム I/O 帯域幅の使用状況を監視し、I/O サブシステムが並列で処理できるデータ・ロード数を判断します。951 ページの「プロセッサおよび I/O アクティビティの監視」を参照してください

並列データ・ロードを大量に実行している場合は、essbase.cfg ファイルで DLSINGLETHREADPERSTAGE を TRUE に設定することで、パフォーマンスが向上する場合があります。

プロセッサおよび I/O アクティビティの監視

データをロードする際には、プロセッサおよび I/O アクティビティを表示できます。オペレーティング・システムごとに、システム・パフォーマンスを表示するための様々なツールがあります。たとえば、Windows のタスク マネージャでは、プロセッサやメモリーの使用状況とプロセスを表示できます。

詳細は次のとおりです:

- Windows XP の場合は、「コントロール パネル」、「管理ツール」、「パフォーマンス」の順に選択します
- Windows Server 2008 の場合は、「コントロール パネル」、「管理ツール」、「信頼性とパフォーマンス モニタ」の順に選択します
- Windows 7 の場合は、「コントロール パネル」、「管理ツール」、「パフォーマンス モニタ」、「リソース モニター」の順に選択します

UNIX で使用可能なツールは、top および vmstat です; iostat は、I/O アクティビティの監視に最も一般的に使用されるユーティリティです。また、サードパーティ製のツールを使用して、システムの使用状況を表示したり分析したりすることもできます。

データ・キャッシュ・サイズ設定への影響

ブロック・ストレージ・データベースの場合、Essbase サーバーにより、圧縮されていないデータ・ブロックを保持するためにデータ・キャッシュ・メモリーが割り当てられます。DLTHREADSWRITE 構成設定を使用している場合、その設定で指定された各スレッドは、拡張ブロックのサイズに等しい、データ・キャッシュの領域を使用します。

ブロックのサイズ、スレッドの数、およびデータ・ロード中に他の同時操作で使用されるデータ・キャッシュの大きさによっては、使用できる量よりも多くのデータ・キャッシュが必要になる可能性があります。このような場合は、スレッドの数を減らすか、データ・キャッシュ・サイズを大きくします。

920 ページの「データ・キャッシュ・サイズの変更」を参照してください。

58 計算の最適化

この章の内容

この章の情報は、ブロック・ストレージ・データベースのみに適用され、集約ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 第 15 章「パーティション・アプリケーションの設計」
- 第 27 章「データ値の動的計算」
- 第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」
- 第 26 章「高機能計算についての理解」

計算パフォーマンスの設計

データベースの構成の仕方によって、計算パフォーマンスを最適化できます。

サイトの最適な構成は、データベースの性質とサイズによって異なります。次の項の情報は、単にガイドラインとして使用してください。

ブロック・サイズとブロック密度

8 KB - 100 KB のデータ・ブロック・サイズならば、大抵の場合は最適なパフォーマンスが得られます。

データ・ブロックが 8 KB よりも大幅に少ない場合、通常インデックスは非常に大きくなり、インデックスのディスクへの書込みやディスクからの取得が Essbase で強制的に実行されます。このプロセスにより、計算の速度が遅くなります。

データ・ブロックが 100 KB よりも大幅に大きい場合は、高機能計算が効率よく機能しません。第 26 章「高機能計算についての理解」を参照してください。

計算パフォーマンスとデータ・ストレージを最適化するには、データベースの密次元と疎次元の構成を調整することによって、データ・ブロック密度とデータ・ブロック・サイズのバランスをとります。次の点に留意してください:

- データ・ブロックのサイズは 8 KB から 100 KB にして、ブロック密度は可能なかぎり高くしてください。
- 代表的なデータが含まれているデータベースの、最も期待できる構成のテスト計算を実行してください。結果を確認して、最高の計算パフォーマンスを得られる構成を決定してください。

データ・ブロックとデータ・ブロック・サイズの潜在的な数や実際の数など、データベースに関する情報を表示できます。

➤ データ・ブロック情報を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データ・ブロック統計の確認	Oracle Essbase Administration Services Online Help
ESSCMD	GETDBINFO	『Oracle Essbase テクニカル・リファレンス』

疎次元の順序

データベース・アウトライン内の標準の(属性以外の)疎次元の順序を変更することで、計算パフォーマンスが向上する場合があります。含まれているメンバーが最も少ない次元を最初に配置して、含まれているメンバーの数によって標準の疎次元を順序付けます。この配列により、サイトによっては多くの点で改善される可能性があります:

- 大規模な次元(1000 以上のメンバーを含む次元など)を持つデータベース・アウトラインの場合は、計算機キャッシュの効率性が上がり、パフォーマンスが約 10%改善されます。
- 最もメンバー数の多い疎の標準次元が、アウトライン内の最後の次元である場合は、並列計算が使用される可能性が高くなります(並列計算が使用可能な場合)。

増分データ・ロード

多くの企業では、データを増分的にロードします。たとえば、月ごとにその月分のデータをロードする場合があります。

データを増分的にロードするときの計算パフォーマンスを最適化するには、時間としてタグ付けされた次元を疎次元にします。時間次元が疎の場合、データベースにはそれぞれの期間のデータ・ブロックが含まれます。データを期間ごとにロードする場合は、該当の期間を含むブロックが少ないため、Essbase でアクセスされるデータ・ブロックが少なくなります。したがって、高機能計算が使用可能になっている場合は、ダーティとしてマークされたブロックのみが再計算されます。たとえば、3 月のデータをロードする場合は、3 月のデータ・ブロックと 3 月に従属する親のみが更新されます。

ただし、本来密である時間次元を疎にすると、インデックスのサイズが大幅に拡大し、その大きなインデックスに対応して物理的出入力アクティビティの量が増えるために、パフォーマンスの速度が低下する可能性があります。

時間としてタグ付けされた次元が密の場合でも、疎次元に対する部分的なデータ・ロードを行うときは、高機能計算の利点の一部を利用できます。たとえば、製品が疎で、1 つの製品のデータをロードする場合は、時間が密で高機能計算が使用可能になっている場合でも、Essbase では部分的なロードによって影響を受けるブロックのみが再計算されます。

増分ロードの詳細は、1057 ページの「集約ストレージ・データベースへのデータのロード」を参照してください。

複数のフラット次元を含むデータベース・アウトライン

データベース・アウトラインに複数のフラット次元が含まれている場合は、計算パフォーマンスが影響を受けることがあります。フラット次元の親の数は非常に少なく、それぞれの親に何千もの子があります。言い換えれば、フラット次元には多数のメンバーがあり、レベルは少ないことになります。

データベース・アウトラインに中間レベルを追加することで、複数のフラット次元を含むアウトラインに対するパフォーマンスを改善できます。

式と計算スクリプト

計算スクリプトの式と次元を慎重にグループ化することで、計算パフォーマンスを大幅に向上できる場合があります。この方法で、計算中にデータベース内のデータ・ブロックが Essbase によって循環される回数を、可能なかぎり少なくできます。

計算スクリプトのコマンドを、データベース計算が可能なかぎり簡単になるように順序付けます。すべての式をデータベース・アウトラインに適用することと、デフォルト計算(CALC ALL)を使用することを検討してください。この方法で、計算パフォーマンスを向上できる場合があります。

第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」および 423 ページの「計算パス」を参照してください。

計算の監視と追跡

計算スクリプトで次の各コマンドを使用すれば、Essbase でのデータベースの計算に関するアプリケーション・ログの情報を表示できます。

SET MSG SUMMARY と SET MSG DETAIL

計算スクリプトで SET MSG SUMMARY と SET MSG DETAIL 計算コマンドを使用すると、次の処理を行うことができます:

- 計算設定(完了通知メッセージが使用可能になっているかどうかなど)の表示
- 作成済、読取り済および書込み済のデータ・ブロック数に関する統計の提供
- 計算済データ・セル数に関する統計の提供

また、SET MSG DETAIL により、Essbase でデータ・ブロックが計算されるたびに、情報メッセージが表示されます。SET MSG DETAIL は、データ・ブロックの計算順序を調べる場合や、高機能な再計算をテストする場合に有用です。

注意 SET MSG DETAIL を使用すると処理のオーバーヘッドが高くなるため、このコマンドはテスト計算のときにのみ使用してください。

SET MSG SUMMARY で生じる処理のオーバーヘッドは、データベースのサイズに応じて約 1% - 5%であるため、このコマンドはあらゆる計算に適しています。

SET NOTICE

計算スクリプトで SET NOTICE 計算コマンドを使用して、計算済のデータベースのパーセンテージを示す計算完了通知を表示できます。SET MSG SUMMARY コマンドを SET NOTICE コマンドと一緒に使用すれば、完了通知間の計算の進捗状況を表示できます。非常に小さいデータベースで使用する場合を除き、完了通知によって計算パフォーマンスが大幅に低下することはありません。

シミュレート計算を使用した計算時間の見積り

計算スクリプトで SET MSG ONLY を使用して、計算のシミュレーションを実行できます。シミュレート計算により、同じデータとアウトラインに基づく実際の計算のパフォーマンスを分析するために役立つ結果が得られます。

SET NOTICE HIGH などのコマンドでシミュレート計算を実行することで、それぞれの疎次元の計算にかかる相対的な時間を把握できます。その後、1 つ以上の次元で実際の計算を実行することで、フル計算にかかる時間を見積ることができます。これは、シミュレート計算の実行にかかる時間が、実際の計算の実行にかかる時間に比例するためです。

たとえば、計算を 9:50:00 AM に開始し、最初の通知のタイムスタンプが 09:50:10 AM、2 番目のタイムスタンプが 9:50:20 AM の場合、この部分の計算にはそれぞれ 10 秒かかったことがわかります。次に、最初の部分のみで実際の計算を実行し、その実行に 30 秒かかったことに気づいた場合、その他の部分にも 30 秒かかることがわかります。合計 2 つのメッセージがあるとすると、実際の計算には約 60 秒 ((20 / 10 * 30 = 60 秒)かかることになります。

次の項目では、シミュレート計算の実行方法、およびシミュレート計算を使用して計算時間を見積る方法について説明します。

シミュレート計算の実行

計算時間を見積る場合は、実際のデータベースに基づいたデータ・モデル上でシミュレート計算を実行する必要があります。

➤ シミュレート計算を実行するには:

1 情報を取得するすべての次元および詳細レベルを使用して、データ・モデルを作成します。

2 すべてのデータをロードします。この手順では、データベースにロードされたデータのみが計算されます。

3 次のエントリを使用して計算スクリプトを作成します:

```
     SET MSG ONLY;
SET NOTICE HIGH;
CALC ALL;
```

密次元に動的計算を使用する場合、CALC ALL コマンドを、計算の必要な特定の次元に置換します(CALC DIM EAST など)。

注: スクリプトを検証しようとすると、Essbase でエラーが報告されます。このエラーは無視してください。

4 スクリプトを実行します。

5 アプリケーション・ログで最初の疎の計算メッセージを見て、メッセージ内の時間を確認します。

6 それ以降の各メッセージの時間も確認します。

7 モデルの密次元で動的計算が行われていないものに対して、計算を実行します:

```
CALC DIM (
DENSE_DIM1
,
DENSE_DIM2
, …);
```

8 モデルの疎次元を計算します:

```
CALC DIM (SPARSEDIM1, SPARSEDIM2, …);
```

9 通知が行われる間隔を見積り、疎の計算結果と照らし合わせて確認します。それによって、計算にかかる時間を見積ることができます。

計算時間の見積り

シミュレート計算を実行した際、結果を記録し、その結果を使用して実際の計算時間を見積ります。

➤ 合計計算時間を見積るには:

1 SET NOTICE HIGH で生成されたアプリケーション・ログのメッセージが、どのくらいの間隔で生成されたかをすべて確認します。

表 185 を参照してください。

2 次の計算方法を使用して、実際の計算にかかる時間を見積ります:

シミュレート計算にかかった合計時間/シミュレート計算における最初の通知間隔 X 実際の計算における最初の間隔

表 185 ログ・メッセージの間隔の例

計算通知番号	計算シミュレーションの間隔(秒)	疎次元の計算間隔(秒)
1	7	45
2	5	
3	6	
4	3	
5	4	
6	2	
7	6	
8	4	
9	3	
10	3	
合計計算時間	43	

この例では、43 / 7 X 45 = 276.4 秒となるため、実際の計算は 276.4 秒かかることになります。

見積りの精度に影響を与える要素

次の問題を除けば、シミュレート計算では通常、誤差 5%の範囲で計算時間を見積ることができます:

- 959 ページの「連鎖的な影響による変動」
- 959 ページの「アウトラインの構造による変動」

これらの要素が存在する場合、Essbase でのシミュレート計算が 30 - 40% (SET NOTICE HIGH によって生成されるメッセージの 30 - 40%)に達したときに、この見積りテクニックによって計算時間がより厳密に予測されます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

連鎖的な影響による変動

計算時間の見積りテクニックとして SET MSG ONLY を使用する場合は、後で CALCNOTICE 間隔と照らし合わせて検証する必要があります。この見積りテクニックの結果は、次のような連鎖的な影響によって変動します:

1. 実際の集計プロセスによって、各ブロックのブロック密度に違いが出る。そのため
2. Essbase で各ブロックがディスクに書き込まれるスピードに違いが出る。そのため
3. 各ブロックがキャッシュで処理されるスピードに違いが出る。そのため
4. 実際の結果は、予想された計算時間と異なる場合がある。

アウトラインの構造による変動

実際の結果を予測とは大きく異なるものにする可能性のあるもう 1 つの要素は、アウトラインの構造です。CALCNOTICE 間隔に基づいた計算では、アウトライン全体を通して処理時間のバランスが均等であることが前提とされます。このバランスを損う可能性のある要素として、次のような状況があります:

- モデル内に他の疎次元と比べて大きな疎次元が 1、2 個含まれている。
- 大きな次元に複数の共有ロールアップを発生させるメンバー構成がある。

シミュレーション結果に基づくアウトラインの変更

シミュレート計算による見積りと分析を実行したら、アウトラインに変更を加えてパフォーマンスを改善できます。

アウトライン内の上から下へ、上位ブロックで最もパーセンテージの増加が少なくなるよう、次のように疎次元を順序付けします:

- モデルのフル・ロードの後の、レベル 0 のブロック 100,000
- 疎次元 1 のみを集計した状態の上位レベルのブロック: 1,000,000
- 疎次元 2 のみを集計した状態の上位レベルのブロック: 3,000,000
- 疎次元 3 のみを集計した状態の上位レベルのブロック: 10,000,000
- 疎次元 4 のみを集計した状態の上位レベルのブロック: 300,000
- 疎次元 5 のみを集計した状態の上位レベルのブロック: 5,700,000

例:

- #4 (メンバー= 10,000、4 レベル)
- #1 (メンバー= 500、2 レベル)
- #2 (メンバー= 100、4 レベル)
- #5 (メンバー= 10,000、4 レベル)
- #3 (メンバー= 20、単層)

シミュレート計算を使用して、上位ブロックのカウントを生成します。これらの数は、前述のアイテムの隣に示されている実際の次元サイズにかかわらず、正確な場合があります。

注意　メンバーの最大のカウントは、必ずしも有効な予測材料とはいえません。

計算によるデータベース・サイズへの影響の見積り

データベース内の現在のブロック数をもとに、CALC ALL で作成されるブロックの数を見積ることができます。

➤ 対話型モードを使用して、計算の結果として作成されるデータベース・サイズを見積るには、次の操作を実行します:

1 データをロードし、CALC ALL コマンドを発行して、平均ブロック・サイズを確認します。

2 MaxL シェルを起動し、Essbase にログオンして、アプリケーションとデータベースを起動します。例:

```
      essmsh
login
      username password
      ;
alter system load application
      appname
      ;
alter application
      appname
       load database
      dbname
      ;
```

3 アプリケーションおよびデータベース名を指定して次の MaxL ステートメントを入力し、ブロック数として戻される値を確認します:

```
      query database
      appname
      .
      dbname
```

```
get estimated size;
```

4 **ブロック数に、データベース内のブロックの平均サイズを掛けます。**

結果は±10%で正確です。

Essbase にクエリーしてデータベースのフル・サイズの見積りを求める場合は、次の条件に注意してください:

- このクエリーは CALC ALL の後に実行する必要があります。その他の計算では正しい結果は得られません。
- 式は疎次元に対するもののみ、正しい結果が得られます。
- データのロック(コミット済アクセス)と組み合せた、メンバーに対するトップダウン計算では、正確な結果は得られません。
- パーティションを見積る必要がある場合は、すべてのパーティションのデータベース・サイズの見積りについて Essbase にクエリーして、その結果を加える必要があります。ソース・データベースのみのサイズについてクエリーした場合、見積りに含まれるのはソース・データベース・サーバー上のデータのみです。

並列計算の使用

サブトピック

- 並列計算
- 現在の並列計算設定の確認
- 並列計算の使用可能化
- 並列計算の追加的タスクの特定
- 並列計算の監視

次の項では、並列計算と、並列計算によってサイトでパフォーマンスを向上させる方法について説明します。

並列計算

サブトピック

- Essbase による実行可能性の分析
- 並列計算のガイドライン
- 並列計算と他の Essbase 機能との関係

Essbase には、計算を呼び出す方法が 2 つあります:

- アウトライン自体によって、計算が暗黙的に指定されます。
- ユーザーが作成する計算スクリプトによって、計算が明示的に指定されます。スクリプトには、式と計算命令が含まれます。

計算がどのように呼び出されたかにかかわらず、Essbase では次のいずれかのモードで計算を実行できます:

- シリアル計算はデフォルトです。シリアル計算では、各計算パスは単一のプロセッサで実行されるようスケジュールされます。計算スクリプトから呼び出された場合、計算は、計算スクリプトに現れる順に実行されます。
- 並列計算では、各計算パスがサブタスクに分割されます。別のサブタスクとは独立して実行できるサブタスクは、最高 64 または 128 のスレッドで同時に実行されるようスケジュールされます。32 ビットのプラットフォームで稼働中のブロック・ストレージ・データベースは最高 64 のスレッドをサポートします。64 ビットのプラットフォームで稼働中のブロック・ストレージ・データベースと、集約ストレージ・データベース(32 ビットまたは 64 ビットのプラットフォームのどちらで稼働中の場合も)は、最高 128 のスレッドをサポートします。各スレッドは異なるプロセッサ上にある場合があります。

➤ デフォルトのシリアル計算から並列計算に変更するには、2 つ以下の構成設定を変更してからサーバーを再起動するか、計算スクリプトに命令を追加します。

967 ページの「並列計算の使用可能化」を参照してください。

次の項目では、並列計算の詳細について説明します。

Essbase による実行可能性の分析

Essbase では、並列計算を有効にした計算パスを実行する前に、必ず並列計算の使用が可能かどうか評価されます。

Essbase では、アウトラインと、各計算パスで要求される計算が分析されます。1 つの計算で複数のパスが要求される場合があることに注意してください。動的計算、2 パスとしてタグ付けされたメンバーの存在、またはある種の相互依存性を引き起こす計算などの状況によって、複数のパスの必要性が生じる場合があります。423 ページの「計算パス」を参照してください。

Essbase で並列計算が可能と判定されると、Essbase では計算が互いに独立した小さなタスクに分割されます。計算の間、Essbase ではこれらの小さなタスクは同時に実行されます。

ただし、パスに含まれている式の間に複雑な相互依存性がある場合は、並列計算が使用可能になっていても、Essbase ではシリアル計算が使用されます。このような相互依存性があると、並列計算が不可能になります。

並列計算のガイドライン

アウトラインの構造とアプリケーションの設計によって、並列計算を使用可能にすることで計算パフォーマンスを改善できるかどうかが決まります。並列計算を使用可能にする前に、次のガイドラインについて検討してください。これらのガイドラインは、並列計算の利点を十分に活用するために役立ちます:

- アンコミット・アクセスの分離レベルを使用してください。コミット・アクセスの分離レベルを使用する場合、並列計算はサポートされません。866 ページの「アンコミット・アクセス」を参照してください。
- 計算の中にある 1 つ以上の式によって、Essbase で使用可能になっていても並列計算を使用できない場合があります。並列計算の設定にかかわらずシリアル計算を強制実行する可能性のある式の説明については、964 ページの「式による制限」を参照してください。
- 計算タスクは、アウトラインの最後の n の疎次元に従って生成されます。タスクの識別に使用されるこれらの疎次元は、タスク次元と呼ばれます。タスク次元の数である n は、Essbase によって動的に選択されるか、ユーザーが essbase.cfg ファイルの CALCTASKDIMS に値を指定することで数値が上書きされます。

  アウトライン内の疎次元は、(MaxL ステートメント「query database DBSNAME get dbstats dimension」で報告されるように)次元の実際のサイズに基づいて、最も小さいものから最も大きいものの順に並べます。順序に関するこの推奨事項は、計算機キャッシュ・サイズの最適化のための推奨事項や、その他のアウトラインに関する推奨事項と一貫性があります。(最後の疎次元よりも多くの)追加の次元の使用が必要な場合の説明と、使用される疎次元の数を増やす方法については、968 ページの「並列計算の追加的タスクの特定」を参照してください。
- 並列計算は、パーティション化されていないアプリケーション、および次のようなパーティション・アプリケーションに対して効果的です:
  - 複製パーティション
  - リンク・パーティション
  - 透過パーティション(計算がターゲット・データベースで行われる場合)。essbase.cfg ファイル内の CALCTASKDIMS、または計算スクリプト内の SET CALCTASKDIMS で指定する疎次元の数は、1 (デフォルト値)に設定する必要があります。透過パーティションでの並列計算の使用に課せられる制限の詳細は、965 ページの「透過パーティションによる制限」を参照してください。CALCTASKDIMS または SET CALCTASKDIMS の使用の詳細は、968 ページの「並列計算の追加的タスクの特定」を参照してください。
- データベースに増分再構築を選択し、アウトラインに加えた変更のために再構築が保留となっている場合は、並列計算は使用しないでください。予測不可能な結果が生じる可能性があります。
- 計算やデータ更新などの更新トランザクションは、MDX クエリーやレポート・スクリプトより多くのリソースを消費する要求です。ストレージ・デバイスが高速な場合、Essbase では、より多くの並列計算スレッドを実行して、妥当なスループットを実現することができます; ストレージ・デバイスが高速でない場合は、Essbase で実行可能な並列計算スレッドやその他の更新スレッドの数は少なくなります。更新トランザクションの数は、Essbase.cfg 設定の MAXACTIVEUPDATETRANSACTIONS を使用して制御できます。詳細は、『Oracle Essbase テクニカル・リファレンス』の MAXACTIVEUPDATETRANSACTIONS に関するトピックを参照してください。

並列計算と他の Essbase 機能との関係

次の項では、並列計算と他の Essbase 機能との関係について説明します。

取得パフォーマンス

並列計算のパフォーマンスを最大化するために、最も大きな疎次元をアウトラインの最後に配置すると、取得パフォーマンスの速度が低下する場合があります。92 ページの「クエリーのパフォーマンスの最適化」を参照してください。

式による制限

ある式が存在することによって、シリアル計算が強制的に実行される場合があります。次のような式の配置により、シリアル計算が強制的に実行される可能性があります:

- 密メンバー(すべての保管済メンバー、および保管済メンバーが依存するあらゆる動的計算メンバーを含む)に設定された式によって、並列計算のタスクを識別するために使用される、次元内のメンバーに相関関係が発生する場合。
- 計算スクリプト内で、@VAR、@ARRAY、@XREF または@XWRITE を使用して宣言された変数への参照が、式に含まれる場合。
- @XREF を使用する疎次元メンバーの式で、疎のメンバーの次元がフル計算される場合。@XREF は、他の保管済メンバーに依存しない密の動的計算メンバーにある場合、バッチ計算中にシリアル計算を強制実行しません。
- メンバーの式によって循環的な依存が発生する場合。たとえば、メンバー A にメンバー B を参照する式があり、メンバー B にメンバー C を参照する式があり、メンバー C にはメンバー A を参照する式があるような場合。
- 密または疎のメンバーに設定された式が、並列処理のタスクの特定に使用される次元の、1 つ以上のメンバーに依存する場合。
- 密次元メンバーの式に、他の疎次元のメンバーへの参照が含まれる場合。

並列計算を妨げる可能性のある式を使用する必要がある場合は、その式のメンバーを動的計算としてマークするか、その式を計算のスコープから除外できます。式によって並列計算が妨げられるかどうかを確認するには、アプリケーション・ログを確認します。969 ページの「並列計算の監視」を参照してください。

計算機のキャッシュ

計算パスの開始時に、Essbase で計算機キャッシュのサイズと並列性の度合いが確認され、その後、最大のパフォーマンスを得るための適切な計算機キャッシュのビットマップ・オプションが使用されます。そのため、並列計算に使用されるビットマップ・オプションは、シリアル計算に使用されるものとは異なる場合があります。

たとえば、Essbase でシリアル計算が実行され、複数のビットマップと単一のアンカー次元が使用されるとします。計算機キャッシュを明示的に変更しないと、Essbase では単一のビットマップと単一のアンカー次元のみを使用して並列計算が実行される可能性があります。

ビットマップ・オプションを制御する計算機キャッシュ・モードは、計算パスの開始時に、アプリケーション・ログの次のような項目を確認することで判断できます:

```
Multiple bitmap mode calculator cache memory usage has a limit of [50000] bitmaps.
```

複数のビットマップ・モードで並列計算を使用している場合は、メモリー使用率が高くなることがあります。このような場合は、構成設定 PARCALCMULTIPLEBITMAPMEMOPT を使用して、複数のビットマップ・モードでのメモリーの使用を最適化できます。この設定は、MULTIPLEBITMAPMEMCHECK と一緒に使用することも、別個に使用することもできます。PARCALCMULTIPLEBITMAPMEMOPT を使用可能にするには、`essbase.cfg` ファイルに次の行を追加します:

```
PARCALCMULTIPLEBITMAPMEMOPT TRUE
```

920 ページの「計算機キャッシュのサイズ設定」を参照してください。

透過パーティションによる制限

透過パーティションが含まれる並列計算には次のような制限があります:

- 計算がターゲットで実行されないかぎり、透過パーティション間で並列計算は使用できません。
- CALCTASKDIMS または SET CALCTASKDIMS を 1 (デフォルト)に設定して、固定次元を 1 つのみにする必要があります。
- 複数のビットマップを使用できるように、計算機キャッシュを増やす必要があります。ビットマップ・オプションを制御する計算機キャッシュ・モードは、計算パスの開始時に、次のようなアプリケーション・ログの項目を確認することで判断できます。

  ```
  Multiple bitmap mode calculator cache memory usage has a limit of [50000] bitmaps.
  ```

  920 ページの「計算機キャッシュのサイズ設定」を参照してください。

再構築の制限

増分再構築を選択した場合は、並列計算を使用しないでください。並列計算では増分再構築はサポートされていません。

コミットのしきい値の調整

Essbase では、「内部コミットまでのブロック数」というデータベース設定で指定されたコミットのしきい値が確認されます。この設定により、内部コミットの前に書込みできるデータが 10 MB 未満である必要がある場合は、計算パスの実行中

のコミットのしきい値が、Essbase により自動的に 10 MB に増やされます。この設定が 10 MB より大きい場合は、Essbase では設定値が使用されます。

一時的なしきい値の増加が発生した場合は、それを示すメッセージが Essbase によりアプリケーション・ログに書き込まれます。

計算に 10 MB 以上のディスク・スペースを余計に割り当てられる場合は、パフォーマンスを改善するためコミットのしきい値(つまり、コミット発生前のブロック数)を非常に大きな数に増やすことを検討してください。

➤ 現在のしきい値を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データの整合性オプションの設定	Oracle Essbase Administration Services Online Help
MaxL	**display database dbs_name**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	GETDBINFO: 内部コミットの発生までに変更が加えられるブロックの数	『Oracle Essbase テクニカル・リファレンス』

➤ コミットのしきい値を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データの整合性オプションの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database dbs_name set implicit_commit after n blocks**	『Oracle Essbase テクニカル・リファレンス』、MaxL ステートメントのリスト
ESSCMD	SETDBSTATEITEM 21	870 ページの「ESSCMD による分離レベル設定の指定例」

866 ページの「アンコミット・アクセス」を参照してください。

分離レベルの制限

並列計算には「アンコミット」モードを使用する必要があります。

➤ 分離レベルをアンコミット・モードに設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データの整合性オプションの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database dbs_name disable committed_mode**	『Oracle Essbase テクニカル・リファレンス』、MaxL ステートメントのリスト

ツール	トピック	場所
ESSCMD	SETDBSTATEITEM 18	870 ページの「ESSCMD による分離レベル設定の指定例」

866 ページの「アンコミット・アクセス」を参照してください。

現在の並列計算設定の確認

サーバーの構成ファイルまたは使用する計算スクリプトを調べれば、並列計算が使用可能になっているかどうかを確認できます。

➤ サーバーの構成ファイルで、並列計算が使用可能になっているかどうかを確認するには:

1 **サーバーの `essbase.cfg` ファイルをテキスト・エディタで開きます。**

2 **パラメータ CALCPARALLEL を探して、そこで指定されている値を確認します。**

計算を完了するために同時にタスクを実行できるスレッドの数には、1 から 128 の値が指定されています。32 ビットのプラットフォームで稼働中のブロック・ストレージ・データベースは最高 64 のスレッドをサポートします。64 ビットのプラットフォームで稼働中のブロック・ストレージ・データベースと、集約ストレージ・データベース(32 ビットまたは 64 ビットのプラットフォームのどちらで稼働中の場合も)は、最高 128 のスレッドをサポートします。『Oracle Essbase テクニカル・リファレンス』を参照してください。

➤ 計算スクリプトで並列計算が設定されるかどうかを確認するには、SET CALCPARALLEL コマンドを探します。スクリプトによって並列計算が複数回使用可能または使用不可になる場合があるため、スクリプトは慎重に見直してください。

並列計算の使用可能化

並列計算を使用するには、次のいずれかの方法により、並列計算をサーバー・レベル、アプリケーション・レベルまたはデータベース・レベルで使用可能にします:

- 適切な構成設定を `essbase.cfg` ファイルに追加または編集します。

  『Oracle Essbase テクニカル・リファレンス』の「CALCPARALLEL」および「CALCTASKDIMS」を参照してください。

- 計算スクリプトに適切な計算コマンドを追加します。

  『Oracle Essbase テクニカル・リファレンス』の「SET CALCPARALLEL」および「SET CALCTASKDIMS」を参照してください。

並列計算設定では、次のような標準的な優先度ルールが使用されます:

- データベース設定は、アプリケーション設定よりも優先されます。
- アプリケーション設定は、サーバー設定よりも優先されます。

並列計算をサーバー・レベルで設定すると、サーバー上のすべてのアプリケーションとデータベースで実行されるすべての計算で、並列計算が使用可能になります。構成ファイル内で並列計算をサーバー・レベルで設定してから、アプリケーション固有またはデータベース固有のエントリを計算スクリプトに追加することで、個々のアプリケーションまたはデータベースの並列計算を使用不可にできます。

注意　並列計算の使用可能化を試みる前に、この章全体をお読みください。

➤ 並列計算を使用可能にするには:

1. **構成ファイル内で並列計算を使用可能にする予定の場合は、現在のステータスを調べて、エントリが存在するかどうかを確認します。**

   967 ページの「現在の並列計算設定の確認」のプロセスを使用します。

2. **サーバーの `essbase.cfg` ファイルに CALCPARALLEL を追加またはこれを変更するか、計算スクリプトに SET CALCPARALLEL を追加します。**
3. **必要な場合は、並列計算のタスクを特定するために、Essbase で複数の疎次元を使用可能にします。**

   968 ページの「並列計算の追加的タスクの特定」のプロセスを使用します。

4. **構成ファイルに項目を追加した場合は、サーバーを再起動します。**
5. **計算を実行します。**

オラクル社は、CALCPARALLEL の値を、計算に使用できるプロセッサの数より 1 つ小さい数に設定することをお薦めします。これにより残りのプロセッサを、オペレーティング・システムまたはキャッシュからダーティ・ブロックの書出しを行う Essbase プロセスで使用できます。

ヒント：　サイトで必要な場合は、CALCPARALLEL と SET CALCPARALLEL を組み合せて使用できます。たとえば、CALCPARALLEL をサーバー・レベルでオフとして設定し、計算スクリプトを使用して、並列計算を必要に応じて使用可能または使用不可にできます。

並列計算の追加的タスクの特定

Essbase では、デフォルトで、反復的な手法を使用して、並列計算に使用する最適な数のタスク次元が選択されます。

必要な場合は、Essbase で、並列計算に特定の数(n)のタスク次元を使用するよう指定できます。たとえば、最後の疎次元のメンバーに FIX ステートメントがある場合、アウトラインから最後の 1 つ前にある疎次元も含めることができます。これら 2 つの次元の一意のメンバーの各組合せは、潜在的なタスクとして特定され、より多くの小さなタスクが作成されて、並列処理の機会が増え、ロード・バランシングが改善されます。

➤ 並列計算用のタスク次元の数を指定するには:

1. **並列計算が使用可能かどうかを確認します(不明な場合)。**

967 ページの「現在の並列計算設定の確認」を参照してください。
CALCPARALLEL (または計算スクリプト内の SET CALCPARALLEL)がない場合、CALCTASKDIMS は効力を持ちません。

2 **オプション: Essbase では、並列計算に使用するデフォルトの数(n)のタスク次元が選択され、この数は情報メッセージとしてアプリケーション・ログ・ファイルに表示されます; たとえば、次のようなメッセージが表示されます:** `[2]タスク次元を使用して並列化しています。` **デフォルトの n の設定を上書きするには、サーバーの** `essbase.cfg` **ファイルに CALCTASKDIMS を追加または変更するか、スクリプトの最初に計算スクリプト・コマンドの SET CALCTASKDIMS を使用します。**

3 **サーバーの** `essbase.cfg` **ファイルに CALCTASKDIMS を追加またはこれを変更した場合は、Essbase を再起動します。**

4 **計算スクリプトを使用している場合は、スクリプトを実行します。**

注: 場合によっては、CALCTASKDIMS または SET CALCTASKDIMS で指定された数よりも少ない次元を使用して、タスクが特定されることもあります。『Oracle Essbase テクニカル・リファレンス』を参照してください。

並列計算の監視

並列計算に関連するイベントを、アプリケーション・ログで表示できます:

➤ アプリケーション・ログを表示するには、『Oracle Essbase Administration Services オンライン・ヘルプ』の「ログの表示」を参照してください。

Essbase では、それぞれの計算パスについて、次のような様々なタイプの情報をアプリケーション・ログに書き込んで、並列計算をサポートします:

- 並列計算を使用可能にしていて、Essbase により並列計算が実行可能であると判断されている場合は、Essbase により、アプリケーション・ログに次のようなメッセージが書き込まれます:

```
Calculating in parallel with
n
 threads
```

  n は、CALCPARALLEL または SETCALCPARALLEL で指定された、同時タスクの数を表します。

- 並列計算の使用を妨げる(シリアル計算を強制的に実行する)ような式については、Essbase により、アプリケーション・ログに次のようなメッセージが書き込まれます:

```
Formula on ((or backward dependence from) mbr memberName prevents
calculation from running in parallel.
```

  memberName は、該当する式が存在しているメンバーの名前を表します。アプリケーション・ログでこのようなメッセージを確認し、式を削除することを

検討できます。また、可能な場合は、該当のメンバーを動的計算としてタグ付けして、それらのメンバーが計算パスで機能しないようにすることを検討できます。

- 同時実行が可能なタスクの数(CALCPARALLEL や SETCALCPARALLEL の値ではなく、データに基づくもの)を示す次のようなメッセージが、Essbase によりアプリケーション・ログに書き込まれます:

```
Calculation task schedule [576,35,14,3,2,1]
```

この例のメッセージは、576 のタスクを同時に実行できることを示しています。576 のタスクが完了した後、他の 35 のタスクを同時に実行できる、という具合に続きます。

並列計算の効果は、最初のいくつかの段階で最も大きく、同時に実行されるタスクが少なくなっていくにつれて減少します。

並列処理の度合いは、タスク・スケジュールにあるタスクの数によって異なります。この数が多いほど、並行して実行できるタスクの数が増え、パフォーマンスが向上します。

- 空の(計算が含まれない)タスクの数を示すメッセージが、Essbase によりアプリケーション・ログ・ファイルに書き込まれます:

```
[Tue Jun 27 12:30:44 2007]Local/CCDemo/Finance/essexer/
Info(1012681) Empty tasks [291,1,0,0,0,0]
```

この例では、レベル 0 のタスクのうち 291 が空であることが Essbase により示されています。

計算タスク・スケジュールで指定されたタスク数に対する、空のタスク数の比率が 50%を超える(たとえば、291 / 576)場合は、データ・モデル内の希薄さ(密度の低さ)により、並列処理を実行してもパフォーマンスが改善されない可能性があります。

密と疎の割当てを変更して空のタスク数を減らせば、並列計算によってパフォーマンスを向上させることができます。

式の使用

データベース・アウトラインで式を慎重に使用することで、計算パフォーマンスが大幅に向上する場合があります。たとえば、式を計算スクリプトに設定するかわりに、データベース・アウトラインにメンバーの式を設定することで、計算パフォーマンスが向上する場合があります。第 23 章「ブロック・ストレージ・データベース用の式の作成」を参照してください。

次の項で、パフォーマンスに影響を与える式の問題を扱う方法について説明します。

集計

データベース・アウトラインを使用して値をロール・アップする方が、式を使用して値を計算するよりも効率的です。たとえば、図 154 に示すように、メンバー 100 に対するメンバー 100-10、100-20 および 100-30 の集計の方が、次の式をメンバー 100に適用するより効率的です:

```
100-10 + 100-20 + 100-30
```

図 154　集計例

簡単な式の使用

簡単な式を使用するときに、ブロック・サイズが異常なほど大きくはない場合は、計算パフォーマンスに大きな影響を与えることなく、疎次元または密次元のいずれかのメンバーに式を設定できます。ブロック・サイズが大きくなるにつれて、簡単な式による計算パフォーマンスの影響は大きくなります。ブロック・サイズと計算パフォーマンスの間の関係の詳細は、954 ページの「ブロック・サイズとブロック密度」を参照してください。

簡単な式とは、比率やパーセンテージなどのことで、次の要件を満たすものです:

- 異なる次元(疎または密)の値を参照しない。たとえば、簡単な式では製品-> 1 月は参照できません。
- 範囲関数を使用しない。たとえば、簡単な式では@AVGRANGE、@MAXRANGE、@MINRANGE または@SUMRANGE は使用できません。
- 関係関数または財務関数を使用しない。たとえば、簡単な式では、@ANCESTVAL、@NEXT、@PARENTVAL、@SHIFT、@ACCUM または @GROWTH は使用できません。関係関数と財務関数の詳細なリストについては、『Oracle Essbase テクニカル・リファレンス』を参照してください。

式による計算パフォーマンスの影響の詳細は、976 ページの「ボトムアップ計算とトップダウン計算」を参照してください。

複雑な式の使用

複雑な式を使用する場合、次のガイドラインを適用すればパフォーマンスを改善できます:

- 可能であれば、式を密次元のメンバーに適用します。
- 計算スクリプトで FIX コマンドを使用して、必要なデータ・ブロックのみを計算します。495 ページの「FIX コマンドの使用」を参照してください。

- データベースの密度(可能なデータ・ブロックに対する既存のデータ・ブロックの比率)を高くします。

  954 ページの「ブロック・サイズとブロック密度」を参照してください。

複雑な式とは、次のいずれかの要件を満たす式のことです:

- 異なる次元(疎または密)のメンバー(製品-> 1 月など)を参照する。
- 1 つ以上の範囲関数(@AVGRANGE、@MAXRANGE、@MINRANGE、@SUMRANGE など)を使用する。
- 関係関数または財務関数(@ANCESTVAL、@NEXT、@PARENTVAL、@SHIFT、@ACCUM または@GROWTH など)を使用する。関係関数と財務関数の詳細なリストについては、『Oracle Essbase テクニカル・リファレンス』を参照してください。

疎次元のメンバーを適用すると、複雑な式では計算オーバーヘッドがより高くなり、そのためパフォーマンスは低下します。複雑な式が存在することにより、複雑な式を含むメンバーに関連するすべての既存のデータ・ブロックとともに、計算可能なすべてのデータ・ブロックに対しても Essbase で計算を実行することが必要になるため、この問題が発生します。疎次元メンバーに関係関数または財務関数が存在すると、既存のデータ・ブロックとともに、計算可能なすべてのデータ・ブロックに対して Essbase で計算の実行が必要になり、オーバーヘッドはさらに高くなります。

したがって、関係関数または財務関数を含む複雑な式では、関係関数または財務関数を含まない複雑な式よりも大きなオーバーヘッドの増加が発生します。

複雑な式による計算パフォーマンスの影響の詳細は、976 ページの「ボトムアップ計算とトップダウン計算」を参照してください。

複雑な式によるオーバーヘッドの例を 2 つ示します。

- データベースに既存のデータ・ブロックが 90 と、存在可能なデータ・ブロックが 100 ある場合は、複雑な式によるオーバーヘッドは大きくはならず、多くとも 10 の余分なブロックでの値の読取りと、場合によっては値の書込みが行われるのみです。
- データベースに既存のデータ・ブロックが 10 と、それに対して存在可能なデータ・ブロックが 100 ある場合は、複雑な式がない場合に比べて、オーバーヘッドは最高 10 倍もかかります(アウトラインの構造やその他の要因によって異なります)。これは 90 もの余分なブロックが読み取られ、場合によっては書き込まれるためです。

どの場合でも、存在可能なデータ・ブロック数に対する、既存のデータ・ブロック数の比率が低くなるにつれて、計算パフォーマンスのオーバーヘッドは高くなり、パフォーマンスが低下します。

大規模なデータベース・アウトライン内の疎次元に基づく式の最適化

SET FRMLBOTTOMUP 計算コマンドを使用して、大規模なデータベース・アウトライン内の疎次元での式の計算を最適化できます。このコマンドを使用すれば、それ以外の場合はトップダウンで計算される疎のメンバー式を、強制的にボトムアップで計算できます。977 ページの「ボトムアップ計算の強制実行」を参照してください。

トップダウンの式に対してボトムアップの計算を強制すると、CALC ALL コマンドと CALC DIM コマンドを効率的に使用できます。『Oracle Essbase テクニカル・リファレンス』の SET FRMLBOTTOMUP 計算コマンドと CALCOPTFRMLBOTTOMUP 構成設定に関する説明を参照してください。

疎次元のメンバーに割り当てられた定数値

疎次元のメンバーに定数を割り当てると、そのメンバーを含んでいる疎次元メンバーの各組合せに対して、Essbase で自動的にデータ・ブロックが作成されます。

たとえば、メンバーまたは計算スクリプトの式に次の式が含まれているとします:

```
California = 120;
```

この式では、「California」は疎次元のメンバーで、120 は定数値です。「California」のすべての存在可能なデータ・ブロックが Essbase で自動的に作成され、すべてのデータ・セルに値 120 が割り当てられます。何千ものデータ・ブロックが作成される場合もあります。パフォーマンスを向上させるには、不要な値を生成する式を作成しないでください。

- 疎次元で値が必要な交差のみに定数を割り当てるには、次の例のように FIX を使用します:

```
    FIX(Colas,Misc,Actual)
 California = 120;
ENDFIX
```

この例では、「Colas」は疎次元「製品」、「Actual」は密次元「シナリオ」、「Misc」は密次元「メジャー」のメンバーです。「California」(市場次元)、「Actual」(シナリオ次元)、「Misc」(メジャー次元)、「Colas」(製品次元)、および年次元のメンバーの交差には、値 120 が割当てられています。これは、年の特定のメンバーがスクリプトで指定されていないためです。

Sample.Basic には 2 つの疎次元のみ含まれているため、この例によって影響を受けるのは 1 つのブロックのみです。3 つ以上の疎次元が存在する場合は、「California」と「Colas」のすべての疎次元の組合せに対して、必要に応じて Essbase でブロックが作成されます。新しいブロックでは、Essbase によりメジャーとシナリオ(値 120 が割り当てられるもの以外)の値が#MISSING に設定されます。

疎次元のメンバーに割り当てられた非定数値

疎次元のメンバーに非定数値を割当てると、ブロックは「等式によるブロックの作成」の設定に基づいて作成されます。「等式によるブロックの作成」の設定は、データベース・レベルで、データベース・プロパティとして定義されます(378 ページの「非定数値」を参照)。

計算スクリプトでは、「等式によるブロックの作成」設定を一時的に上書きできます。「West」に値がなく、「等式によるブロックの作成」を使用可能にしている場合の、次の計算による影響を考えてください:

```
   West = California + 120;
```

「West」のすべての子に対応するブロック値が#MISSING の場合でも、「West」とのすべての疎メンバーの交差に不要なブロックが作成されることがあります。大規模なデータベースでは特に、不要なブロックの作成と処理に余分な処理時間が必要になります。

疎次元のメンバーに非定数値を割り当てるときに、ブロックの作成を制御するには、次のスクリプトに示すように、SET CREATEBLOCKONEQ ON | OFF 計算コマンドを使用します:

```
   FIX (Colas);
  SET CREATEBLOCKONEQ OFF
  West = California + 120;
  SET CREATEBLOCKONEQ ON
  East = “New York” + 100;
ENDFIX
```

「等式によるブロックの作成」設定はスクリプト開始時は使用不可になっているため、「West」ブロックは「West」の子の値が存在する場合のみ作成されます。後になって、「等式によるブロックの作成」設定が使用可能になるため、「East」のすべてのブロックが作成されます。

注： SET CREATEBLOCKONEQ を使用することで影響を受けるのは、このコマンドが含まれている計算スクリプトの実行中の、ブロックの作成のみです。このコマンドで、データベース全体の「等式によるブロックの作成」設定が変更されることはありません。

次元間演算子の使用

次の状況で次元間演算子(->)を使用するときは、注意を要します:

- 975 ページの「等式の左辺」
- 975 ページの「密次元における等式」

等式の左辺

計算スクリプトのパフォーマンスを向上させるために、等式の左辺で次元間演算子が含まれた式を使用するかわりに、計算スクリプト内で FIX を使用して式の使用を制限します。

たとえば、Sample.Basic で「Jan」->「Sales」の値を 5%増加させるとします。関連するメンバーの組合せのみを計算することでパフォーマンスを向上させるには、FIX コマンドを使用します:

```
   FIX(Jan)
  Sales = Sales * .05;
ENDFIX
```

FIX コマンドを使用すると、指定したメンバーの組合せのみに対して Essbase で式が計算されます。この例では、「Jan」が含まれる組合せのみに対して式が計算されます。

このテクニックを、時間のかかる次元間演算子を使用するアプローチと比較してみます。前述の例の場合、データベース・アウトライン内の「Sales」メンバーに次の式を指定します:

```
Sales(Sales -> Jan = Sales -> Jan * .05;)
```

Essbase でデータベースを循環するにつれて、計算が必要なのは 1 月の組合せのみであっても、次元の「時間」タグが付けられたメンバー(1 月、2 月、3 月など)を含むすべてのメンバーの組合せに対して、式が計算されます。

495 ページの「FIX コマンドの使用」および『Oracle Essbase テクニカル・リファレンス』を参照してください。

密次元における等式

密次元の等式で次元間演算子を使用すると、次の 2 つの条件がともに当てはまる場合には、必要なブロックが Essbase で自動的には作成されません:

- 計算結果が、密次元からの値である。
- オペランドが、疎次元からの値である。

次の方法でブロックを作成して、パフォーマンスの問題を回避します。

- 結果としてメンバーが、密次元からではなく疎次元から得られたことを確認します。次の例では、結果として得られたメンバー「Budget」は疎次元のメンバーです:

```
      FIX(Sales)
  Budget = Actual * 1.1;
ENDFIX
FIX(Expenses)
  Budget = Actual *  .95;
ENDFIX
```

- DATACOPY 計算コマンドを使用して、必要なブロックを作成してから、計算します。502 ページの「DATACOPY を使用した既存ブロックのコピー」を参照してください。
- 密のメンバーの等式が含まれている、次のようなメンバー式を使用します。

```
     FIX(Sales, Expenses)
Budget (Sales = Sales -> Actual * 1.1;
Expenses = Expenses -> Actual * .95;)
ENDFIX
```

式の実行レベルの管理

ブロック・ストレージ・アウトラインでの式には、それによって 1 つ以上のブロック内の式の実行がネストされるといった、別の式に対する依存関係がある可能性があります。このような式を、再帰式といいます。再帰式は、結果としてサーバーの異常終了を招く、大きなループや終わりのないループになることがあります。

異常終了を回避するには、CALCLIMITEFORMULARECURSION 構成設定を使用して、実行レベルが 31 に達した式実行を停止できます『Oracle Essbase テクニカル・リファレンス』を参照してください

ボトムアップ計算の使用

トップダウン計算は、ボトムアップ計算より効率が低くなります。これは、必要以上のブロックが計算されるためです。トップダウン計算はボトムアップ計算より効率が低くなりますが、トップダウン計算は、確実に正しい計算結果を得るために必要な場合があります。

次のトピックでは、様々な状況でどの計算を使用すべきかを説明します:

ボトムアップ計算とトップダウン計算

Essbase では、データベース・アウトラインのフル計算を実行するために、ボトムアップ計算またはトップダウン計算という 2 つの計算方法のいずれかが使用されます。デフォルトでは、Essbase ではボトムアップ計算が実行されます。

ボトムアップ計算の場合は、データベースの計算前に計算が必要なデータ・ブロックが Essbase により決定されます。それにより、計算の必要なブロックのみが Essbase で計算されます。計算はブロック番号が最も小さい既存のブロックから始まり、ブロック番号が最も大きい既存のブロックに達するまで、番号順に各ブロックに対して実行されます。414 ページの「ブロックの計算順序」を参照してください。

データベース・アウトラインに複雑なメンバー式が含まれている場合は、Essbase で該当するメンバーに対するトップダウン計算が実行されます。

簡単な式や複雑な式の、ボトムアップ計算およびトップダウン計算との連携の詳細を理解するには、次の各項を参照してください:

ボトムアップ計算と簡単な式

簡単な式の場合は、Essbase でボトムアップ計算を実行することで、フル計算の実行前に計算の必要のあるブロックが決定されます。たとえば、メンバーに基づく簡単な式(A = B + C など)では、A は、データベース内に B または C が存在する場合のみ計算されます。つまり、計算の開始前に、B と C に対する式の依存関係がわかります。

トップダウン計算と複雑な式

計算の開始前に、Essbase ではデータベース・アウトラインが検索され、次元間の参照が含まれているメンバー式など、トップダウン計算が必要な複雑な式がマークされます。Essbase ではトップダウン式を持つメンバーに達すると、そのメンバーに対するトップダウン計算が行われます。

メンバーに基づく式が複雑なときは、そのメンバーに対して存在可能なすべてのブロックを調べて、既存のブロックの変更または新しいブロックの作成の必要があるかどうかを確認する必要があります。ブロックの、他のブロックに対する依存関係を計算の開始前に確認することは困難です。トップダウンの方法では、式を実行するために Essbase で計算対象のブロックを検索する必要があるため、計算パフォーマンスが低下します。

式をコンパイルすると、その式がトップダウンで計算される場合は、Essbase によりアプリケーション・ログ・ファイルにメッセージが記録されます。

次のような複雑な式を考えてみます:

```
A = B -> D + C -> D
```

この式を計算するために、Essbase で A のあらゆる組合せを検討して、B -> D または C -> D が存在するかどうかを判断する必要があります。

971 ページの「複雑な式の使用」を参照してください。

ボトムアップ計算の強制実行

環境に適しているのであれば、トップダウンの式に対してボトムアップ計算を強制実行できます。

➤ ボトムアップ計算を強制実行するには、次のツールを使用します:

メソッド	説明が記載されている項目	場所
計算関数	式の@CALCMODE	『Oracle Essbase テクニカル・リファレンス』
計算スクリプト・コマンド	SET FRMLBOTTOMUP	『Oracle Essbase テクニカル・リファレンス』
essbase.cfg ファイルの設定	CALCOPTFRMLBOTTOMUP または CALCMODE	『Oracle Essbase テクニカル・リファレンス』

式に対してボトムアップ計算を強制実行すると、通常パフォーマンスは向上します。式に複雑な関数(範囲関数など)が含まれている場合や、式の依存関係が単純でない場合は、トップダウン計算とは異なる結果が得られる場合があります。

注意 本稼動の環境で CALCOPTFRMLBOTTOMUP 設定を変更したり、計算スクリプト・コマンド SET FRMLBOTTOMUP を使用する前に、同じデータのボトムアップ計算による結果とトップダウン計算による結果を比較して、その有効性を確認してください。

キャッシュの管理によるパフォーマンスの改善

次の項では、ブロック・ストレージ・データベースで使用されるキャッシュについて説明します。集約ストレージ・キャッシュについては、1129 ページの「集約ストレージ・キャッシュの管理」を参照してください。

データベースの計算時には、Essbase ではデータベース・アウトライン内の 1 メンバー当たり約 30 バイトのメモリーが使用されます。そのため、データベースに 5,000 のメンバーがある場合は、Essbase ではデータベースの計算に約 150 KB のメモリーが必要になります。

注： 計算スクリプトを組み合せることで、余分なメモリーの使用を回避できます。また、単一の計算スクリプトで並列計算を使用することで、パフォーマンスを向上させることができます。961 ページの「並列計算の使用」を参照してください。

Essbase では、計算パフォーマンス(特に大規模な計算)を最適化するために、メモリーが使用されます。データベース・アウトラインのサイズを変更する場合を除き、使用されるメモリーの量は制御できません。ただし、メモリー・キャッシュのサイズによって Essbase での計算を最適化できます。

Essbase では、次のメモリー・キャッシュを使用して、メモリー使用率を調整します:

- 計算機のキャッシュ。計算機のキャッシュを十分な大きさにして、計算パフォーマンスを最適化できるようにします。
- 動的計算キャッシュ。
- インデックス・キャッシュ。データベースが大規模な場合は、デフォルトのインデックス・キャッシュでは十分でないため、最適な計算パフォーマンスが得られません。
- データ・キャッシュ。
- データ・ファイル・キャッシュ。

注： 初めてデータベースを計算するときに、計算パフォーマンスのために非常に重要なものは、計算機キャッシュのサイズです。可能であれば、計算機キャッシュを、Essbase で最適な計算機キャッシュ・オプションを使用するために十分な大きさにしてください。

917 ページの「キャッシュのサイズ設定」を参照してください。変更を加える前に、トピック全体をお読みください。

ブロック・ロック・システムの処理

ブロックの計算時には、Essbase ではブロックおよびブロックの子が含まれるすべてのブロックがロックされます。Essbase ではブロックが計算されると、ブロックおよび子が含まれるすべてのブロックは解放されます。

デフォルトでは、Essbase ではブロックの計算時に最高 100 ブロックが同時にロックされます。このブロック・ロックの数は、ほとんどのデータベース計算では十分です。疎次元の式を計算する場合は、Essbase で必要な子ブロックすべてを同時にロックできると、最も効率がよくなります。したがって、疎次元の式を計算するときに、集計する子が非常に多い場合(100 を超える場合など)、100 よりも大きいロック数を設定することをお薦めします。この数を増やすことで、Essbase で必要なすべてのブロックを確実にロックでき、パフォーマンスが損われることはありません。

Essbase によるロック動作は、分離レベル設定によって異なります。864 ページの「コミット・アクセスでのロック」および 867 ページの「アンコミット・アクセスでのロック」を参照してください。

注： 疎次元での集計の場合は、子を含むすべてのブロックを Essbase で同時にロックする必要はないため、ブロックのロックを考慮する必要はありません。

SET LOCKBLOCK および CALCLOCKBLOCK の使用

計算スクリプトの SET LOCKBLOCK コマンドを `essbase.cfg` ファイルの CALCLOCKBLOCK 設定とともに使用することで、ブロックの計算中に Essbase で同時にロックできるブロックの最大数を指定できます。デフォルト設定を変更せず、計算中にデフォルトの 100 ブロックでは不十分な場合、要求した時間よりも長い時間が計算に必要になる場合があります。

ユーザーの同時アクセスの管理

Essbase では、ブロック・ロック・システムを使用して、ユーザーに対する同時アクセスが管理されます。このシステムにより、特定のデータ・ブロックを更新または計算できるのが、確実に一度に 1 人のユーザーのみになります。Essbase でのブロックのロックとデータのコミットの処理方法は、分離レベル設定によって変わります。

Essbase でデータ・ブロックが計算されると、排他ロックが生じて、他のユーザーはそのデータ・ブロックを更新または計算できなくなります。ただし、それらのユーザーが読取り専用アクセス権を持つことはできます。Essbase での計算が終了すると、ブロックは解放されます。他のユーザーは、適切なセキュリティ・アクセス権を持っている場合は、その後でブロックを更新できます。

ユーザーがデータ・ブロックを更新しているとき、ブロックはロックされています。データベース計算で、別のユーザーによって更新されるデータ・ブロックが必要な場合、次のような状況のいずれかになるまで計算が待機されます:

- 分離レベル設定が「アンコミット・アクセス」の場合、データ・ブロックの解放。
- 分離レベル設定が「コミット済アクセス」の場合、計算の完了。

Essbase では、計算がデータ・ブロックの解放待ちになっていることを示すメッセージは出力されません。

ロックされたブロックの解放待ちによる計算の遅延が生じないようにするには、Essbase のセキュリティ・オプションを使用して、次のいずれかを実行します:

- 他のユーザーに対するアクセスの拒否
- Essbase からのユーザーの切断

セキュリティ・オプションの詳細は、687 ページの「Essbase ネイティブ・セキュリティ・モードでのユーザーの切断および要求の終了」および 688 ページの「Essbase ネイティブ・セキュリティ・モードでのパスワードとユーザー名の管理」を参照してください。

Essbase によるロックとトランザクションの処理方法については、869 ページの「Essbase によるトランザクションの処理方法の理解」および 862 ページの「データ・ロック」を参照してください。

注： Essbase で計算のためにブロックがロックされるときは、依存する子ブロックに排他ロックは設定されないため、別のユーザーがその子ブロックの値を更新できます。必要な場合は、前述のセキュリティ・オプションを使用して、そのような更新を防ぐことができます。

2 パス計算の使用

アプリケーションに適している場合は、データベース・アウトラインで会計次元のメンバーを 2 パスとしてタグ付けすることで、パフォーマンスを大幅に改善できます。データと計算のニーズの組合せによっては、正確を期すために、2 パスのタグ付けのかわりに計算スクリプトを使用して式を 2 回計算する必要がある場合があります。

次の項を使用して、2 パス計算の詳細を理解してください。会計次元のメンバーを 2 パスとタグ付けしてパフォーマンスを改善できるかどうか、または計算スクリプトを使用して式を 2 回計算する必要があるかどうかを判断してください。この項では、2 パス計算を使用可能にする方法や、2 パス計算の計算スクリプトを作成する方法についても説明します。

2 パス計算と属性メンバーの相互作用については、表 18 を参照してください。

2 パス計算の理解

2 度計算する必要があるメンバー式に 2 パス計算を使用すれば、正確な値を算出できます。

2 パスの式は、可能な場合は常に、Essbase によりデータ・ブロック・レベルで計算され、主計算と同時に計算されます。したがって、Essbase にデータベース全体で特別な計算パスを実行する必要はありません。ただし、場合によっては、Essbase には、データベース全体で特別な計算パスが必要になります。

Essbase での 2 パスの式の計算方法は、時間としてタグ付けされた次元と勘定科目としてタグ付けされた次元があるかどうかと、時間次元と会計次元の密と疎の構成によって異なります。

2 パス計算の例の確認

Profit%に必要な次の計算について考えてみます:

```
Profit % = Profit % Sales
```

表 186 は、密次元としてメジャーと年を持つデータ・ブロックのサブセットを示しているとします。メジャーは勘定科目としてタグ付けされ、年は時間としてタグ付けされています。AGGMISSG 構成設定はオフ(デフォルト)になっています。

データ値は入力セルにロードされています。Essbase により、番号 1 から 7 が表示されたセルがこの順序で計算されます。たとえば、Profit % -> Jan が最初に計算されます; Profit % -> Qtr1 には、複数の集計パスがあります。

表 186　2 パス計算の例: データおよび計算順序

メジャー->年	1 月	Feb	Mar	第 1 四半期
利益	75	50	120	**5**
売上	150	200	240	**6**
利益率	**1**	**2**	**3**	**4, 7**

注： データベース構成によってセルの計算順序がどのように変わるかについては、416 ページの「セルの計算順序」を参照してください。

Essbase では、次の計算順序が使用されます:

1. Essbase で、利益率-> 1 月、利益率-> 2 月、利益率-> 3 月、および利益率->第 1 四半期(表内の 1、2、3、4)に対して、式 `Profit % Sales` が計算されます。
2. Essbase で、1 月、2 月および 3 月の値(表内の 5、6)が加算され、利益->第 1 四半期と売上高->第 1 四半期が計算されます。
3. Essbase で、Profit % -> Jan、Profit % -> Feb、および Profit % -> Mar(表内の 7)の値が加算され、Profit % -> Qtr1 が計算されます。このようなパーセンテージの加算によって値 125%が得られますが、これは正しい結果ではありません。

表 187　2 パス計算の例: 正しくない結果

メジャー->年	1 月	Feb	Mar	第 1 四半期
利益	75	50	120	245 (5)
売上	150	200	240	590 (6)
利益率	50% (1)	25% (2)	50% (3)	0% (4) 125% (7)

4. データベース・アウトラインで Profit%に 2 パスとタグ付けすると、Essbase では `Profit % Sales` の式を使用して利益率の値が再計算され、正しい結果が得られます。

表 188　2 パス計算の例: 正しい結果

メジャー->年	1 月	Feb	Mar	第 1 四半期
利益	75	50	120	245 (5)
売上	150	200	240	590 (6)
利益率	50% (1)	25% (2)	50% (3)	0% (4) 125% (7) 42% (8)

複数の計算パスの詳細については、423 ページの「計算パス」を参照してください。

2 パス計算と高機能計算の交差の理解

次の項では、2 つのシナリオを詳しく説明します。高機能計算を使用している場合は、データベースの構成に合うシナリオを使用してください。それぞれのシナリオでは、Essbase で 2 パスの式を正確に計算できるようにする方法が示されています。

これらのシナリオでは、高機能計算の概念を理解していることが必要です。第 26 章「高機能計算についての理解」を参照してください。

シナリオ A

このシナリオでは、式をアウトラインに設定し、次に特定の式に適宜 2 パスをタグ付けして、パフォーマンスを最適化します。

2 パスの式のために特別な計算パスは不要

Essbase ではデータ・ブロックの計算中に 2 パスの式が計算されるため、Essbase においてはデータベース全体で特別な計算パスを実行する必要はありません。

クリーンとしてマークされる全データ・ブロック

計算後、すべてのデータ・ブロックは、高機能計算に備えてクリーンとしてマークされます。

アウトラインでメンバー式に 2 パスとタグ付けすると、Essbase では各データ・ブロックの計算中に 2 パス計算が行われます。ただし、計算スクリプトで式を繰り返すときは、式を再計算するために Essbase でデータ・ブロックの読取りやメモリーへの書込みが必要になります。

シナリオ B

このシナリオでは、計算スクリプトを作成して式の計算を実行し、パフォーマンスを最適化します。

2 パスの式における特別な計算パス

Essbase では、データベースが計算されてから、特別な計算パスを実行して 2 パスの式が計算されます。すべてのデータ・ブロックが最初のデータベース計算の後にクリーンとしてマークされても、Essbase では 2 パスの式に関連するブロックのクリーン・ステータスは無視され、これらのブロックは再計算されます。

クリーンとマークされない、2 パス計算式のデータ・ブロック

初回の計算後、Essbase では高機能計算のためにすべてのデータ・ブロックがクリーンとしてマークされます。データベースの 2 回目の計算パスでは、2 パスの式に必要なデータ・ブロックが Essbase で再計算されます。ただし、2 回目の計算はデータベースの部分的な計算であるため、再計算されたブロックは Essbase でクリーンとしてマークされません。高機能計算をオンにしてデータベースを再計算すると、このようなデータ・ブロックが不必要に再計算される場合があります。

データベース構成により Essbase でシナリオ B を使用できる場合は、2 パスの式の計算に計算スクリプトを使用することを検討してください。計算スクリプトを使用すると、Essbase ではデータベースで余分な計算パスが実行されますが、計算後にすべてのデータ・ブロックが Essbase でクリーンとしてマークされたことを確認できます。985 ページの「2 パス計算および高機能計算用の計算スクリプトの作成」を参照してください。

2 パス計算タグまたは計算スクリプトの選択

勘定科目メンバーを 2 パスとタグ付けすると、パフォーマンスが向上する場合がありますが、アプリケーションによってはこの方法は使用できません。次の制限を確認して、最高のパフォーマンスと精度を得るために、「2 パス」タグを適用するべきか、計算を 2 回実行する計算スクリプトを作成するべきか確認してください:

- メンバーは、会計次元としてタグ付けされている次元の場合は、2 パスとタグ付けできます。データベースでデフォルトの計算を実行すると、データベース・アウトライン内の勘定科目としてタグ付けされた次元内にある場合は、2 パスとしてタグ付けされた式が Essbase で自動的に再計算されます。

- 次元のメンバーは、動的計算メンバーまたは動的計算および保管メンバーであれば、2 パスとタグ付けできます。第 27 章「データ値の動的計算」を参照してください。
- 「2 パス」タグでパフォーマンスが向上する場合でも、正確な結果を得るために、計算スクリプトを使用して 2 パスの式を計算する必要がある場合があります。985 ページの「2 パス計算および高機能計算用の計算スクリプトの作成」を参照してください。
- 高機能計算を効率的に使用するには、「2 パス」タグではなく計算スクリプトを使用します。982 ページの「2 パス計算と高機能計算の交差の理解」を参照してください。
- 982 ページの「シナリオ A」で説明したように、データベース構成に応じて Essbase でシナリオ A を採用する場合や、式が別のデータ・ブロックの値を参照している場合は、計算スクリプトを使用して、式を 2 回計算する必要があります。
- 983 ページの「シナリオ B」で説明したように、データベース構成に応じて Essbase でシナリオ B を採用する場合は、計算スクリプトを使用して、2 パスの式を計算した方がよい場合があります。

デフォルト計算における 2 パスの有効化

デフォルト設定により、デフォルト計算での 2 パス計算を使用可能にできます。2 パス計算が使用可能になっている(デフォルト設定)データベースでデフォルト計算を実行すると、データベース・アウトライン内の勘定科目としてタグ付けされた次元にある、2 パスとタグ付けされた式が Essbase で自動的に計算されます。これは、デフォルトの計算スクリプトがカスタマイズされている場合でも同様です。

➤ デフォルト計算を実行するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ブロック・ストレージ・データベースの計算	Oracle Essbase Administration Services Online Help
MaxL	**execute calculation**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	CALCDEFAULT	『Oracle Essbase テクニカル・リファレンス』

➤ 2 パス計算を使用可能にするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	デフォルト計算での 2 パスの使用	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATE	『Oracle Essbase テクニカル・リファレンス』

2 パス計算および高機能計算用の計算スクリプトの作成

高機能計算で 2 パス計算を実行するための計算スクリプトを作成して、計算をできるかぎり正確かつ高速にするには、次のような方法を使用します:

- 2 パスの式を再計算する計算スクリプト・コマンドの前に、SET UPDATECALC OFF コマンドを追加して、高機能計算を使用不可にします。高機能計算が使用可能になっている場合(デフォルト)、Essbase ではクリーンとマークされていないデータ・ブロックのみが計算されますが、高機能計算が使用可能になっているデータベースのデフォルト計算を実行すると、すべてのデータ・ブロックがクリーンとマークされるため、Essbase で 2 パスの式の再計算は実行されません。
- 計算スクリプトを使用すると、2 パスの式は Essbase で自動的には再計算されません。CALC TWOPASS コマンドを使用してください。
- CALC ALL のデフォルトの計算を変更済で、高機能計算が使用可能な場合、最初の計算後にデータ・ブロックがクリーンとしてマークされない場合があります。第 26 章「高機能計算についての理解」を参照してください。また、Oracle Essbase Administration Services Online Help の「デフォルト計算の設定」も参照してください。

データベースの最初のフル計算を実行するときに、高機能計算を使用してパフォーマンスを向上させるには、計算の要件やアウトラインの構造に応じて、次のいずれかの方法を使用します:

- 985 ページの「大規模なインデックスでの高機能計算の使用」
- 987 ページの「小規模なインデックスでの高機能計算の使用」
- 988 ページの「2 パス式のための高機能計算の無効化」

これら 3 つのオプションは、次の状況を例として使用しています:

アウトラインには勘定科目としてタグ付けされた次元があり、その次元は密次元です。すべての製品の売上のパーセンテージとして、各製品の売上を計算するとします。次の式で、次元が計算されるものとします:

```
Sales % Sales -> Product
```

Essbase で各製品のデータ・ブロックが計算されても、「Sales」->「Product」の値はまだ計算されていないため、総収入のパーセンテージとしての各製品の売上の結果は正しくありません。

大規模なインデックスでの高機能計算の使用

大規模なインデックスで高機能計算を使用する場合は、次のいずれかのオプションを使用して、最適なパフォーマンスを得ることができます。これら 3 つのオプションはすべて、同じタスクを実行します。

1. 高機能計算を使用可能にします。

2. データベース全体を計算し、データ・ブロックをクリーンとしてマークします。
3. 高機能計算を使用不可にします。
4. この計算がデータベースの部分的な計算であっても、再計算されたブロックはクリーンとしてマークされます。SET CLEARUPDATESTATUS AFTER コマンドを使用しない場合、Essbase でデータ・ブロックがクリーンとしてマークされるのは、データベースのフル計算の後のみです。
5. Essbase でデータベースを循環して、該当するメンバー(この例では「Share of Sales」)の式のみが計算されるか、またはデータベース・アウトラインにおいて、「2 パス」タグが付けられた式すべてが計算されます。

計算スクリプトの使用

高機能計算が使用可能な状態でデータベースのフル計算を実行する計算スクリプトを作成するには、このモデルを使用します:

```
   SET UPDATECALC ON;
CALC ALL;
SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
"Share of Sales" = Sales % Sales -> Product;
```

計算スクリプトと「2 パス」タグの使用

➤ メンバーを 2 パスとタグ付けし、計算スクリプトを使用して、まずデータベース全体を計算してから、次に 2 パス・メンバーを計算するには:

1 **式をデータベース・アウトラインに設定し、それに 2 パスをタグ付けします。**
2 **勘定科目とタグ付けされた次元内の該当するメンバー(この例では「Share of Sales」)に、式を設定します。**
3 **次のように、データベースのフル計算の次に、2 パス計算を実行する計算スクリプトを作成します:**

```
     SET UPDATECALC ON;
CALC ALL;
SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
CALC TWOPASS;
```

クライアントと計算スクリプトの使用

➤ クライアントからデフォルト計算を実行し、次に計算スクリプトを使用して式の計算を実行するには:

1 **高機能計算のデフォルトを変更した場合は、使用可能にします。**

2 表 189 にリストされているツールのいずれかを使用して、フル計算を実行します。

3 この例のような計算スクリプトを使用して、高機能計算を使用不可にし、式の計算を実行します:

```
     SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
"Share of Sales" = Sales % Sales -> Product;
```

または:

```
     SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
CALC TWOPASS;
```

表 189　フル計算を実行する方法

ツール	トピック	場所
Administration Services	データベースの計算	Oracle Essbase Administration Services Online Help
MaxL	**execute calculation**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	CALCDEFAULT	『Oracle Essbase テクニカル・リファレンス』

第 26 章「高機能計算についての理解」、第 23 章「ブロック・ストレージ・データベース用の式の作成」、および第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」を参照してください。

小規模なインデックスでの高機能計算の使用

➤ インデックスが小さい場合に高機能計算を使用するには:

1 データベースを計算する計算スクリプトを作成します。ただし、計算済のデータ・ブロックが、Essbase でクリーンとしてマークされないように設定します。

2 すべてのデータ・ブロックをクリーンとしてマークし、データ・ブロックの再計算は行いません。

```
     SET CLEARUPDATESTATUS OFF;
CALC ALL;
CALC TWOPASS;
SET CLEARUPDATESTATUS ONLY;
CALC ALL;
```

この例のスクリプトでは、Essbase で次のタスクが実行されます:

3 SET CLEARUPDATESTATUS OFF コマンドにより、計算済のデータ・ブロックは Essbase でクリーンとしてマークされなくなります。

4 最初の CALC ALL コマンドにより、Essbase でデータベースが循環されて、すべてのダーティ・データ・ブロックが計算されます。Essbase では、計算済データ・ブ

ロックがクリーンとしてマークされません。データベース・アウトラインで 2 パスとしてタグ付けされた式は、Essbase で自動的には再計算されません。

5 CALC TWOPASS コマンドにより、Essbase でデータベースが循環されて、データベース・アウトライン内の勘定科目のタグの付いた次元にある、2 パスとタグ付けされた式が再計算されます。必要なデータ・ブロックが、前の CALC ALL でクリーンとしてマークされていないので、Essbase で式が再計算されます。再計算済のデータ・ブロックは Essbase でクリーンとしてマークされません。

6 SET CLEARUPDATESTATUS ONLY コマンドで、Essbase でデータ・ブロックをクリーンとしてマークしても、そのデータ・ブロックを計算しないよう設定します。このコマンドにより、計算が使用不可になります。

7 最後の CALC ALL コマンドにより、Essbase でデータベースが循環され、すべてのデータ・ブロックがクリーンとしてマークされます。Essbase でインデックスが検索され、データ・ブロックはクリーンとしてマークされます。データ・ブロックは計算されません。

2 パス式のための高機能計算の無効化

➤ 2 パス式のための高機能計算を無効にするには、次のようなタスクを許可する計算スクリプトを作成します:

- 高機能計算を使用不可にする。
- フル計算を実行する。
- 次のように、2 パスの式を繰り返す:

```
        SET UPDATECALC OFF;
CALC ALL;
"Share of Sales" = Sales % Sales -> Product;
```

メンバー・セット関数またはパフォーマンスの選択

「動的計算」または「動的計算および保管」タグが付いたメンバーを参照するクエリーや計算は、そのメンバーに次のいずれかの関数を含む式がある場合、同じメンバーに関するクエリーや計算よりも、かかる時間が大幅に長くなる可能性があります:

- @CURRMBR
- @PARENT
- @SPARENTVAL
- @ANCEST
- @SANCESTVAL

パフォーマンス速度が遅い場合は、「動的計算」タグを削除するか、式からこれらの関数を削除するか、いずれかを考慮してください。

#MISSING 値の集計

次元メンバーの組合せに対する値が存在しない場合は、Essbase では#MISSING という組合せの値が設定されます。Essbase では、#MISSING 値とゼロ(0)値の扱いは異なります。

#MISSING の計算の理解

表 190 は、Essbase で#MISSING 値がどのように計算されるかを示しています。この表で、X は任意の数を表しています:

表 190　Essbase での#MISSING 値の処理方法

計算/演算	結果
X + #MISSING	X
X - #MISSING #MISSING - X	X -X
X * #MISSING	#MISSING
X / #MISSING #MISSING / X X / 0	#MISSING #MISSING #MISSING
X % #MISSING #MISSING % X X % 0	#MISSING #MISSING #MISSING
X == #MISSING	X が#MISSING でないかぎり FALSE
X != #MISSING X < > #MISSING	X が#MISSING でないかぎり TRUE X が#MISSING でないかぎり TRUE
X <= #MISSING	(X <= 0)
X >= #MISSING	(X >= 0)または(X == #MISSING)
X > #MISSING	(X > 0)
X < #MISSING	(X < 0)
X AND #MISSING: Y AND #MISSING、Y は 0 以外の値を示します 0 AND #MISSING #MISSING AND #MISSING	#MISSING 0 #MISSING

計算/演算	結果
X OR #MISSING: Y OR #MISSING、Y は 0 以外の値を示します 0 OR #MISSING #MISSING OR #MISSING	Y #MISSING #MISSING
IF (#MISSING)	IF (0)
f (#MISSING)	1 つの変数を持つ任意の Essbase 関数に対して#MISSING が戻されます
f (X)	f ドメインでない任意の X、および(具体的に記載された場合を除き)複数の変数を持つ任意の Essbase 関数に対して#MISSING が戻されます

デフォルトでは、#MISSING 値は Essbase によってロール・アップされません。ただし、常にレベル 0 でデータをロードし、親レベルでロードすることがない場合は、#MISSING 値の集計の設定を使用可能にする必要があります。この設定により、計算パフォーマンスが 1%-30%改善されます。パフォーマンスの改善度は、データベースのサイズと構成によって異なります。

注意 データを子レベルではなく親レベルでロードする場合は、デフォルト(#MISSING 値を集計しない)を有効にする必要があります。すべての子メンバーの組合せにゼロ(0)を含むその他の値が設定されている場合は、Essbase によって子の値がロール・アップされて親の値が正確に上書きされるため、デフォルトを安全に変更できます。

集計の変更によるパフォーマンスの改善

集計を行うには、前述のいずれかの方法を使用して、#MISSING 値の集計の設定を使用可能にします。達成できるパフォーマンスの改善度は、データベースの上位レベルのブロックと入力ブロックとの比率によって異なります。

➤ Essbase での#MISSING 値の集計方法を変更するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	計算時の Missing 値の集約	Oracle Essbase Administration Services Online Help
計算スクリプト	SET AGGMISSG	『Oracle Essbase テクニカル・リファレンス』
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM	『Oracle Essbase テクニカル・リファレンス』

注： #MISSING 値の集計の設定を使用可能にする場合は、データ・ブロック内のセルの計算順序が変更されます。416 ページの「セルの計算順序」を参照してください。

#MISSING 値の集計の設定が使用不可にされると、次の状況では、パフォーマンスのオーバーヘッドが特に高くなる点に注意してください:

- 入力データ・ブロックに対する計算済データ・ブロックの比率が低い場合
- 疎次元の親レベルで多くのデータ値をロードする場合。たとえば、Sample.Basic データベースで、疎の市場次元の「East」に多数のデータ値をロードする場合

このような状況では、パフォーマンスのオーバーヘッドは 10%-30%です。計算パフォーマンスが重大な意味を持つ場合は、データベース構成またはデータ・ロード方法を再検討することをお薦めします。

Essbase での#MISSING 値の集計方法の詳細は、989 ページの「#MISSING 値の集計」を参照してください。

#MISSING ブロックの削除

CLEARDATA コマンドを使用して、ブロック内のセルの値を#MISSING に変更できます。データ・ブロックは削除されません。これらの余分なブロックにより、取得と計算のパフォーマンスが低下する場合があります。

#MISSING のブロックによってパフォーマンス速度が低下している場合は、次のいずれかのアクションを実行します:

- CLEARBLOCK コマンドを使用してデータ・ブロックを削除します。
- データをエクスポートしてから、再度インポートします(『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照)。

注： データ値がすでにロードされている場合は、空のブロックを削除するとパフォーマンスが向上します。ただし、新しい値によりそのブロックの作成が必要になる場合は、データ・ロードの処理時間が長くなります。

計算の最適化に関するその他の問題の特定

計算とパフォーマンスの関係については、次の各章でも説明しています:

- 第 27 章「データ値の動的計算」の次の項目を参照してください:
  - 445 ページの「動的計算による利点」
  - 449 ページの「「動的計算」または「動的計算および保管」の選択」
  - 455 ページの「取得時間に対する影響の軽減」
  - 460 ページの「パーティション内のデータの動的計算」
- 第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」の次のトピックを参照してください:
  - 482 ページの「データベース計算用のグローバル設定の指定」
  - 504 ページの「パーティション用の計算スクリプトの作成」

パス計算と SET CLEARUPDATESTATUS コマンドの関係については、『Oracle Essbase テクニカル・リファレンス』を参照してください。

CCONV コマンドを使用して通貨を換算すると、換算後のデータ・ブロックは、高機能計算のためにダーティとしてマークされます。つまり、データベースの再計算時に、換算済のブロックがすべて Essbase で再計算されます。第 26 章「高機能計算についての理解」を参照してください。

59 レポートおよび他の取得機能の最適化

この章の内容

バッファのサイズの変更

レポートを生成するのに必要な時間は、レポート用のデータベースのサイズ、スクリプトに含まれるクエリーの数、およびレポート・バッファのサイズなどの要因によって決まります。

構成可能変数により、取得によって抽出されるデータの保管とソートに使用するバッファのサイズを指定します。バッファは、不要な読取りおよび書込みアクティビティを防ぐために十分な大きさである必要があります。546 ページの「レポート・エクストラクタ」を参照してください。

次のレポート変数は、条件付取得コマンドおよびデータ・ソート・コマンド内で使用します。

取得バッファのサイズの設定

データベース取得バッファは、評価される前の抽出された行データ・セルを保持する、データベースごとのサーバー・バッファです。Essbase の取得ウィザードやレポート・ライターなどの取得ツールでは、このバッファが使用されます。

取得バッファがいっぱいの場合、行は処理され、バッファは再使用されます。このバッファが小さすぎる場合は、この領域を頻繁に再使用することで取得時間が長くなる可能性があります。このバッファが大きすぎる場合、ユーザーが同時にクエリーを実行するときに使用されるメモリーが多すぎて、取得時間も長くなる可能性があります。

次の項では、取得バッファのサイズを管理する方法について説明します。

取得バッファ・サイズの手動設定

各データベースには、変更可能な取得バッファ設定(キロバイト単位)があります。デフォルトのバッファ・サイズは、32 ビットのプラットフォームでは 10 KB、64 ビットのプラットフォームでは 20 KB です。バッファのサイズを大きくする場合は、サイズ制限が 100,000 KB に設定されている場合でも、100 KB を超えないようにすることをお薦めします。

➤ 取得バッファ・サイズを手動で設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	取得バッファ・サイズの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM	『Oracle Essbase テクニカル・リファレンス』

注： アウトラインに、動的計算メンバー、動的時系列メンバーおよび属性メンバーが含まれていない場合、VLBREPORT 構成パラメータを使用すると取得バッファのサイズを動的に決定できます。その場合、データベースの取得バッファ・サイズの設定は無視されます。994 ページの「取得バッファ・サイズの動的な設定の使用可能化」を参照してください。

取得バッファ・サイズの動的な設定の使用可能化

データベースのブロック・サイズが大きく、複数のブロック全体で各ブロックから取得するセルの割合が大きい場合は、Essbase 構成ファイル `essbase.cfg` の VLBREPORT オプションを TRUE に設定することを検討してください。

VLBREPORT 設定が TRUE の場合は、Essbase によって、複数のブロック全体で各ブロック内の 20%を超えるセルへアクセスするレポート用に最適化された取得バッファ・サイズが、内部的に決定されます。この設定は、アウトラインに動的計算、動的時系列または属性メンバーが含まれていない場合のみ有効です。VLBREPORT 構成設定により、手動で設定した取得バッファ・サイズが上書きされます。

VLBREPORT を TRUE に設定すると、各クエリーに必要なメモリー量はわずかに増えますが、同時クエリーや連続クエリーに対する応答時間を短縮できます。

取得ソート・バッファのサイズの設定

取得ソート・バッファのサイズの設定は、レポート・ライターでの取得中、データをソートされた状態に保つサーバー・バッファのサイズを、KB 単位で指定します。ソート・バッファがいっぱいになると、Essbase によってエラー・メッセージが通知されます。

Essbase では、64 ビットのプラットフォームには、32 ビットのプラットフォームよりも大きな取得ソート・バッファが必要です。取得ソート・バッファ制限を超えたことを示すエラーが発生した場合は、この設定を 2 倍に増やしてください。

バッファ・サイズはデータベースごとに調整できます。デフォルトのバッファ・サイズは、32 ビットのプラットフォームでは 10 KB、64 ビットのプラットフォームでは 20 KB に設定されます。

➤ 取得ソート・バッファ・サイズを設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	取得バッファ・サイズの設定	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	SETDBSTATEITEM	『Oracle Essbase テクニカル・リファレンス』

数値の有効桁数の設定

RESTRICT コマンドで使用する NUMERICPRECISION 構成設定によって、レポート・エクストラクタの内部数値比較で考慮される有効桁数が定義されます。データに必要以上の有効桁数を設定すると、取得時間が本来より長くなります。正確な有効桁数レベルを特定し、それに従って NUMERICPRECISION を調整してください。『Oracle Essbase テクニカル・リファレンス』を参照してください。

`essbase.cfg` 設定を変更した場合は、Essbase サーバーを再起動して変更を適用します。

対称型レポートの生成

レポート・ライターを使用する際、レポートの処理時間が第一に重要である場合は、すべてのレポートを対称にすることを検討してください。対称型レポートは、非対称型レポートより処理パフォーマンスが高くなります。

対称型レポートでは、レポート・エクストラクタは 1 パスを使用して、すべての可能なメンバーの組合せに基づいてメンバー・リストを構成します。次の例では、データ値(Actual Jan および Feb と、Budget Jan および Feb)が 1 パスで取得されます。

Symmetric Report Member Combinations Supporting One Pass

```
          Sales South

        Actual        Budget
       Jan     Feb     Jan     Feb
    ========  ========  ========  ========
100-10     757     773     930     950
100-20     450     487     550     590
100-30  #MISSING  #MISSING  #MISSING  #MISSING
 100     1,207   1,260   1,480   1,540
```

非対称型レポートでは、レポート・エクストラクタは可能なメンバーの組合せの各ブロックを別々に取得して処理する必要があります。次の例では、データ値(Actual Jan と Budget Jan)が 2 パスで取得されます。

Asymmetric Report Member Combinations Requiring Multiple Passes

```
      Sales South

     Actual   Budget
       Jan     Jan
    ========  ========
100-10     757     950
100-20     450     590
100-30   #MISSING  #MISSING
 100     1,207    1,540
```

546 ページの「レポート・エクストラクタ」を参照してください。

大きな次元での取得パフォーマンスの向上

大きな次元に対するクエリーには、大抵の場合は大きなリソース要件があります。ただし、これらのクエリーは一般的には疎です。これは、戻される空以外の値の数が、入力クエリーのサイズと比べて小さいことを意味します。このような、大きくても疎のクエリーでは、Essbase でメモリーとプロセッサのリソースをより効率的に使用するため、次のような特殊な MDX およびレポート・ライター関数を使用することをお薦めします。これらの関数では、空以外の組合せのみの処理を試みることによって、取得パフォーマンスが最適化されます。

- MDX の Leaves()関数
- MDX の NonEmptySubset()関数
- MDX の最適化プロパティ: NONEMPTYMEMBER および NONEMPTYTUPLE
- レポート・ライターの Leaves コマンド
- レポート・ライターの Descendants および Idescendants コマンドでの世代またはレベルの指定(Link コマンド内で使用する場合)

データ抽出を最適化するためのメンバーの組織化

レポート・エクストラクタでは、データがレポート・ライターでの特定の順序で抽出されます。フォーマットしたレポートを作成する必要がなく、レポート・ライターを使用している場合は、次のいずれかの方法を使用して、レポートの生成に要する時間を短縮できます:

- レポート・エクストラクタのデータ抽出順序と同じ順序で、レポート・スクリプトを作成します
- 密次元を列にグループ化し、疎次元を行にグループ化します

大規模な運用レポートを作成する場合、これらの方法は特に効果的です。

レポート・エクストラクタでは、下から上、右から左にデータが検索され、まず下端の列メンバーから上端の列メンバーへ検索され、次に、最も内側の行メンバー(右)から最も外側の行メンバー(左)へ検索されます。

次の例では、列メンバーは密次元から取得され、行メンバーは疎次元から取得されます。レポートが読み取られる順序は、(1)のように、丸カッコで囲まれた 1 から 3 の数字で表されます:

```
     Sales South

   (3) Actual   Budget (2)
      Jan    Jan (1)
    ========  ========
100-10     757    950
100-20     450    590
100-30   #MISSING  #MISSING
 100     1,207   1,540
```

データを抽出するための時間を短縮するには、まず密次元をグループ化し、次に、アウトライン内で表示される順序で、疎次元をグループ化します。

レポート列で密次元をネストすると、レポート・エクストラクタでは各データ・ブロックが一度のみ検査され、パフォーマンス時間が向上します。

属性は疎次元であり、動的に計算されるため、Essbase では、レポートに属性次元が含まれる場合は、疎のデータ抽出方法を使用できません。

動的メンバーまたは透過メンバーが含まれるアウトラインのレポートの理解

動的計算および保管メンバーが含まれるデータベース・アウトラインにアクセスするレポートを生成する場合、最初にレポートを生成する時間が、以降に同じデータ・ブロックにアクセスして取得する時間より長くかかります。

動的計算または動的時系列メンバーが含まれるデータベース・アウトラインにアクセスするレポートを生成する場合、Essbase では、レポートの生成のたびにメンバーが計算され、レポート生成の時間が増大します。

第 27 章「データ値の動的計算」を参照してください。

透過メンバーが含まれるレポートを実行すると、必要なデータの取得で、複数のサーバーへアクセスする必要があるため、レポートの生成時間が増大します。

LRO ファイル・サイズの制限

ユーザーがデータベースにリンクできるファイルのサイズは、制限できます。サイズを制限することで、ユーザーが非常に大きなオブジェクトを格納することによってサーバーのリソースが過度に消費されるのを防止できます。191 ページの「ストレージ保持のための LRO ファイル・サイズの制限」を参照してください。

第 X 部

集約ストレージ・データベースの作成、計算および管理

集約ストレージ・データベースの作成、計算および管理の内容：

60 集約ストレージとブロック・ストレージの比較

この章の内容

はじめに

Essbase は、多次元データベースに対するパーシスタンス・メカニズムとして、集約ストレージ・カーネルを備えています。集約ストレージ・データベースにより、データベースの集約時間と次元のスケーラビリティの両面で、劇的な改善が可能になります。集約ストレージ・カーネルはブロック・ストレージ・カーネルに代わるものです。集約ストレージ・データベースは、通常、次のアプリケーションのような、大きな次元性を持つ読取り専用の「ラック・アンド・スタック型」のアプリケーションに対応します:

- 顧客の分析。あらゆる次元のデータが分析されるため、膨大な顧客数になる可能性があります。
- 調達の分析。多くのベンダーをまたがって数多くの製品が追跡されます。
- ロジスティックスの分析。製品の出荷に関して、ほぼリアルタイムの更新が行われます。

サンプル・アプリケーション(ASOsamp)、データ・ファイルおよびルール・ファイルを使用して、集約ストレージの機能を説明します。

集約ストレージ・アプリケーション。概念と設計の点でブロック・ストレージ・アプリケーションとは異なり、ブロック・ストレージ・アプリケーションには適用されない制限があります。次の項で、これらの相違点について説明します。

本質的な相違点

表 191　集約ストレージとブロック・ストレージの本質的な相違点

本質的な相違点	集約ストレージ	ブロック・ストレージ
ストレージ・カーネル	集約をすばやく行えるアーキテクチャになっており、高い次元性と疎データをサポートするよう最適化されています	密次元と疎次元およびそのメンバーによって複数のブロックが定義され、財務アプリケーションに最適です
物理的なストレージ定義	Administration Services の「アプリケーション・プロパティ」ウィンドウの「テーブルスペース」タブ	Administration Services の「データベース・プロパティ」ウィンドウの「ストレージ」タブ
データベースの作成	ブロック・ストレージ・アウトラインを移行するか、アプリケーションの作成後に定義します **注：**　ファイル・システムを使用して、ブロック・ストレージ・アウトラインを集約ストレージ・アプリケーション内にコピーしないでください。アウトラインを移行するには、Administration Services の移行ウィザードを使用してください。	アプリケーションの作成後に定義します
データベースのコピー	サポートされていません	サポートされています
アプリケーションごとにサポートされるデータベースの数	1 つ	複数(推奨されるのは 1 つ)
アプリケーション名とデータベース名	1201 ページの「アプリケーションとデータベースの命名規則」を参照してください。テーブルスペース用に予約された名前は、アプリケーション名またはデータベース名としては使用できません: ● デフォルト ● ログ ● メタデータ ● temp	1201 ページの「アプリケーションとデータベースの命名規則」を参照してください。
アプリケーションおよびデータベースに関する情報の表示	Administration Services の「アプリケーション・プロパティ」ウィンドウと「データベース・プロパティ」ウィンドウに表示されます。 集約ストレージ・アプリケーションでサポートされていない情報や関連のない情報は表示されません。集約ストレージ固有の情報の説明は、「アプリケーション・プロパティ」ウィンドウおよび「データベース・プロパティ」ウィンドウで Oracle Essbase Administration Services Online Help を参照してください。	Administration Services の「アプリケーション・プロパティ」ウィンドウと「データベース・プロパティ」に表示されます
構成設定(essbase.cfg)	集約ストレージ・データベースに適用される設定のリストは、『Oracle Essbase テクニカル・リファレンス』を参照してください。	ブロック・ストレージ・データベースに適用されない設定のリストは、『Oracle Essbase テクニカル・リファレンス』を参照してください。

アウトラインに関する相違点

表 192　集約ストレージとブロック・ストレージのアウトラインに関する相違点

アウトラインの機能	集約ストレージ	ブロック・ストレージ
密次元または疎次元の指定	関係しません	関係します
複数階層の有効化、動的階層または保管階層の指定	関係します	無関係です
動的階層の会計次元とメンバー	次の場合を除き、サポートしています: ● 2 パス計算はサポートされません(計算順序の指定の詳細は1076 ページの「計算順序」を参照) ● 勘定科目タグが付けられた次元に対する属性次元の関連付けはサポートされません ● 共有メンバーに対する追加の制限。1015 ページの「代替階層」を参照してください。	完全サポート
保管階層のメンバー	次の場合を除き、サポートしています: ● ~ (非集計)演算子(ラベルのみメンバーの下のみ)および+ (加算)演算子 ● 式を持つことはできません ● ラベルのみメンバーには制限があります(メンバーのストレージ・タイプを参照)。 ● 動的時系列メンバーはサポートされません ● 保管階層次元は共有メンバーを持つことはできません。複数階層次元内の保管階層は、共有メンバーを持つことができます。1013 ページの「保管階層」を参照してください。	完全サポート
メンバーのストレージ・タイプ	次の場合を除き、サポートしています: ● 動的計算および保管は関係しません ● 保管階層では、ラベルのみメンバーの場合に次の 2 つの制限が適用されます: 　❍ ラベルのみメンバーと同じレベルにあるすべての次元メンバーは、ラベルのみである必要があります 　❍ メンバーの親はラベルのみである必要があります **注：**　動的階層では、ラベルのみメンバーのタグを付けることができます **注：**　ブロック・ストレージ・データベースからの変換時に、属性次元メンバーに動的計算のタグが付けられます。標準次元メンバーでは、動的計算タグが変換されて保管済メンバーとしてタグ付けされます。これにより、Administration Services の「データベース・プロパティ」ウィンドウの「次元」タブにある「保管されたメンバー」の値が変更されます。	属性次元を除き、すべての次元タイプでメンバーのストレージ・タイプをすべてサポートしています
不規則階層および 10 レベルを超える階層	サポートしていますが、パフォーマンスに影響を及ぼす可能性があります	サポート

アウトラインの機能	集約ストレージ	ブロック・ストレージ
アウトラインの検証	• データベースを開始するとき • アウトラインを保存するとき • ブロック・ストレージ・アウトラインから集約ストレージ・アウトラインに変換するとき • ユーザーが要求したとき	• アウトラインを保存するとき • ユーザーが要求したとき
アウトラインのページング	サポート	サポートしていません
データベースの再構築	再構築のレベル(1131 ページの「集約ストレージ・データベースの再構築」を参照)	再構築のレベル(933 ページの「データベースの再構築の最適化」を参照)

計算に関する相違点

表 193 集約ストレージとブロック・ストレージの計算に関する相違点

計算の機能	集約ストレージ	ブロック・ストレージ
データベース計算	データベースの集約。これは、集約ビューを定義することで事前に定義できます	計算スクリプトまたはアウトライン集計
式	次の制限付きで、使用できます: • MDX で書かれた有効な数値式である必要があります。%演算子を含めることはできないため、(value1/value2) * 100 という式に置換してください • Essbase の計算関数はサポートされていません • 動的階層のメンバーの場合は、その他の制限なしに式を使用できます	Essbase の計算関数がサポートされています
計算スクリプト	サポートされていません	サポートされています
属性計算次元	Sum をサポートしています	Sum、Count、Min、Max、Average をサポートしています
計算順序	メンバー式の計算順序は、解決順メンバー・プロパティで定義できます	アウトライン集計手順または計算スクリプトでユーザーが定義します

パーティション化に関する相違点

表 194 集約ストレージとブロック・ストレージのパーティション化に関する相違点

パーティション化機能	集約ストレージ	ブロック・ストレージ
パーティション化	次の制限付きで、サポートしています: • アウトラインは同期化されません	完全にサポートしています

データ・ロードに関する相違点

表 195　集約ストレージとブロック・ストレージのデータ・ロードに関する相違点

データ・ロードの機能	集約ストレージ	ブロック・ストレージ
データ・ロードによるセルのロード	アウトラインの式に値が依存しないレベル 0 のセルのみをロードします	すべてのレベルのセルをロードできます(動的計算メンバーを除く)
データベース値の更新	データ・ロードの最後に集約が存在する場合、集約内の値は再計算されます	値の自動更新は行われません。データ値を更新するには、必要なすべての計算スクリプトを実行する必要があります。
データ・ロード・バッファ	集約ストレージ・データベースへの複数のデータ・ソースのロードは、一時データ・ロード・バッファによって管理されます	サポートされていません
データベースのコンテンツのアトミックな置換	集約ストレージ・データベースにデータをロードするときは、データベースのコンテンツまたはデータベース内のすべての増分データ・スライスのコンテンツを置換できます	サポートされていません
データ・スライス	集約ストレージ・データベースには、データの複数のスライスを含めることができます。データ・スライスはマージ可能です	サポートされていません
共有メンバーの次元構築	親子の構築方法を完全にサポートしています。重複する世代(DUPGEN)の構築方法は、世代 2 (DUPGEN2)までの代替階層の構築に制限されています。	すべての構築方法をサポートしています
日付にマップされたデータのロード	日時次元では、メンバー名ではなく、サポートされている日付フォーマットの文字列を使用して、データをレベル 0 メンバーにロードできます	日時次元タイプはサポートされません

クエリーに関する相違点

表 196　集約ストレージとブロック・ストレージのクエリーに関する相違点

クエリーの機能	集約ストレージ	ブロック・ストレージ
レポート・ライター	データの疎および密に関係するコマンドを除き、サポートしています	完全にサポートしています
Smart View	データの変更(ライトバック)を制限付きでサポートしています	完全にサポートしています
API	サポートされています	サポートされています
エクスポート	次の制限付きで、サポートしています: ● レベル 0 のデータのみエクスポートされます(上位レベルはエクスポートされません) ● 縦欄式エクスポートはありません	サポートされています
MDX クエリー	サポートされています	サポートされています

クエリーの機能	集約ストレージ	ブロック・ストレージ
非レベル 0 のメンバーに関連付けられている属性メンバーのクエリー	非レベル 0 のメンバーの子孫の値を戻します。 1018 ページの「属性クエリーに関する設計時の考慮事項」も参照してください。	非レベル 0 のメンバーの子孫の値として#MISSING を戻します
属性メンバーと共有メンバーのクエリー	共有メンバーは、対応する非共有メンバーに関連付けられている属性を、自動的に共有します	共有メンバーは、対応する非共有メンバーに関連付けられている属性を共有しません
クエリーのロギング	サポートされていません	サポートされています
クエリー・パフォーマンス	複数階層を持つ次元からデータをクエリーする際の考慮事項。 1019 ページの「集約ストレージのクエリーの設計時の考慮事項」を参照してください。	階層は関係しません

機能に関する相違点

表 197　集約ストレージとブロック・ストレージの機能に関する相違点

機能	集約ストレージ	ブロック・ストレージ
別名	サポートされています	サポートされています
通貨換算	サポートされていません	サポートされています

ハイブリッド分析	次の制限付きで、サポートしています: リレーショナル・メンバーと、同じクエリー内の式を持つ Essbase メンバーを含むクエリーは、サポートされていません。
	たとえば、「California」がリレーショナル・メンバーで、メンバー「Profit」が式を持っている場合、次のレポート・スクリプトではエラーが戻されます:
	`Jan`
	`California`
	`Profit`
	`!`
		サポートされています

増分データ・ロード	サポートされています	サポートされています
LRO	サポートされていません	サポートされています
タイム・バランス・レポート	次の制限付きで、サポートしています: ● ゼロのスキップはサポートされていません ● 時間次元には、保管階層のみが含まれている必要があります ● 共有メンバーはレベル 0 にする必要があります	サポートされています
トリガー	更新完了時トリガーをサポートしています	更新時トリガーおよび更新完了時トリガーをサポートしています

機能	集約ストレージ	ブロック・ストレージ
Unicode	サポートされています	サポートされています
差異レポート	サポートされていません	サポートされています
日時次元タイプとリンク属性次元	サポートされています	サポートされていません
ユーザーによるデータの変更(ライトバック)	制限付きのライトバックを使用可能にするために、透過パーティション手法を使用しています	完全にサポートしています

61 集約ストレージ・アプリケーション、データベースおよびアウトライン

この章の内容

この章で示す情報は、集約ストレージ・アプリケーション、データベースおよびアウトラインに適用され、これらのプロセスとブロック・ストレージ・プロセスにどのような違いがあるかに関する情報も含まれています。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 113 ページの「アプリケーションとデータベースの作成」
- 125 ページの「データベース・アウトラインの作成および変更」
- 139 ページの「次元およびメンバーのプロパティの設定」
- 161 ページの「属性の操作」

はじめに

集約ストレージのアプリケーションおよびデータベースと、ブロック・ストレージのアプリケーションおよびデータベースには、概念と設計の点で違いがあります。一部のブロック・ストレージのアウトライン機能は、集約ストレージには適用されません。たとえば、密次元と疎次元の概念は適用されません。第 60 章「集約ストレージとブロック・ストレージの比較」を参照してください。

新しいサンプル・アプリケーション(ASOsamp)、データ・ファイルおよびルール・ファイルを使用して、集約ストレージの機能を説明します。

集約ストレージ・アプリケーションを作成するためのプロセス

このトピックでは、集約ストレージ・アプリケーションを作成するためのプロセスの概略を説明します。

1. 集約ストレージ・アプリケーション、データベースおよびアウトラインを作成します。

   1010 ページの「集約ストレージ・アプリケーション、データベースおよびアウトラインの作成」を参照してください。

2. テーブルスペースを使用して、データの格納および取得を最適化します。

   1128 ページの「集約ストレージ・アプリケーションのストレージ管理」を参照してください。

3. 集約ストレージ・キャッシュの最大サイズを指定します。

   1129 ページの「集約ストレージ・キャッシュの管理」を参照してください。

4. 集約ストレージ・データベース内にデータをロードします。データ・ロードは次元構築と組み合せることができます。

   1052 ページの「集約ストレージ・データベースの準備」を参照してください。
   データのサブセットは Administration Services でプレビューできます。Oracle Essbase Administration Services Online Help の「データのプレビュー」を参照してください。

5. 選択された集約を事前に計算して、取得時間を最適化します。

   1075 ページの「集約ストレージ・データベースの計算」を参照してください。

6. データベース統計を表示します。

   Oracle Essbase Administration Services Online Help の「集約ストレージ統計の表示」を参照してください。

7. 必要に応じて、「集約ストレージ・パーティション・ウィザード」を使用して、ライトバックを使用可能にします。

   1035 ページの「透過パーティションの使用による集約ストレージ・データベースへのライトバックの使用可能化」を参照してください。

8. Oracle ツール(Smart View など)またはサードパーティ製ツールを使用して、データを表示します。

集約ストレージ・アプリケーション、データベースおよびアウトラインの作成

集約ストレージ・データベースを組み込むための、集約ストレージ・アプリケーションを作成する必要があります。集約ストレージ・アプリケーションに組み込めるのは、1 つのデータベースのみです。集約ストレージのアプリケーション、データベースおよびアウトラインは、次の方法で作成できます:

- ブロック・ストレージ・アウトラインを集約ストレージ・アウトラインに変換してから、変換したデータベースとアウトラインを含める新しい集約ストレージ・アプリケーションを作成します。

  Essbase では、ブロック・ストレージ・アウトラインから集約ストレージ・アウトラインに変換する次のシナリオがサポートされています:

  - 非 Unicode ブロック・ストレージ・アウトラインから、非 Unicode 集約ストレージ・アウトライン
  - 非 Unicode ブロック・ストレージ・アウトラインから、Unicode 集約ストレージ・アウトライン
  - Unicode ブロック・ストレージ・アウトラインから、Unicode 集約ストレージ・アウトライン

  次の変換シナリオはサポートされていません:

  - Unicode ブロック・ストレージ・アウトラインから、非 Unicode 集約ストレージ・アウトライン
  - 集約ストレージ・アウトラインから、ブロック・ストレージ・アウトライン
- 集約ストレージ・アプリケーションおよびデータベースを作成します。集約ストレージ・アウトラインは、データベースを作成すると自動的に作成されます。

集約ストレージ・アウトラインへの次元とメンバーのロードの詳細は、1053 ページの「集約ストレージ・データベースでの次元構築」および1057 ページの「集約ストレージ・データベースへのデータのロード」を参照してください。

集約ストレージのアプリケーションとデータベースの情報は、ブロック・ストレージの情報とは異なり、集約ストレージのアプリケーションとデータベースには特有の命名規則が適用されます。表 191 を参照してください。

➤ ブロック・ストレージ・アウトラインを集約ストレージ・アウトラインに変換するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**集約ストレージへのブロック・ストレージ・アウトラインの変換** **集約ストレージ・アウトライン変換ウィザード**	Oracle Essbase Administration Services Online Help
MaxL	**create outline**	『Oracle Essbase テクニカル・リファレンス』

注： ファイル・システムを使用して、ブロック・ストレージ・アウトラインを集約ストレージ・アプリケーション内にコピーしないでください。アウトラインを変換するには、Administration Services の「集約ストレージ・アウトライン変換ウィザード」を使用してください。

➤ 集約ストレージ・アプリケーションまたはデータベースを作成するには、次のツールを使用します:

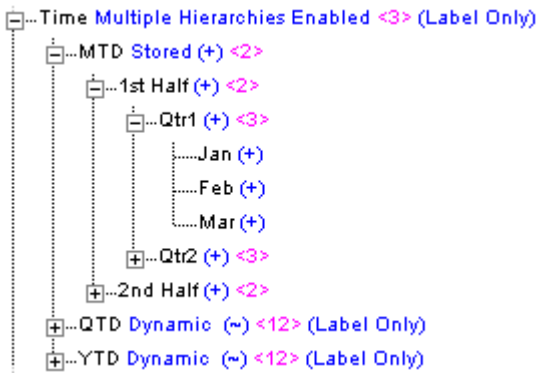
ツール	トピック	場所
Administration Services	アプリケーションの作成 データベースの作成	Oracle Essbase Administration Services Online Help
MaxL	**create application** **create database**	『Oracle Essbase テクニカル・リファレンス』

集約ストレージ・アプリケーション、データベースおよびアウトラインを作成する際は、集約ストレージとブロック・ストレージの相違点と、集約ストレージに固有の問題について考慮してください。次の項と、第 62 章「集約ストレージの時間ベースの分析」も参照してください。

階層

集約ストレージ・アウトラインとブロック・ストレージ・アウトラインでは、次元はレベル内の関連レベルおよびメンバーの 1 つ以上の階層が含まれるように構造化されます。たとえば、ASOsamp.Sample データベースの時間次元(図 155 を参照)には、階層 MTD、QTD および YTD が含まれています。

図 155　時間次元に複数階層とメンバーが含まれるアウトライン



集約ストレージ・データベースでは、次の 2 つのタイプの階層を作成できます:

- 保管
- 動的

この 2 つのタイプの階層の利点と制限には違いがあります。次元には両方のタイプの階層を含まれることがあります。次元内で複数階層を使用するには(すべて保管階層の場合も)、その次元で複数階層を使用可能にする必要があります。

➤ 次元で複数階層を使用可能にするには、次のいずれかのツールを使用して、次元メンバーに複数階層が有効であることを示すタグを付けます:

ツール	トピック	場所
Administration Services	**集約ストレージ・アウトラインでの階層の定義**	Oracle Essbase Administration Services Online Help
MaxL	**import database**	『Oracle Essbase テクニカル・リファレンス』

次元メンバーに「複合階層: 使用可能」タグを付けると、ラベルのみのタグが自動的に付けられます。

次元に「複合階層: 使用可能」タグを付けない場合は、Essbase によって、次元に保管階層タグが自動的に付けられます(動的階層のタグが自動的に付けられる、勘定科目タグが付けられた次元を除く)。

注: 「複合階層: 使用可能」の次元の最初の階層は保管階層でなければなりません。

保管階層

保管階層のメンバーは、アウトライン構造に従って集約されます。集約ストレージ・データベースは集約のために最適化されているため、保管階層のデータ値の集約は非常に高速で行われます。この高速な集約を可能にするため、保管階層のメンバーには次のような制限があります:

- 保管階層は、非集計(~)演算子(ラベルのみメンバーの下のみ)または加算(+)演算子を持つことができます。
- 保管階層は式を持つことはできません。

保管階層には、ラベルのみメンバーに対する制限があります。表 192 を参照してください。

図 156 では、「All Merchandise」階層と「High End Merchandise」階層は保管階層です。「All Merchandise」メンバーと「High End Merchandise」メンバーは階層の最上位で、両方とも保管階層の最上位としてタグ付けされています。

➤ 保管階層を指定するには、次のツールを使用して、階層の最上位のメンバーに保管階層の最上位のタグを付けます:

ツール	トピック	場所
Administration Services	**集約ストレージ・アウトラインでの階層の定義**	Oracle Essbase Administration Services Online Help
MaxL	**import database**	『Oracle Essbase テクニカル・リファレンス』

保管階層の最上位のタグは、次のメンバーに付けることができます:

- **次元メンバー(世代 1)。**次元メンバーに保管階層の最上位のタグが付いている場合、次元全体が単一の保管階層とみなされ、次元内の他のメンバーには保管階層の最上位または動的階層の最上位のタグを付けることはできません。
- **次元メンバーの子(世代 2)。**世代 2 のメンバーに保管階層の最上位のタグが付いている場合、次元内の世代 2 のすべてのメンバーにも、保管階層の最上位または動的階層の最上位のいずれかのタグを付ける必要があります。次元内の最初の階層は、保管階層である必要があります。

勘定科目のタグが付けられた次元は、自動的に動的階層とみなされます。会計次元は、保管階層としては指定できません。

動的階層

動的階層を評価する際、Essbase では、集約ではなく、メンバーと式が計算されます。メンバーと式が評価される順序は、解決順プロパティによって定義されます。1076 ページの「計算順序」を参照してください。

取得時に、Essbase によって、必要なメンバーの組合せが計算されてから、任意の必要なアウトライン・メンバー式が計算されます。動的階層が計算されるため、保管階層からデータを取得する場合よりも、データの取得時間が長くなる場合があります。ただし、データベースを設計する場合、動的階層には次のような利点があります:

- すべての集計演算子を含めることができる。
- 式を持つことができる。

➤ 動的階層を指定するには、次のツールを使用して、階層の最上位のメンバーに動的階層の最上位のタグを付けます:

ツール	トピック	場所
Administration Services	**集約ストレージ・アウトラインでの階層の定義**	Oracle Essbase Administration Services Online Help
MaxL	**import database**	『Oracle Essbase テクニカル・リファレンス』

動的階層の最上位のタグは、次のメンバーに付けることができます:

- **次元メンバー(世代 1)。**次元メンバーに動的階層の最上位のタグが付いている場合、次元全体が単一の動的階層とみなされ、次元内の他のメンバーには動的階層の最上位または保管階層の最上位のタグを付けることはできません。
- **次元メンバーの子(世代 2)。**世代 2 のメンバーに動的階層の最上位のタグが付いている場合、次元内の世代 2 のすべてのメンバーにも、動的階層の最上位または保管階層の最上位のいずれかのタグを付ける必要があります。次元内の最初の階層は、保管階層である必要があります。

注： メンバーのすべての子が非集計演算子(~)を持っている場合、そのメンバーにはラベルのみのタグが付いている必要があります。

勘定科目のタグが付けられた次元は、自動的に動的階層とみなされます。会計次元は、保管階層としては指定できません。

Essbase では、集約ビューの動的階層メンバーは選択できません。1079 ページの「集約ストレージ・データベースの集約」を参照してください。

代替階層

代替階層は次のいずれかの方法でモデル化できます:

- 属性次元として。属性を使用して、次元内のメンバーを論理的に分類します(たとえば、製品次元は、サイズや種類などの属性を持つことができます)。第 10 章「属性の操作」を参照してください。

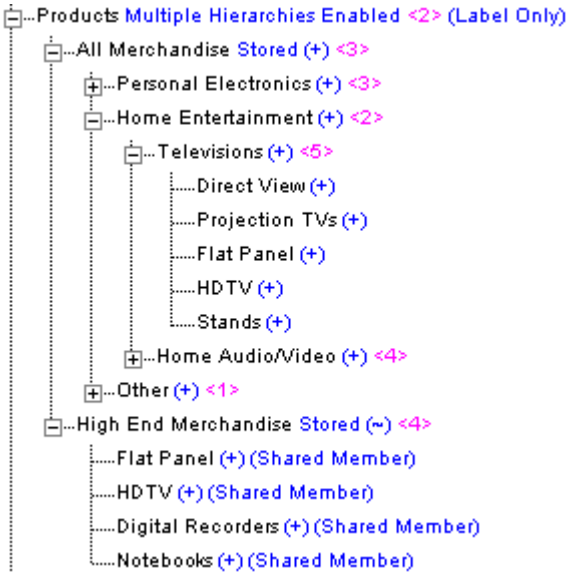
  注： 属性次元を使用して代替階層を作成する場合は、属性次元内のメンバーと基本次元内のメンバーのクロス集計レポート・クエリーを作成できます。たとえば、製品売上情報のクロス集計レポートで、列見出しとしてサイズ属性(小や大など)を示し、行見出しとして製品を示すことができます。共有メンバーを使用して代替階層を作成する場合、共有メンバーとプライマリ階層内の非共有メンバーの同等のクロス集計レポート・クエリーは作成できません。

- 共有メンバーの階層として。代替階層には、アウトライン内の以前の階層の非共有メンバーを参照する、共有メンバーが含まれます。共有メンバーは、それらが参照する非共有メンバーとは異なる階層に従ってロール・アップを行います。動的階層の共有メンバーは式を持つことができます。148 ページの「共有メンバーの理解」を参照してください。表 198 は、ASOsamp.Sample データベースの階層を示しています。この「Products」次元は、図 156 に示されています。

表 198　ASOsamp.Sample の「Product」次元の階層と代替階層の例

製品	階層	代替階層(共有メンバーを含む)
Flat Panel	Products、All Merchandise、Personal Electronics、Home Entertainment、Televisions	Products、High End Merchandise
HDTV	Products、All Merchandise、Personal Electronics、Home Entertainment、Televisions	Products、High End Merchandise

図 156　集約ストレージ・アウトラインに表示された「Product」次元の代替階層「High End Merchandise」



集約ストレージ・アウトラインに代替階層を作成するときには、次のような制限が適用されます:

- メンバーの非共有インスタンスは、アウトライン内で、メンバーの共有インスタンスより前に存在する必要があります。たとえば、図 156 では、メンバー「HDTV」は、「High End Merchandise」の代替階層内の共有メンバーとして表示される前に、「All Merchandise」階層内に表示されます。
- 複数階層が有効な次元の最初の階層には、共有メンバーを含めることはできません。
- 保管階層次元は共有メンバーを持つことはできません。複数階層次元内の保管階層は、共有メンバーを持つことができます。
- 値が二重計算されないようにするため、保管階層に同じ共有メンバーの複数のコピーを含めることはできません。たとえば、保管階層に共有メンバーとその祖先のいずれかを含めることはできません。図 156 の場合、共有メンバー「Televisions」を「High End Merchandise」の子として追加はできません。このように追加すると、「Televisions」がその子である共有メンバー「Flat Panel」および「HDTV」の兄弟になり、「Flat Panel」と「HDTV」の値が 2 回追加されるためです。
- メンバーの非共有インスタンスは、共有メンバーと同じ次元に存在している必要があります(ブロック・ストレージ・アウトラインと同じ)。
- 保管階層には、同じメンバーの非共有インスタンスと共有インスタンスを含めることはできません。
- 動的階層のメンバーが式を持たないレベル 0 メンバーの場合にのみ、動的階層のメンバーの共有インスタンスを保管階層に含めることができます。

注： 集約ストレージ・データベースでは、共有メンバーはその非共有メンバーと関連付けられている属性を自動的に共有します。これは、暗黙の共有メンバー(たとえば、1 つの子のみを持つメンバー)にも適用されます。151 ページの「暗黙的な共有の理解」を参照してください。暗黙的な共有は、「共有しない」プロパティを設定することで防ぐことができます(146 ページの「メンバーによるデータ値の保管方法の決定」を参照)。集約ストレージ・データベースでのこのような共有メンバーと属性の動作は、ブロック・ストレージ・データベースでの動作とは異なります。

属性次元

このトピックでは、属性次元に関しての集約ストレージとブロック・ストレージのデータベースの相違点について説明します。このトピックの情報を利用するには、ブロック・ストレージ・データベースの属性次元の概念に精通している必要があります。第 10 章「属性の操作」を参照してください。

次の情報は、集約ストレージ・データベースで使用する属性次元に適用されます:

- 属性次元で使用できるのは加算(+)集計演算子のみです。
- 指定された属性次元のすべての関連付けは、基本次元の 1 つのレベルを持つ必要があります。たとえば、ASOsamp.Sample データベースの場合、「Store Manager」属性次元の関連付けは「Stores」次元のレベル 0 を持ちます。属性の関連付けには、次の制限が適用されます:
  - レベル 0: 式を持たない動的階層または保管階層のレベル 0 の任意のメンバーと属性を関連付けられます。
  - 非レベル 0: プライマリ保管階層の上位レベルのメンバーのみに属性を関連付けできます。

属性次元には階層タイプはありません。属性次元を動的階層または保管階層として指定できません。Essbase では、属性次元は基本次元の保管代替階層として扱われます。たとえば、Essbase では、ASOsamp.Sample データベースの「Store Manager」属性次元は、「Store Manager」次元が「Stores」次元の保管代替階層であるかのように扱われます。

クエリーの追跡の使用時には、Essbase により属性次元データに対するクエリーが考慮され、集約ビュー選択の属性次元メンバーが組み込まれる場合があります。1085 ページの「使用状況に基づいたビューの選択」および1075 ページの「集約ストレージ・データベースの計算」を参照してください。

注： 非レベル 0 のメンバーと関連付けられている属性次元に対するクエリーでは、非レベル 0 のメンバーの子孫の値が戻されます。集約ストレージ・データベースの属性メンバーでのこのようなクエリーの動作は、ブロック・ストレージ・データベースでの動作とは異なります。

属性クエリーに関する設計時の考慮事項

属性のクエリー・データに基づいてビューを選択および構築する際には、属性データに対する一部のクエリーは取得時に常に動的に計算されるため、クエリー・パフォーマンスに影響を与える場合があります。

属性次元メンバーを含むすべてのクエリーには、基本次元のメンバーを少なくとも 1 つ含める必要もあります。クエリーに単一の属性次元と次元の合計メンバーが含まれている場合は、Essbase によって、そのクエリー・データが集約されるため、クエリー・パフォーマンスが向上する可能性があります。その他の場合は、Essbase によって取得時にクエリーが計算される必要があります。

表 199 で、属性クエリーのタイプと、Essbase でのクエリーの計算方法について説明します:

表 199　属性クエリーと計算のパフォーマンス

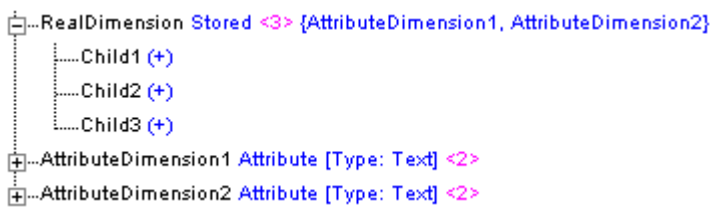
属性クエリーのタイプ	クエリー計算のタイプ
クエリーに基本次元の合計メンバーと 1 つの属性次元のメンバーが含まれる場合。	Essbase でクエリー・データを集約できるため、クエリー・パフォーマンスが向上する可能性があります。
クエリーに基本次元の任意のメンバーと複数の属性次元のメンバーが含まれる場合。	Essbase では、クエリーは、レベル 0 の入力データに基づいて取得時に計算されます。
クエリーに基本次元のメンバー(またはラベルのみのタグが付いた次元メンバー)の子メンバーと 1 つの属性次元のメンバーが含まれる場合。	Essbase では、クエリーは、レベル 0 の入力データまたは基本次元の集約されたデータに基づいて取得時に計算されます。

図 157 で表示されているアウトラインでは、「RealDimension」はその子孫すべての合計です(ラベルのみのタグは付けられません)。クエリーに単一属性次元(たとえば、「AttributeDimension1」)の 1 つ以上のメンバーが含まれ、それが基本次元メンバー(「RealDimension」)と交差している場合は、Essbase でそのデータの集約セルを構築できるため、クエリー・パフォーマンスが向上する可能性があります。

ただし、次のクエリーは常に取得時に計算されます:

- 属性次元(AttributeDimension1 など)のメンバーおよび RealDimension の子のデータを要求するクエリーは、レベル 0 の入力データまたは集約のデータに基づいて、取得時に動的に計算されます。
- 複数の属性次元(「AttributeDimension1」と「AttributeDimension2」など)および基本メンバー次元(「RealDimension」など)のデータを要求するクエリーは、レベル 0 の入力データに基づいて、取得時に動的に計算されます。

図 157　属性クエリーの例のアウトライン



集約ストレージ・アウトラインの設計時の考慮事項

このトピックでは、集約ストレージ・データベース・アウトラインを作成する際の、設計に関する重要な考慮事項を記載ます。これらの設計の考慮事項の実施例については、ASOsamp.Sample データベースを参照してください。集約ストレージ・アウトラインを設計する際は、次の情報について考慮してください:

- 可能なかぎり(動的階層ではなく)保管階層を使用します。
- 必要な場合にかぎり代替階層(共有メンバー)を使用します。
- 階層数を最小限にします(保管階層を追加するたびにビューの選択が遅くなり、集約データのサイズが大きくなる可能性があります)。
- 階層が最初の階層の小さいサブセットの場合は、その小さい階層を動的階層にすることを検討してください。考慮事項には、階層データがクエリーされる頻度と、その階層データが取得時に動的にクエリーされる際のクエリー・パフォーマンスの影響があります。
- 属性のパフォーマンスは保管階層のメンバーの場合と同じです。
- 基本メンバーに対する属性の関連付けレベルが高いほど、クエリーの取得が速くなります(1018 ページの「属性クエリーに関する設計時の考慮事項」も参照)。

集約ストレージのクエリーの設計時の考慮事項

複数階層がある次元のデータをクエリーする場合は、次の方法でデータをクエリーするとクエリー・パフォーマンスが向上する場合があります:

1. クエリーを行う階層を選択します。
2. 詳細なデータを探すために移動します(たとえば、Oracle Hyperion Essbase Spreadsheet Toolkit で階層をズーム・インします)。

動的階層のメンバーと保管階層のメンバーを同じクエリーに含めると、大量の内部メモリー・キャッシュが必要になって、クエリー・パフォーマンスが低下する可能性があります。

集約ストレージ・データベース・アウトラインに対する 64 ビットの次元サイズ制限

集約ストレージ・データベース・アウトラインは 1 つの次元について 64 ビットを超えることはできません。

ある次元に必要なビット数は、代替階層および関連付けられた属性次元のレベル 0 の子を含む、あらゆるレベル 0 の子で使用される中で最大のビット数になります。メンバーを番号付けする目的のため、属性次元はその基本次元の代替階層として扱われます。

通常、ある次元の任意のメンバーに必要なビット数を求める式は次のように表されます:

```
#_bits_member's_parent + log(
x
)
```

ここで、x は親に属す子の数です。

たとえば、メンバーの親がメンバー A で、5 ビットが必要であり、A に 10 の子がいる場合、個々の子に必要なビット数は次のようになります:

```
5 +log(10) = 9 bits
```

次元または階層の最上位のメンバーは通常、0 ビットを使用します。ただし、ラベルのみメンバーで構成されている最上位の世代が 1 つ以上あるとき、ラベルのみメンバーは、(保管済メンバーとみなされないため)メンバー番号を受け取りません。したがって、ラベルのみではない最初の世代に x メンバーがいる場合、このメンバーは log(x)ビットを使用します。このメンバーの下の残りの子は正常に番号付けされます。

同様に、次元または階層が動的な場合、保管または共有されたレベル 0 メンバーのみがメンバー番号を受け取ります。このメンバーに必要なビット数は log (x)で、x は保管または共有されたレベル 0 メンバーの数(つまり、式メンバーではないレベル 0 メンバーの数)を表します。

ただし、代替階層が(非共有の)レベル 0 メンバーを保管している場合、次元(関連付けられた属性次元を含む)内の全階層の各メンバーは追加の log (x)ビットを使用します。ここで x は、この基本次元に対する階層および関連付けられた属性次元の合計数を表します。

次の例では、ASOsamp.Sample データベースの Products 次元を使用します:

Products 次元には次の 2 つの階層があります: All Merchandise および High End Merchandise (High End Merchandise は代替階層)。High End Merchandise には次の 1 つの保管済レベル 0 メンバーがあります: Stored Member。Products 次元には関連する属性次元はありません。

メンバーの All Merchandise および High End Merchandise は log(2) = 1 ビットを使用します。

注： 代替階層 High End Merchandise に保管済のレベル 0 メンバーがなかった場合、各階層(および関連付けられた属性次元)の最上位メンバーは、それぞれ 0 ビットを使用します。

それぞれのレベル 0 の子で必要なビット数の計算は次のようになります:

```
  All Merchandise = 1 bit
Personal Electronics, Home Entertainment, Other = 1 + log(3) = 3 bits
  Digital Cameras/Camcorders, Handhelds/PDAs, Portable Audio = 3 + log(3) = 5
    Children of Digital Cameras/Camcorders = 5 + log(3) = 7

  Children of Handhelds/PDAs = 5 + log(3) = 7

    Children of Portable Audio = 5 + log(2) = 6
  Televisions, Home Audio/Video = 3 + log(2) = 4

  Children of Televisions = 4 + log(5) = 7


    Children of Home Audio/Video = 4 + log(4) = 6
  Computers and Peripherals = 3 + log(1) = 3 **
  Systems, Displays, CD/DVD drives = 3 + log(3) = 5
    Children of Systems = 5 + log(2) = 6
High End Merchandise = 1 bit
  Flat Panel, HDTV, Stored Member = 1 + log(3) = 3 bits
```

メンバーの Computers and Peripherals は、その親 Other と同じビット数(3)になります。

Products 次元のレベル 0 の子で使用される最大のビット数は 7 (Digital Cameras の子および Televisions の子)です。したがって、Products は 7 ビットを使用することになり、これは 64 ビットの次元サイズ制限を下回ります。

次元サイズが 64 ビットを超えると、次のようになります:

- Essbase で、アウトラインの保存時に次のエラーが発生します:

```
      Hierarchy [DimensionName] is too complex. It exceeds the maximum member
  number width of 64 bits. See application log for details.
```

- Essbase で、次のメッセージに類似したメッセージがアプリケーション・ログに記録されます:

```
      Member number for member [level0member] requires [65] bits to encode
  Member [level0member] contributes [5] bits to member number
  Member [level1parent] contributes [20] bits to member number
  Member [level2parent] contributes [20] bits to member number
  Member [level3parent] contributes [20] bits to member number
```

エラーを修正するには、次のいずれかの推奨操作を使用します:

- 可能な場合は、メッセージで参照されているいずれかのメンバーの兄弟をいくつか削除します。兄弟の数を 2 の累乗ずつ削除すると、1 ビット節約できます。たとえば、メンバー番号に対して 5 ビットを占めている level0member に、それ自身も含めて 18 の兄弟があると仮定します。兄弟の数を 16 以下に削除すると、log(16) = 4 のため、1 ビット節約できます。同様に、兄弟の数を 8 以下に削除すると、2 ビット節約できます。
- メッセージで参照されているメンバーの兄弟をいくつか再分類します。たとえば、level0member の 18 の兄弟の半数を、それほど多くの子を持たない別の親に移動します。もしくは、level1parent の 1 兄弟として新規の親を作成し、level1parent の子の半数を新規メンバーの元に移動します。このアプローチでは 1 ビット節約できます。
- 中間レベルをいくつか結合します。たとえば、level0member とその兄弟すべてを level2parent の子に移動してから、level1parent を削除します。このアプローチはさらに複雑ですが、多くのビットを節約できます。

➤ 管理サービス・コンソールで、次元に必要なビット数を参照する手順は次のとおりです:

1 **エンタープライズ・ビューまたはカスタム・ビューから、データベースを選択します。**

2 **右クリックして「編集」、「プロパティ」の順に選択します。**

3 **「データベース・プロパティ」ダイアログ・ボックスで、「統計」タブを選択します。**

「集約ストレージ統計」領域に、各次元で使用されるレベルおよびビットの数が表示されます。

集約ストレージ・データベースの圧縮次元の理解

デフォルトでは、集約ストレージ・データベース内の圧縮次元は会計次元です。圧縮次元を変更すると、データベース全体の再構築がトリガーされます。Essbase では、圧縮次元が単一の動的階層である必要があります。次元に複数階層のような別の階層設定がある場合、その次元は自動的に単一の動的階層に設定されます。元の設定は失われます(別の次元を圧縮と設定しても、下の階層設定は戻りません)。属性次元は、圧縮次元にすることはできず、属性が関連付けられている次元にすることもできません。

圧縮次元の選択は、パフォーマンスに大きな影響を与えることがあります。圧縮次元の適切な候補は、取得パフォーマンスを維持しながら、データ圧縮を最適化するものです。このトピックでは、最適な圧縮次元の選択に関する情報を説明します。

注: このトピックの情報は、ロード済データベースに適用されます。1057 ページの「集約ストレージ・データベースへのデータのロード」を参照してください。

取得パフォーマンスの維持

圧縮次元は動的に計算されるため、圧縮次元を選択する際は、動的に計算される次元の設計に関する考慮事項を検討する必要があります。動的次元は取得時に計算されるため、データの取得時間は保管階層の場合よりも長くなります。

多数のレベル 0 のメンバーを持つ次元に圧縮のタグが付いている場合、上位レベルのクエリーには時間がかかります。これは、それらのクエリーで多数のレベル 0 のメンバーを取得する必要があるためです。ユーザーが大きな次元で多数の上位レベルの取得を行う予定の場合、これは圧縮次元として推奨される候補ではありません。

取得パフォーマンスを維持したデータベース圧縮の管理

圧縮次元を選択する際のもう 1 つの考慮事項は、データベースがどの程度圧縮されるかです。圧縮されたデータベースのサイズは、どの次元に圧縮のタグを付けるかによって変わります。

Administration Services では、圧縮の概算を表示してから、同じダイアログ・ボックスで圧縮次元を選択できます。Administration Services で新しい圧縮次元を選択すると、アウトラインが自動的に再構築されます。

➤ 仮定として圧縮のタグを付けるときに、各次元のデータベースの予想されるレベル 0 のサイズを表示する方法と、圧縮次元を選択する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「集約ストレージの圧縮次元の選択」を参照してください。

圧縮の概算統計の表示

Administration Services と MaxL では、詳細な圧縮およびクエリー統計を表示できます。保管されるレベル 0 のメンバー(取得パフォーマンスに影響する)、バンドル量の平均値と値の長さの平均値(圧縮に影響する)およびレベル 0 のサイズを表示できます。

➤ 詳細なクエリーおよび圧縮統計を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	「データベース・プロパティ」ウィンドウ	Oracle Essbase Administration Services Online Help
MaxL	Query Database (集約ストレージ)	『Oracle Essbase テクニカル・リファレンス』

次の項では、圧縮およびクエリーに関連する各統計について説明します。

保管されるレベル 0 のメンバー

保管されるレベル 0 のメンバーが多数ある次元は、圧縮のタグが付いていると適切に実行されません。動的に計算される次元と同様、圧縮次元からの上位レベルの取得は、通常は時間がかかります。1023 ページの「取得パフォーマンスの維持」を参照してください。

バンドル量の平均値

圧縮の効果が高いのは、値の間に多数の#MISSINGデータがあるアウトライン全体ではなく、次元または階層の連続したメンバー内で値をグループ化する場合です。Essbaseでは、グループの場所とコンテンツをメンバーのそれぞれに個別に保存するのではなく、それらに関する情報を保管することによって、メモリーを節約します。バンドル量の平均値は、グループに保管される値の平均数です。この値は、1から16まで変わることがあり、16が最適です。バンドル量の平均値が高い圧縮次元を選択することは、データベースがより適切に圧縮されることを意味します。

一部のアウトラインでは、圧縮次元内の数値を、頻繁に取り込まれるメンバーがグループ化されるように順序付けることによって、圧縮を改善できます。取り込まれるメンバーがグループ化されていると、より多くのメンバーが各バンドルに組み込まれるため、バンドル量の平均値が大きくなり、圧縮が改善されます。

値の長さの平均値

値の長さの平均値は、セルに保管された値に必要な、バイト単位の平均ストレージ・サイズです。この値は、2バイトから8バイトまでで変わることがあり、2バイトが最適です。圧縮していない場合、セル内の値を保管するには8バイト必要です。圧縮する場合は、必要なバイト数は値の長さに応じてさらに小さくなります。たとえば、10.050001では、圧縮されている場合でも保管するには8バイト必要ですが、10.05では、圧縮されている場合は保管するために2バイト-4バイトのみですみます。値の長さの平均値が小さい次元では、データベースはより適切に圧縮されます。

データ値を小数点以下2桁に四捨五入すると、値の長さの平均値が小さくなり、圧縮の効率が高くなります。

予想されるレベル0のサイズ

このフィールドは、圧縮されたデータベースの概算サイズを示します。予想されるレベル0のサイズが小さいことは、この次元を選択することで適切な圧縮が可能になることを示します。

アウトラインの確認

集約ストレージ・アウトライン・ファイルは、ファイル拡張子(.otl)がブロック・ストレージ・アウトライン・ファイルと同じであり、同等のディレクトリ構造に格納されます。アウトラインを保存すると、Essbaseによりエラーが確認されます。アウトラインを保存する前に、その正確性を確認することもできます。ブロック・ストレージ・データベースの機能の一部は集約ストレージ・データベースには適用されず、確認プロセスでは集約ストレージ・データベースのルールが考慮されます。第60章「集約ストレージとブロック・ストレージの比較」を参照してください。

➤ 集約ストレージ・アウトラインを確認する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「アウトラインの確認」を参照してください。

アウトラインのページング

集約ストレージ・データベース・アウトラインはページング可能です。この機能により、非常に大きいデータベース・アウトラインのメモリー使用率が大幅に削減される場合があります。集約ストレージ・データベースでは、Essbase によって、データベース・アウトラインの一部がメモリーにプレロードされます。その後、データの取得中に、Essbase によって、必要に応じてアウトラインの他の部分がメモリーにページングされます。

集約ストレージ・データベースを作成すると、アウトラインはページング可能なフォーマットで作成されます。「集約ストレージ・アウトライン変換ウィザード」を使用して、既存のブロック・ストレージ・アウトラインを集約ストレージに変換すると、アウトラインは自動的にページング可能なフォーマットに変換されます。

アウトラインをメモリーにページングすることで、Essbase で、非常に大きなアウトライン(たとえば、1,000 万以上のメンバー)を処理できるようになりますが、データ取得時間が長くなる可能性があります。

注： ページング可能なアウトラインが存在する集約ストレージ・データベースには、メモリー・ページが含まれるので、バイナリのブロック・ストレージ・データベースのアウトライン・ファイルより大きくなることがあります。

アウトラインのページング制限

構築可能なアウトラインの最大サイズ(メンバー数)は、次に示すいくつかの要因に依存します:

- Essbase で使用可能なメモリー
- 他のユーザーに割り当てられた Essbase のメモリー容量
- 各メンバー(および各メンバーの別名)に必要なメモリー容量

表 200 に、サポート対象の 32 ビット・オペレーティング・システムで Essbase に使用できるアドレス可能なメモリー容量を示します:

表 200　32 ビット・オペレーティング・システムのアドレス可能なメモリー

オペレーティング・システム	アドレス可能なメモリー
Windows 2003 サーバー	3 GB `boot.ini` ファイル内の設定が必要

オペレーティング・システム	アドレス可能なメモリー
AIX	3.25 GB 最大 13 (256 MB)セグメント。LDR_CNTRL 環境変数を次のように設定する必要があります: `0xD0000000@DSA`
HP-UX	2.9 GB 次のコマンドを使用して、Essbase サーバー・プロセスの ESSSVR にアドレス可能なメモリーを設定する必要があります: `chatr +q3p enable ESSSVR`
Redhat Linux	デフォルトで 3.9GB 使用可能
Solaris	デフォルトで 3.9GB 使用可能

Essbase では、起動時に約 40 MB のメモリーが使用されます。さらに、各種キャッシュには次のようなメモリー割当てが必要です:

- アウトライン・ページング・キャッシュ: 8 MB
- 集約ストレージのデータ・キャッシュ: 32 MB
- 集約ストレージの集約キャッシュ: 10 MB

したがって、Essbase の初期のメモリー・フットプリントは約 90 MB です。さらに、着信クエリー要求を処理するためにメモリーを割り当てる必要があります。この目的のために予約する通常のメモリーは、約 300 MB です。したがって、Essbase に割り当てられる合計メモリーは 390 MB です。

1.85 GB のアドレス可能メモリーがある Windows システムでは、アウトラインの構築とロードに使用できるメモリーは約 1.46 GB (1.85 GB - 390 MB = 1.46 GB)になります。

アウトラインの最大サイズは、アウトラインが次元構築を使用して構築されるか、Essbase にすでにロードされているアウトラインから構築されるかによって異なります。

次元構築の制限

次元構築を使用してアウトラインを構築する際、Essbase では、約 100 バイトにメンバー名のサイズ、メンバーのすべての別名(最大 10 個の別名を許容)のサイズを加算した容量が、メンバーごとに割り当てられます。

サンプルのアウトライン(シングル・バイトのコード・ページを使用)では、メンバー名は平均 15 文字であり、メンバーごとに 1 つの別名(20 文字)が存在しています。各メンバーに必要なメモリー要件は、次のように加算されます:

100 + 15 + 20 バイト= 135 バイト

次元構築で追加できるメンバーの総数は、使用可能なメモリー(1.46 GB または 153,092,060 バイト)をメンバーごとのバイト数(135)で割った数、約 1100 万メンバーになります。

2 GB 以上のアドレス可能メモリーがあるシステムでは、使用可能なメモリーの増分に比例して、アウトラインに可能なサイズが大きくなります。

次元構築が完了すると、dbname.otn ファイルがデータベース・ディレクトリに保存されます。.otn ファイルは、アウトライン再構築プロセスの入力として使用されます。このプロセスでは、古いアウトラインが新しいアウトラインと置き換えられます。再構築中に、古いアウトライン(空の可能性がある)と新しいアウトラインという、アウトラインの 2 つのコピーがメモリーにロードされるため、再構築できるアウトラインの最大サイズは、古いアウトラインのサイズによって変わります。

次元構築では、空のアウトラインで処理が開始するため、メモリーにロードされるアウトラインは 1 つのみです。

ロードされるアウトラインの制限

ランタイム時または再構築中に Essbase にロードされるアウトラインに対するメモリー要件は、次元構築に対するメモリー要件とは異なります。Essbase では、メンバー当たり約 60 バイト、メンバー名のサイズにプラス 5 バイト、メンバーのすべての別名のサイズ(最大 10 個の別名が指定可能)にプラス 5 バイトが割り当てられます。平均のメンバー名が 15 文字で、メンバー当たり 1 つの別名(20 文字)があるサンプルのアウトラインの場合、追加されるメンバーごとのメモリー要件は次のとおりです:

60 + 15 + 5 + 20 + 5 バイト=メンバーごとに 105 バイト

1.46 GB のメモリーが使用可能であると想定すると、ロード可能なアウトラインの最大サイズは 1400 万個のメンバーを格納できるサイズ(1.46 GB / 105 バイト)になります。

1400 万個のメンバーは、再構築中にロードされる 2 つのアウトラインの合計です。たとえば、既存のアウトラインに 500 万個のメンバーがある場合、新しいアウトラインは最大 900 万個のメンバーを持つことができます。増分次元構築では、アウトラインの最大サイズを使用可能にするため、最初に小さいほうの次元を構築し、最後に大きいほうの次元を構築することをお薦めします。

アウトライン・ファイルのコンパクト化

メンバーを集約ストレージ・アウトラインから削除すると、アウトライン・ファイル(.otl ファイル)内のメンバーの対応するレコードには削除済とマークされますが、ファイル内には残されます。メンバーが削除されたり追加されたりしている間、アウトライン・ファイルは増大し続けます。アウトライン・ファイルのサイズは、ファイルをコンパクト化して削除されたメンバーのレコードを削除することで最小化できます。アウトライン・ファイルのコンパクト化により、Essbase によって、アウトラインが再構築されます。また、アウトライン・ファイルのコンパクト化は、他のユーザーまたはプロセスによってデータベースがアクティブに使用されていない場合のみ行われます。Essbase では、アウトラインのコンパクト化ではデータは消去されません。

➤ アウトライン・ファイルをコンパクト化するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	アウトライン・ファイルのコンパクト化	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	COMPACTOUTLINE	『Oracle Essbase テクニカル・リファレンス』

集約ストレージ・アウトラインでの式の作成

式により、データベース・アウトライン内のメンバー間の関係が計算されます。ブロック・ストレージ・アウトラインでの式の使用に精通している場合、集約ストレージ・アウトラインで式を使用する際は次の相違点について考慮してください:

- Essbase は、ブロック・ストレージ・アウトラインで式を記述するための、ネイティブ計算言語(Calc 言語、つまり Calc)を備えています。集約ストレージ・アウトラインで式を記述するには、MDX (多次元式)言語が必要です。
- データベース・アウトライン内のメンバーに直接式を適用します。ブロック・ストレージ・データベースでは、式を計算スクリプト内に配置できます。集約ストレージ・データベースでは、式を計算スクリプト内に配置できません。

この章では、集約ストレージ・データベースで式を記述する際の MDX の使用に焦点を当てて説明します。クエリーを記述する際の MDX の使用の詳細は、第 35 章「MDX クエリーの記述」を参照してください。ブロック・ストレージ・アウトラインでの式の記述の詳細は、第 23 章「ブロック・ストレージ・データベース用の式の作成」を参照してください。また、『Oracle Essbase テクニカル・リファレンス』の MDX に関する項も参照してください。

MDX 式の使用

MDX 式は、常に MDX 数値式である必要があります。MDX の数値式とは、次のアクションのいずれかを実行する関数、演算子およびメンバー名の、任意の組合せです:

- 値の計算
- 条件のテスト
- 算術演算の実行

数値式はセット式とは異なります。セット式はクエリー軸で使用され、メンバーおよびメンバーの組合せを記述します。数値式は値を指定します。

数値式は、計算済メンバーを構築するためのクエリーで使用されます。計算済メンバーはクエリーの WITH セクションに分析目的のために作成される論理メンバーです。ただし、アウトライン内には存在しません。

次のクエリーは、計算済メンバーを定義し、数値式を使用して値を設定しています:

```
    WITH MEMBER
[Measures].[Prod Count]
AS
'Count (
  Crossjoin (
   {[Units]},
   {[Products].children}
  )
 )', SOLVE_ORDER=1
SELECT
{[Geography].children}
ON COLUMNS,
{
 Crossjoin (
   {[Units]},
   {[Products].children}
  ),
  ([Measures].[Prod Count], [Products])
}
ON ROWS
FROM
 ASOsamp.Sample
```

サンプル・クエリーでは、次のように WITH 句で計算済メンバー「Product Count」を「Measures」次元に定義します:

```
    WITH MEMBER
[Measures].[Prod Count]
```

数値式は WITH 句の後に続き、一重引用符で囲まれています。サンプル・クエリーでは、数値式は次のように指定されています:

```
    'Count (
  Crossjoin (
   {[Units]},
   {[Products].children}
  )
 )'
```

SOLVE_ORDER プロパティでは、メンバーと式が評価される順序を指定します。1076 ページの「計算順序」を参照してください。

注： 例のような数値式を構築するための構文規則の説明は、『Oracle Essbase テクニカル・リファレンス』の Count、CrossJoin および Children 関数に関するドキュメンテーションを参照してください。

数値式を MDX 式として使用して、既存のアウトライン・メンバーの値を計算することもできます。

したがって、例のようなクエリーを作成するのではなく、サンプル・クエリーで仮想「Prod Count」を計算したのと同じ方法で、アウトラインで計算された「Prod Count」というアウトライン・メンバーを「Measures」次元に作成します。

➤ 式を持つ計算済メンバーを作成するには:

1 **メンバーを作成します。**

2 **MDX 式をメンバーに添付します。**

例のように「Prod Count」メンバーを作成したことを想定して、次の数式を使用します。これは、例のクエリーで計算済メンバーの作成に使用した数値式と同じです:

```
Count(Crossjoin ( {[Units]}, {[Products].children}))
```

3 **アウトラインを確認し、式を確認します。**

集約ストレージ・データベースからデータを取得するときに、メンバー値の計算に式が使用されます。

式の中では代替変数を使用できます。たとえば、「EstimatedPercent」という代替変数を定義し、代替変数値として様々なパーセンテージを入力できます。118 ページの「代替変数の使用」を参照してください。

アウトライン内のメンバーに式を適用する前に、計算済メンバーを含む MDX クエリーを記述できます。求めている計算済メンバー結果を戻す MDX クエリーを記述できる場合は、数値式の論理をアウトライン・メンバーに適用して、式の検証とテストができる状態です。第 35 章「MDX クエリーの記述」を参照してください。MDX の構文情報は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

集約ストレージ・データベースの式計算

Essbase で集約ストレージ・アウトライン内の式が計算されるのは、データが取得されるときのみです。計算順序は計算結果に影響を及ぼす場合があります。集約ストレージ・アウトライン内の複数次元で MDX 式を使用するときは常に、各メンバーまたは次元に解決順を設定することをお薦めします。1076 ページの「計算順序」を参照してください。

注: 集約ストレージ・データベース計算を設計する際は、MDX 式を含む集約ストレージ・データベース・メンバーが動的に計算されることを考慮してください。動的に計算されるメンバーには、クエリーされるまでは#MISSING の値が含まれます。

集約ストレージ・データベースの式構文

集約ストレージ・アウトラインのメンバー式を作成する際は、次の規則に従ってください:

- メンバー名が次のいずれかの条件に該当する場合は、そのメンバー名を大カッコ([])で囲みます:
  - 数字で始まるかスペースが含まれている場合(たとえば、[100])。明確にしてコードを読みやすくするため、すべてのメンバー名で大カッコを使用することをお薦めします。
  - 演算子または関数名と同じ場合。演算子と関数のリストについては、『Oracle Essbase テクニカル・リファレンス』を参照してください。
  - 英数字以外の文字が含まれている場合。たとえば、ハイフン、(-)、アスタリスク(*)またはスラッシュ(/)。

  注: 式では、$または&で始まるメンバー名を引用符と大カッコで囲む必要があります。たとえば、$testmember は式では["$testmember"]/100 と表されます
- 単一の else 条件を持つ条件付テストを記述するには、IIF 関数を使用します。IIF 関数の構文では、else 条件を識別するための ELSEIF キーワードも、ステートメントを終了するための ENDIF キーワードも必要ありません。IIF 関数をネストすることで、より複雑な式を作成できます。
- 複数の条件を持つ条件付テストを作成するには、CASE、WHEN、THEN 構造体を使用します。
- タプルは必ず正確に指定します。タプルとは、同じメンバーからの 2 つのメンバーは存在できないという制限を持つメンバーの集合のことです。タプルは、たとえば(Actual, Sales)のように、カッコで囲みます。
- セットは必ず正確に指定します。セットとは、1 つ以上のタプルの順序付けられた集合のことです。セットに複数のタプルが含まれる場合は、次のルールが適用されます: セットの各タプルのメンバーは、セットの他のタプルのメンバーと同じ次元を表す必要があります。さらに、次元は同じ順序で表す必要があります。つまり、セットのすべてのタプルは同じ次元性を持つ必要があります。

  596 ページの「セットの指定のルール」を参照してください。

  次のように、セットを大カッコで囲みます:

  ```
  { [Year].[Qtr1], [Year].[Qtr2], [Year].[Qtr3], [Year].[Qtr4] }
  ```

集約ストレージ・アウトラインでの式の作成

式を作成するには、式エディタを使用します。式エディタは、アウトライン・エディタの「メンバーのプロパティ」ダイアログ・ボックスのタブの 1 つです。式はプレーン・テキストです。式を式のテキスト領域に直接入力することも、事前定義された式テンプレートを使用することも、選択したテキスト・エディタで式を作成してから、それを式エディタに貼り付けることもできます。

次元構築データ・ソースに式を含めることもできます。293 ページの「フィールド・タイプ情報の設定」を参照してください。

➤ 式を作成する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「集約ストレージ・データベースの式の作成」を参照してください。

式構文の検査

Essbase には、式の構文エラーを通知する、MDX ベースの構文検査機能が組み込まれています。たとえば、関数名を誤って入力したり、存在しないメンバーを指定した場合に Essbase によって通知されます。不明な名前は関数名のリストで検証できます。Essbase サーバーまたはアウトラインに関連付けられたアプリケーションに接続していない場合は、不明な名前を検証するために Essbase により接続が行われる場合があります。

構文チェックは、式を保存するときに行われます。エラーは「メッセージ」パネルに表示されます。エラーが発生した場合は、式を保存するかしないかを選択します。エラーのある式を保存すると、アウトラインの確認または保存時に警告が出されます。エラーのある式を計算すると、その式は無視され、メンバーには $MISSING の値が指定されます。

構文チェッカでは、式のセマンティク・エラーは通知できません。セマンティク・エラーは、式が期待どおりに動作しないときに発生します。式のセマンティク・エラーを検出する方法の 1 つは、クエリーに式を定義する数値式を配置し、そのクエリーを実行して、結果が期待どおりであることを確認することです。クエリーに式を配置する方法は、1028 ページの「MDX 式の使用」を参照してください。

クエリーを作成するには、MDX スクリプト・エディタを使用できます。MDX スクリプト・エディタには、MDX 構文のカラー・コーディングやオートコンプリートなどの機能があります。Oracle Essbase Administration Services Online Help の「MDX スクリプト・エディタについて」を参照してください。

式の表示

➤ 式を表示するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	集約ストレージ・データベースの式の作成	Oracle Essbase Administration Services Online Help
MaxL	**query database**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	GETMBRCALC	『Oracle Essbase テクニカル・リファレンス』

集約ストレージ・アウトラインでの式の作成

次の項では、集約ストレージ・アウトラインのメンバーに対する様々な式の記述方法について説明し、いくつかの例を示します。

集約ストレージ・アウトラインの基本的な等式

式で算術演算を使用して、基本的な等式を作成できます。たとえば、次の式は、ASOsamp.Sample データベースの「Avg Units/Transaction」メンバーに適用されます:

```
[Units]/[Transactions]
```

「Avg Units/Transaction」内の式は、ユニットの数をトランザクション数で割って、トランザクション当たりのユニット数の平均を求めます。

集約ストレージ・アウトラインでは、メンバーに支出アイテムのタグは付けられません。したがって、Calc の@VAR や@VARPER などの関数は、支出タグを考慮することで 2 つのメンバー間の差異を決定しますが、集約ストレージ・アウトラインでは関係がありません。

MDX の減算演算子は、2 つのメンバー間の差異を計算するために使用できます。たとえば、次の式を ASOsamp.Sample の「Price Diff」という新しいメンバーに適用して、支払価格と元値の差異を計算できます:

```
[Price Paid]-[Original Price]
```

集約ストレージ・アウトラインでの次元をまたいだメンバー

ASOsamp.Sample には、メンバーに対する「合計に占める割合%」という式が含まれています。このメンバー式では、「Transactions」によって生成されるメジャー合計のパーセンテージが識別されます。「合計に占める割合%」の式は次のとおりです:

```
Transactions/
(Transactions,Years,Months,[Transaction Type],[Payment Type],
Promotions,Age,[Income Level],Products,Stores,Geography)
```

この式では、タプル(「Transactions, Years, ...」)で除算したメンバー(「Transactions」)を指定します。この式では、すべての次元の最上位のメンバーをリストして、キューブ内のすべての「Transaction」データを計上しています。つまり、「Curr Year」メンバーの「Transaction」データではなく「Years」次元のすべてのメンバーの「Transaction」データ、最初の 2 つの四半期の中での月に対する「Transaction」データではなくすべての月に対する「Transaction」データ、などと続きます。このようにして、「合計に占める割合%」の値は、「Transactions」によって生成されるメジャーの合計のパーセンテージを表します。

集約ストレージ・アウトラインの式の条件付テスト

1 つまたは一連の条件付テストを使用する式を定義して、メンバーの値を特定できます。1 つの else 条件を持つテストを実行するには、IIF 関数を使用します。IIF 関数をネストすれば、より複雑なクエリーを作成できます。

次の例では、会社がクレジット・カード取引に対して支払う必要がある価格(5%の手数料を含む)を表すメンバーの式を指定しています。次の例では、ASOsamp.Sample データベースのメジャー次元に「Credit Price」メンバーが追加されていると仮定しています。「Credit Price」には次の式が含まれ、支払タイプがクレジット・カードの場合、これにより「Price Paid」に 5%加算されます。

```
   IIF (
 [Payment Type].CurrentMember=[Credit Card],
 [Price Paid] * 1.05, [Price Paid]
)
```

複数のテストおよび else 条件が存在する式を作成するには、CASE、WHEN、THEN 構造体を使用します。

Filter 関数では、指定された検索条件の基準を満たす入力セットのタプルが戻されます。たとえば、すべての製品に基準線(100)を設定する場合は、基準線メンバーを追加し、そのメンバーに対する式を次のように作成します:

```
   Count(Filter(Descendants([PersonalElectronics],
[Products].Levels(0)),[Qtr1] > 100.00))
```

集約ストレージ・アウトラインの式での UDA の指定

UDA は、メンバーに対して作成する単語または語句です。たとえば、Sample.Basic の場合、「Market」次元の最上位レベルのメンバーには、「Small Market」という UDA、または「Major Market」という UDA が含まれます。

このトピックで使用する「Major Market」の例は、すべての主要市場メンバーの売上高合計を示すメンバーに対する式の作成方法を示しています。この例では、Sample.Basic に新しいメンバー(「Major Market Total」)が追加されていると仮定しています。

1. MDX には、IsUDA というブール関数があります。この関数では、メンバーに関連する UDA タグがある場合、TRUE が戻されます。次の構文では、「Market」次元の現在のメンバーに UDA の「Major Market」がある場合、TRUE が戻されます:

   ```
   IsUda([Market].CurrentMember, "Major Market")
   ```

2. 次の構文で示すように IsUDA とともに Filter 関数を使用すると、「Market」次元のメンバーを循環して、「Major Market」という UDA を持つ各メンバーの値を戻します:

   ```
   Filter([Market].Members, IsUda([Market].CurrentMember, "Major Market"))
   ```

3. Sum 関数では、Filter 関数で戻された値が加算されます。「Major Market」の例では次の数式が作成されています:

```
        Sum (Filter([Market].Members, IsUda([Market].CurrentMember, "Major
Market")))
```

この式が、「Major Market Total」メンバーに添付されます。

透過パーティションの使用による集約ストレージ・データベースへのライトバックの使用可能化

ライトバック・パーティションでは、ソース集約ストレージ・データベース内のデータを変更しないまま、ターゲット・ブロック・ストレージ・データベース内のデータをオンザフライで更新できます。ライトバック・パーティションを作成することで、計算時間が削減され、データベース・サイズが縮小される可能性があります。

ライトバック・パーティションを作成するときは、次のガイドラインに従ってください:

- 集約ストレージ・データベースが作成されているのとは別のアプリケーションにブロック・ストレージ・データベースを作成します。

  通常、ブロック・ストレージ・データベースには、集約ストレージ・データベースの次元のサブセットが含まれています。

- データの格納先にする場所に基づいて、透過パーティションを作成します。ブロック・ストレージ・データベースをターゲット、集約ストレージ・データベースをソースにします。

  第 15 章「パーティション・アプリケーションの設計」を参照してください。

  注： ある期間のデータが集約ストレージ・データベースに格納されていて、別の期間のデータがブロック・ストレージ・データベースに格納されている場合は、時間次元でパーティション化を行うこともできます。たとえば、1 月から 3 月までの実績データがあり、そのデータが集約ストレージ・データベースに格納されている場合に、ブロック・ストレージ・データベース内のライトバック・メンバーを使用してその年の最後の 9 か月の予算を設定するとします。

ユーザーは、ブロック・ストレージ・データベースに対するクエリーとライトバックを行います。クエリーは、ブロック・ストレージ・データベースによって、または透過的に集約ストレージ・データベースによって処理されます。

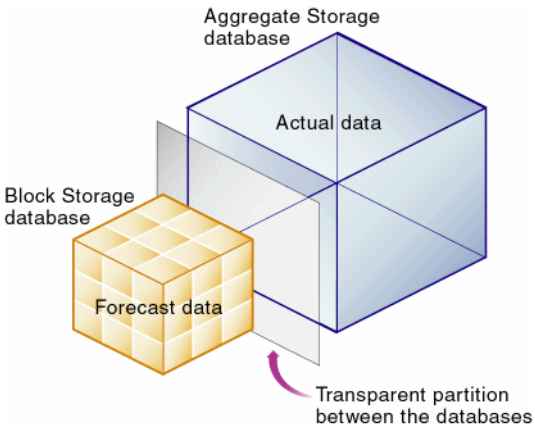
➤ 集約ストレージおよびブロック・ストレージのライトバック・パーティションを作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	集約ストレージ・パーティション・ウィザード パーティションの作成	Oracle Essbase Administration Services Online Help
MaxL	**create database** **create partition**	『Oracle Essbase テクニカル・リファレンス』

パーティション・アプリケーションの作成には、データベース・マネージャ権限が必要です。

図 158 は、予測データと実績データの差異を分析するための透過パーティションの使用を示しています:

図 158　予測データと実績データの差異を分析するために使用される透過パーティション



次の手順では、集約ストレージのサンプル・データベース(ASOsamp.Sample)を基にしていて、Administration Services の「集約ストレージ・パーティション・ウィザード」を使用します(Oracle Essbase Administration Services Online Help を参照)。

➤ ライトバック・パーティションを作成するには:

1 **ASOsamp.Sample データベースを選択します。このデータベースには、現在の年と以前の年の実績データが含まれています。**

図 159 を参照してください。

図 159　ASOsamp.Sample 集約ストレージ・データベースのアウトライン

2　ASOsamp.Sample の次元のサブセットであるメジャー、年、時間、製品、店舗、地域次元を含むブロック・ストレージ・データベースを作成します。

3　年次元を編集し、次のメンバーをブロック・ストレージ・データベース・アウトラインに追加します:

- 「Next Year」というメンバー。予測データはこのメンバーに入れられます。
- 「Forecast Variance」というメンバー。このメンバーに式を追加して、実績データと予測データの差異を計算します。

図 160　実績データと予測データの差異を計算する年次元のメンバーを示すブロック・ストレージ・データベース・アウトライン

4　次のメンバー式を削除します:

- メジャー次元の、「Avg Units/Transaction」および「合計に占める割合%」のメンバー式
- 年次元の、「Variance」および「Variance%」のメンバー式
- 時間次元の、QTD および YTD の下のメンバー式

これらの式は、集約ストレージ・データベースからコピーされる、MDX で記述される式です。MDX の数式はブロック・ストレージ・データベースでは解釈されません。

5 **データベースを、ターゲットがブロック・ストレージ・データベース(予測データ)、ソースが集約ストレージ・データベース(実績データ)として、年次元の透過パーティションとリンクさせます。**

ライトバック・メンバー(「Forecast」および「Variance」)はパーティション領域に含めないでください。

注： Administration Services の「集約ストレージ・パーティション・ウィザード」を使用している場合、この手順は自動的に実行されます。手順 3 で年次元を選択したため、データベースは自動的に年次元でパーティション化されます。ライトバック・メンバーは、パーティション領域に含まれません。

ブロック・ストレージ・データベースのライトバック・メンバーに予測値を入力します。追加されたメンバーはパーティション化された領域の外側にあるため、それらに記述してから、更新済データに基づいてデータを計算してレポートを生成できます。透過パーティションにより、両方のデータベースをシームレスに表示できます。

62 集約ストレージの時間ベースの分析

この章の内容

この章の情報は、集約ストレージ・データベースのみに適用され、ブロック・ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

はじめに

ほとんどのデータ分析には、時間(履歴)という観点が含まれています。集約ストレージ・アプリケーションには、時間次元と日時次元による組込みの時間ベースの機能があります。

集約ストレージの時間次元は、ブロック・ストレージの時間次元と同じです。これらの次元では、タイム・バランス計算などのために、勘定科目に期首、期末、平均および累計の値としてタグを付けることができます。時間次元は簡単に変更できます。

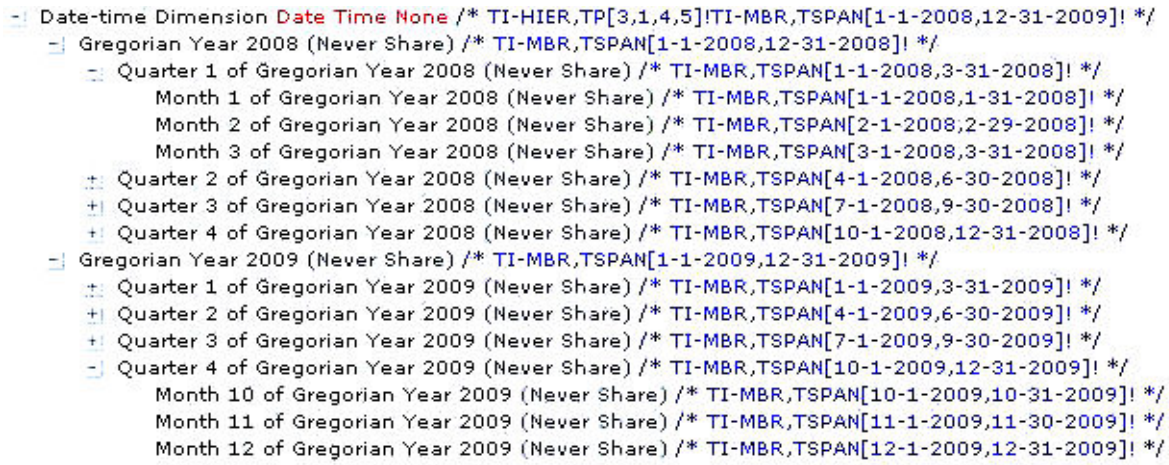
集約ストレージの日時次元により、期首、期末および平均のタイム・バランスのサポートが可能になります。さらに、Administration Services には、特殊なカレンダに基づいたすばやい日時次元の設定に加え、時間関連のデータのクロス集計レポートを可能にするリンク属性次元の設定のための、強力な日時次元の作成ウィザードがあります。日時次元の構造は堅牢で、一度作成した後は簡単には変更できません。

この章の残りの部分では、日時次元と、その次元に関連付けることができるリンク属性次元について説明します。集約ストレージ・アウトラインのタイム・バランス計算の詳細は、1090 ページの「集約ストレージの会計次元のタイム・バランスおよびフロー・メトリック計算の実行」を参照してください。

日時次元の理解

日時次元には、いくつかの標準的な企業カレンダの 1 つの選択に基づいた日付階層が含まれます。次元プロパティとメンバー・プロパティには、図 161 に示すように、Essbase サーバーで階層の管理に使用されるテキスト文字列が含まれます。

図 161　サンプルの日時階層



この階層の最上部の次元メンバーには、次のコメントが示されています:

```
/* TI-HIER,TP[3,1,4,5]!TI-MBR,TSPAN[1-1-2008,12-31-2009]! */.
```

感嘆符(!)で囲まれた TI-MBR,TSPAN[date1, date2]の情報では、次元がカバーする時間範囲(この例の場合は、2008 年 1 月 1 日から 2009 年 12 月 31 日までの 2 年間)を定義します。時間スパンの前に配置されている配列 TI-HIER,TP[3,1,4,5]は、階層を示しています。配列の最初の数字である 3 は、階層が 3 つの期間に含まれていることを示しています。配列の残りの数字は、次のような指定可能なレベルに基づいて、どの期間が含まれているかを示しています:

- 1- 年
- 2- 半期
- 3- 三半期
- 4- 四半期
- 5- 月
- 6- 期間
- 7- 週
- 8- 日

配列 TI-HIER,TP[3,1,4,5]は、配列に年、四半期および月という 3 つの期間が含まれていることを示しています。

同じ時間スパン TI-MBR,TSPAN 構文を使用して、各メンバーのコメント・フィールドで、カバーされる時間範囲を定義します。たとえば、メンバー「Quarter 1 of

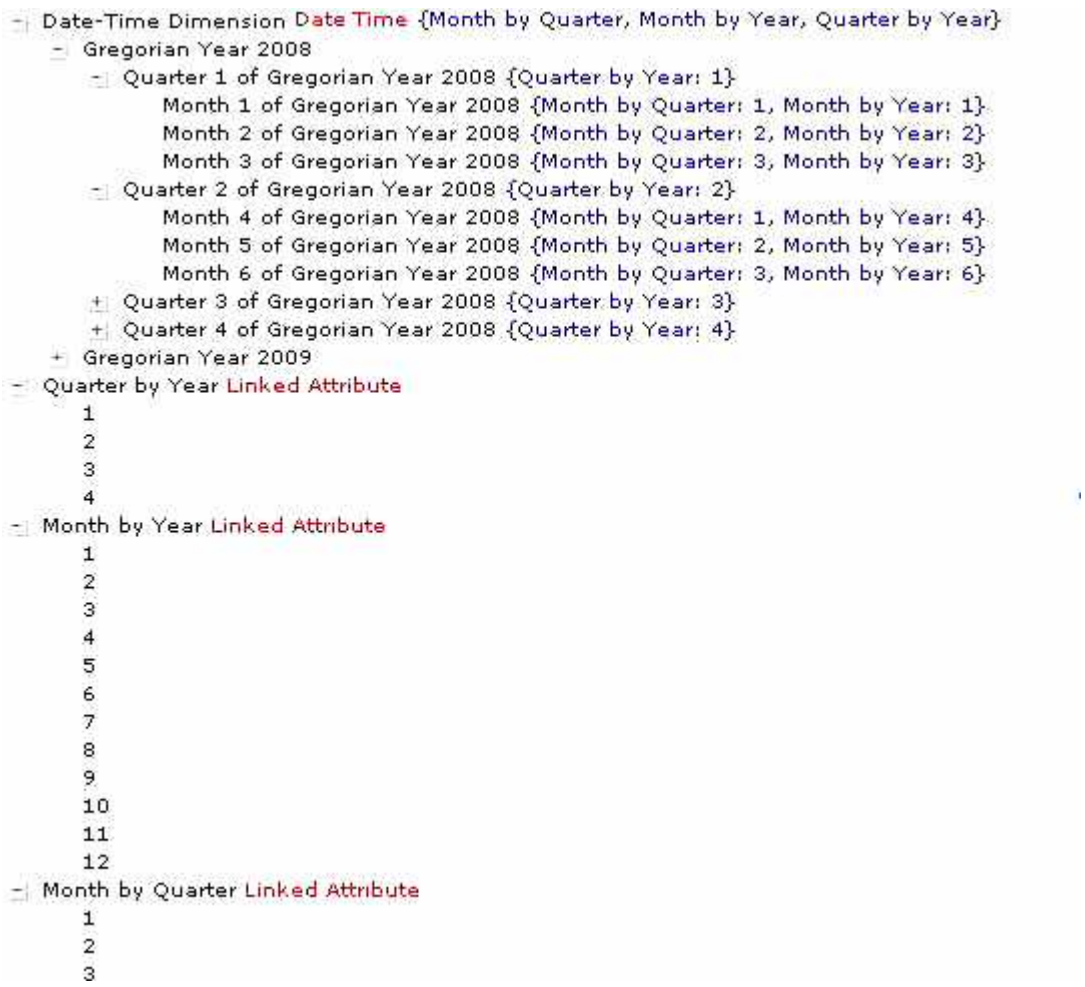
Gregorian Year 2008」は 2008 年 1 月 1 日から 2008 年 3 月 31 日までをカバーし、メンバー「Quarter 2 of Gregorian Year 2008」は、2008 年 4 月 1 日から 2008 年 6 月 30 日までをカバーします。

リンク属性の理解

複数の期間内での相対的な位置という点でのデータの分析が、有用な場合があります。たとえば、あるコストが各四半期の最後の週に増加することを知ることは、支出のバランスをとるために役立ちます。リンク属性により、四半期の最初の月や月の 4 週目などの相対期間に基づいてデータを分析できます。

リンク属性次元は、日時次元のみと関連付けることができます。リンク属性次元の定義は、日時次元の作成ウィザードの一部分です。ユーザーが作成するリンク属性次元は、日時次元のカレンダを定義するときに選択する時間レベルに基づきます。たとえば、図 162 に示すように、年、四半期および月の時間レベルを選択してリンク属性を作成する場合、このウィザードでは、「Quarter by Year」、「Month by Year」および「Month by Quarter」という 3 つのリンク属性次元(デフォルトの次元名)が作成されます。各属性次元には、連続する数値名を持つメンバーが含まれます。この数値名は、該当の属性と関連付けられているメンバーを分析する際に使用される相対位置を示します。

図 162　リンク属性を持つ日時次元のサンプル



リンク属性により、次に示すような単純なクロス集計分析で、四半期の最初の月の売上高を、四半期の 2 番目と 3 番目の月の売上高と比較できます:

	100-10	100-10	100-10
	Sales	Sales	Sales
	Month by Quarter_1	Month by Quarter_2	Month by Quarter_3
Gregorian Year 2007	11326.72	13544.89	24893.33
Gregorian Year 2008	21245.18	24977.32	29786.93

注： リンク属性は、年の時間レベルのメンバーには割り当てられません。

リンク属性次元は、2 つの時間コンポーネントを反映しています。これは、「Quarter by Year」などのデフォルト名を使用する場合に簡単に理解できます。たとえば、「Quarter by Year」次元のメンバーは、図 162 の「Quarter 1 of Gregorian Year 2008」などの四半期の時間レベルのメンバーと関連付けられます。最初の時間コンポーネントである「Quarter」は、属性が関連付けられる日時次元メンバーを示すため、関連付けレベルと呼ばれます。

この例の 2 つ目の時間コンポーネントである「Year」は、添付レベルと呼ばれます。添付レベルは、関連付けられた日時次元内のメンバー・レベルを示します。リンク属性次元の各メンバーには、関連レベルと添付レベルの間の関係を示す数字が含まれています。「Quarter by Year: 1」は最初の属性メンバー、「Quarter by Year: 2」は 2 番目のメンバー、というように続きます。

リンク属性「Quarter by Year: 1」と関連付けられている日時次元メンバーは、それぞれの年の最初の四半期です。属性「Month by Quarter: 1」と関連付けられているすべての日時次元メンバーは、それぞれの四半期の最初の月です。そのため、リンク属性により、1 番目、2 番目、3 番目などの一般的な期間の関係に基づいた情報の分析が可能になります。

日時分析のためのアウトラインの設計と作成

Administration Services の「日時次元の作成」ウィザードでは、カレンダ・テンプレートによる日時次元を含む時間分析構造の設計に、オプション選択を使用します。時間分析のニーズに合うように、カレンダ・タイプを選択し、それをビジネスの要件に合わせてカスタマイズして、その他のオプションを選択します。このウィザードでは、次のアウトライン・コンポーネントを作成します:

- 日時次元。

  1040 ページの「日時次元の理解」を参照してください。

- (オプション)リンク属性次元とメンバー関連付け。

  1041 ページの「リンク属性の理解」を参照してください。

- (オプション)日レベルのメンバーと関連付けられた属性次元。これらは、特定の日付または曜日に基づく標準のブールまたはテキスト属性です。これらの属性を使用して、日レベルの情報を含めたり除外したりできます。たとえば、このウィザードでは次のような状況を定義できます:

- 選択した曜日に「週末」などのブール属性次元を割り当てることで、結果としてすべての曜日[週末].[True]または[週末}.[False}属性を持たせる。
- 指定した特別の日や休日のための属性を割り当てる。
- 適切な平日に「月曜日」や「火曜日」などのテキスト属性を割り当てる。

Administration Services の日時次元の作成ウィザードでは、指定されたカレンダと時間範囲のすべてのメンバーを構築します。

日時次元の作成の準備

日時次元の作成ウィザードを実行する前に、次の点を考慮してください:

- データベースのモデルにする必要があるのはどの期間ですか?開始日と終了日を指定する必要があります。
- ビジネスでの使用に適しているのは、どのタイプのカレンダですか?

  1043 ページの「日時カレンダの理解」を参照してください。
- データベースがそのアウトライン内で反映する必要がある期間の粒度は、どれですか?ビジネス分析は週レベルまでドリル・ダウンしますか、または、レポート作成が月単位かそれ以上のレベルのため、週を省略できますか?すべてのビジネス・メジャーを日レベルまで分析およびレポートしますか?半期または三半期の分析結果は必須ですか?
- 時間関係の中でのデータの分析は有益ですか?たとえば、四半期を含むビジネス・カレンダ内で、売上高が各四半期の期末に向かうほど増加するかどうかなどの各第 4 四半期のトレンドを表示する機能は、有用ですか?

日時次元の作成ウィザードは、多数のオプションがある強力なツールです。このウィザードで構築される次元について理解を深めるため、何度か試験的に実行することを検討してください。また、Oracle Essbase Administration Services Online Help のこのウィザードの各ページのトピックも参照してください。

日時カレンダの理解

日時次元の作成ウィザードのカレンダ・テンプレートでは、日時タイプの次元内のメンバーの階層構造を決定します。すべてのカレンダに対して、日時次元名、カレンダ・タイプ、週の最初の曜日、および日時次元メンバーを構築する開始日と終了日を選択する必要があります。また、カレンダ・メンバー階層を決定する時間レベルも選択する必要があります。すべてのカレンダで年の時間レベルが必須です。その他の時間レベルの可用性は、選択したカレンダ・タイプとその他の時間レベルによって変わります。たとえば、半期と三半期の時間レベルは相互に排他的です。

時間レベルを選択すると、ウィザードによって、それぞれの年の中でその時間レベルの各インスタンスのメンバーが作成されます。たとえば、月の時間レベルを選択すると、各年の下の階層内に 12 個の月メンバーが作成されます。命名規則には各メンバーの命名パターンが示されていて、これにはその年の祖先の中での相対位置を示す数字が含まれます。たとえば、図 162 に示されているグレゴリオ暦

カレンダでは、「Month 4 of Gregorian Year 2008」などの月番号を含むデフォルトの命名パターンの後に月メンバーが続きます。

注： 時間次元の命名規則を選択するときには、重複メンバーが作成される可能性があります。これにより、アウトラインがメンバー名の重複を許可するよう設定されていないかぎり、日時次元の作成手順が失敗することがあります。アウトラインがメンバー名の重複を許可するよう設定されていない場合は、重複を避ける必要があります。

カレンダ・テンプレートと選択した時間レベルによっては、期間の開始日または終了日、週の数、または月や週がその親メンバーでどのようにグループ化されているかなどの、年、月または期間の特徴(セマンティク・ルール)の定義が必要な場合があります。これらの特徴は、メンバーの構築と命名のルールに影響を与えます。一部のカレンダでは、年、期間または月のセマンティク・ルールの可用性は、選択したその他の年、期間または月のオプションに依存します。Oracle Essbase Administration Services Online Help で日時次元の作成ウィザード - カレンダ階層の選択パネルに関する項を参照してください。

日時次元の作成ウィザードには、次のようなカレンダ・タイプが含まれています:

グレゴリオ暦

グレゴリオ暦カレンダは、1 月 1 日から 12 月 31 日の、標準の 12 か月のカレンダです。時間レベルは、年、半期、三半期、四半期、月、週および日です。

会計

会計カレンダの定義は、企業の財務報告の要件に基づいて、任意の日付で開始できます。「Weeks In」財務カレンダには、7 日間が含まれます。12 か月のレポート期間には、4 週の月が 2 か月と 5 週の月が 1 か月が含まれ、3 か月のパターンが四半期ごとに繰り返されます(4-4-5、4-5-4 または 5-4-4 週)。その年に 53 週ある場合、1 か月に追加の週がある場合があります。

週の定義によって、カレンダ年を週に分割する方法が決まります。週の定義を調整して、52 週または 53 週の年にすることができます。時間レベルは、年、半期、三半期、四半期、月、週および日です。

リテール

National Retail Federation のリテール・カレンダは、年をまたがった前週比データを分析するようにモデル化されています。このカレンダには、5 年から 6 年ごとにうるう年の週がある、4-5-4 の四半期のパターンがあります。開始日は年ごとに異なりますが、常に 2 月上旬に期限が切れます。前年同期比を比較するときの標準的な方法は、53 週の年の最初の週を除外して追加の週の標準化を行い、一方でそれらの年で同じ休日を維持することです。使用可能な時間レベルは、年、半期、四半期、月、週および日です。

製造

製造カレンダでは、7 日の週で構成される 13 期間の年を定義します。製造期間は、それぞれ 3 期間の 3 つの四半期と、4 期間の 1 つの四半期に分割されます。53 週の年を除き、すべての期間に 4 週が含まれます。53 週の年の場合、1 つの期間には 5 週が含まれます。

13 期間を定義するときは、どの四半期に追加の期間があるかを指定します。その年に 53 週ある場合は、どの期間に追加の週があるかを指定する必要があります。その年が特定の日付で始まると指定する場合は、その年が 52 週か 53 週かを指定する必要があります。52 週の場合は、週を最初に 53 週まで減らし、その後 52 週に減らすため、52 週と 53 週の両方のオプションの指定が必要な場合があります。使用可能な時間レベルは、年、半期、四半期、期間、週および日です。

ISO 8601

ISO 8601 カレンダには、7 日の週が含まれます。1 年は 1 月 1 日の前でも後でも開始でき、ISO カレンダの最初の週に 1 月の最初の木曜日が含まれるなどのように、日で始まるようモデル化されています。週の最初の曜日は月曜日です。ISO カレンダでは年、週および日の時間レベルが必須で、追加の時間レベル・オプションはありません。

日時次元の変更または削除

日時次元を変更する唯一の方法は、手動による変更です。その密な構造のため、日時次元の変更には危険が伴う可能性があります。大抵の場合、変更にメンバーの追加または削除が伴う場合は特に、次元を削除してから、日時次元の作成ウィザードを使用して、このウィザードによって構築された変更で次元を再作成することをお薦めします。

軽微な変更の場合でも、1040 ページの「日時次元の理解」で説明しているコメント構造に注意する必要があります。TI-HIER および TI-MBR,TSPAN セクションは、コメント・フィールドの最初に指定する必要があります。次のように、コメントは最後の!マークの後に追加できます:

```
TI-MBR,TSPAN[1-1-2006,1-31-2006]! This is the First Month
```

注： メンバーのコメント内の TI-HIER および TI-MBR,TSPAN セクションを編集する場合は、階層全体を通して確実に正しいメンバーが存在するようにする必要があり、祖先または子孫の依存範囲を編集する必要があります。たとえば、指定された月の範囲を変更する場合は、四半期の範囲も変更する必要があります。

日時次元を削除する場合は、関連するすべての属性次元も削除してください。リンク属性が意味を持つのは、それらが完全である場合のみです。つまり、すべてのメンバーが存在し、正しい順序で関連付けられている場合です。ほんのわずか

のメンバーからリンク属性の関連付けを解除しても、アウトラインは無効になります。

次の項で要約する確認ルールに注意してください。

日時次元の確認ルール

- この次元には、日時のタグが付けられています。
- この次元は、単一階層の、保管される次元です。
- この階層はバランスがとられています。レベル 0 のすべてのメンバーの世代は同じです。
- メンバーはすべて、加算(+)集計でタグ付けされます。
- 各メンバーのメンバー・コメント・フィールドには、コメントの一番左に次の情報が含まれています(詳細は、1040 ページの「日時次元の理解」を参照)。
    - 最上位の次元メンバーのメンバー・コメント・フィールドには、階層構造を記述する配列を含む TI-HIER, TP[ ]指定が含まれます。
    - 上位次元メンバーを含む各メンバーのメンバー・コメント・フィールドには、メンバーの日付範囲(開始日から終了日まで)が含まれる TI-MBR, TSPAN[ ]指定が含まれます。
- 世代に従って、日付範囲は連続している必要があり、昇順で増加する必要があります。たとえば、月世代では、各月メンバーの日付範囲には、前の期間の終了日から論理的に続く開始日が含まれている必要があります。時間の空白はないようにする必要があり、親メンバーはその子の合計時間スパンと等しい必要があります。

リンク属性次元の確認ルール

- リンク属性次元には、リンク属性のタグが付けられています。
- リンク属性次元は、日時次元と関連付けられています。
- 各リンク属性次元には、親レベルと子属性メンバーという、2 つのレベルのみがあります。
- リンク属性のメンバー名は、1 から始まる連続番号です。
- 属性次元が存在する場合は、日時次元の添付レベルと関連レベルのすべての組合せが存在する必要があります。たとえば、日時次元に年、半期および月の時間レベルが含まれている場合に、リンク属性次元を作成する場合は、「Month by Year」、「Month by Semester」および「Semester by Year」を作成する必要があります。

日付にマップされたデータのロード

日時次元のレベル 0 のメンバーにデータをロードするときは、メンバー名のかわりに日付文字列を使用できます。日付階層が日の粒度レベルにスパンしていない場合でも、データ・ソースは個々の日付によって指定できます。ロード・プロセスでは、値が集約され、その値が適切なレベルで保管されます。

日付に基づいたデータのロードには、次のような利点があります:

- 日付は、データ・ファイルの日付フォーマットがサポートされていれば、日レベルのロード・ファイルからロードできます。
- データ・ロードを「既存のセルに追加」するよう設定すると、日レベルのロード・ファイルから階層内の粒度の低いレベルに(たとえば、週または月レベル 0 のセルに)データをロードできます。

表 201 に、データ・ロード・ルール・ファイルの日付フィールドを定義するときに使用できる日付フォーマットの文字列をリストします:

表 201　日付フォーマット文字列

日付フォーマット文字列	例
mon dd yyyy	Jan 15 2006
Mon dd yyyy	January 15 2006
mm/dd/yy	01/15/06
mm/dd/yyyy	01/15/2006
yy.mm.dd	06.01.06
dd/mm/yy	15/01/06
dd.mm.yy	15.01.06
dd-mm-yy	15-01-06
dd Mon yy	15 January 06
dd mon yy	15 Jan 06
Mon dd yy	January 15 06
mon dd yy	Jan 15 06
mm-dd-yy	01-15-06
yy/mm/dd	06/01/15
yymmdd	060115
dd Mon yyyy	15 January 2006
dd mon yyyy	15 Jan 2006
yyyy-mm-dd	2006-01-15
yyyy/mm/dd	2006/01/15
ロング名	Sunday, January 15, 2006
ショート名	1/8/06 (m/d/yy)

注： 内部フォーマット文字列に含まれない余分な空白を使用すると、エラーが戻されます。日付フォーマットに一致すると、その後の文字は無視されます。日付フォーマットが mm/dd/yy の場合に日付文字列を 06/20/2006 と誤って使用すると、末尾の 06 が無視され、日付は 2020 年 6 月 20 日と解釈されます。ロング名フォーマットは、指定された日付に対する曜日の一致については確認されません。

時間ベースのデータの分析

MDX と Smart View のインタフェースでは、データ内の期間の関係を分析するために、リンク属性を活用します。

Smart View の使用による時間関連データの分析

Smart View では、リンク属性を次のようないくつかの方法で使用できます:

- フリーフォーム・グリッドでの操作。選択されたグリッド上の日時メンバーで「メンバー選択」を使用すると、次元タイプとして日時が表示され、その次元のメンバーがリストされます。「期間」フィルタと「範囲」フィルタを使用して、リンク属性を使用する情報を選択および表示できます。
- クエリー・デザイナの使用。
  - クエリー・デザイナのツールバーの日時次元を選択すると、属性リストにリンク属性が組み込まれます。
  - POV ツールバーを使用して、リンク属性を選択し、それをフリーフォーム・グループにドラッグして、クエリーを実行できます。

追加情報は、Oracle Hyperion Smart View for Office User's Guide を参照してください。

MDX による時間ベースのメトリックの分析

表 202 で、日付階層の分析用に用意されている MDX 関数をリストします。これらの関数の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

表 202　日付階層の分析用の MDX 関数

関数	説明
DateDiff	入力された 2 つの日付間の差異が戻されます。
DatePart	日付部分(Year/Quarter/Month/Day/DayOfYear/Weekday)が数値として戻されます。
DateRoll	指定された日付に、特定の時間間隔の数が更新(加算または減算)されて、別の日付が戻されます。
DateToMember	入力日付で指定された日付階層メンバーが戻されます。
FormatDate	フォーマットされた日付文字列が戻されます。

関数	説明
GetFirstDate	日付階層メンバーの開始日が戻されます。
GetLastDate	日付階層メンバーの終了日が戻されます。
Today	Essbase コンピュータの現在の日付を表す数値が戻されます。
ToDateEx	日付文字列が、計算で使用できる日付に変換されます。

63 集約ストレージ・データのロード、計算および取得

この章の内容

この章の情報は、集約ストレージ・データベースのみに適用され、ブロック・ストレージ・データベースとは関係がありません。第 60 章「集約ストレージとブロック・ストレージの比較」も参照してください。

はじめに

データベース情報の処理に関する最も一般的なプロセスには、アウトラインの管理、データベースへのデータ値のロード、値の計算およびデータベース情報の取得が含まれます。集約ストレージ・データベースでのこれらのタスクの実行は、ブロック・ストレージ・データベースでの実行とは異なります。

この章の例は、図 163 のアウトラインを引用しています。

図 163　集約ストレージ・アウトラインのサンプル

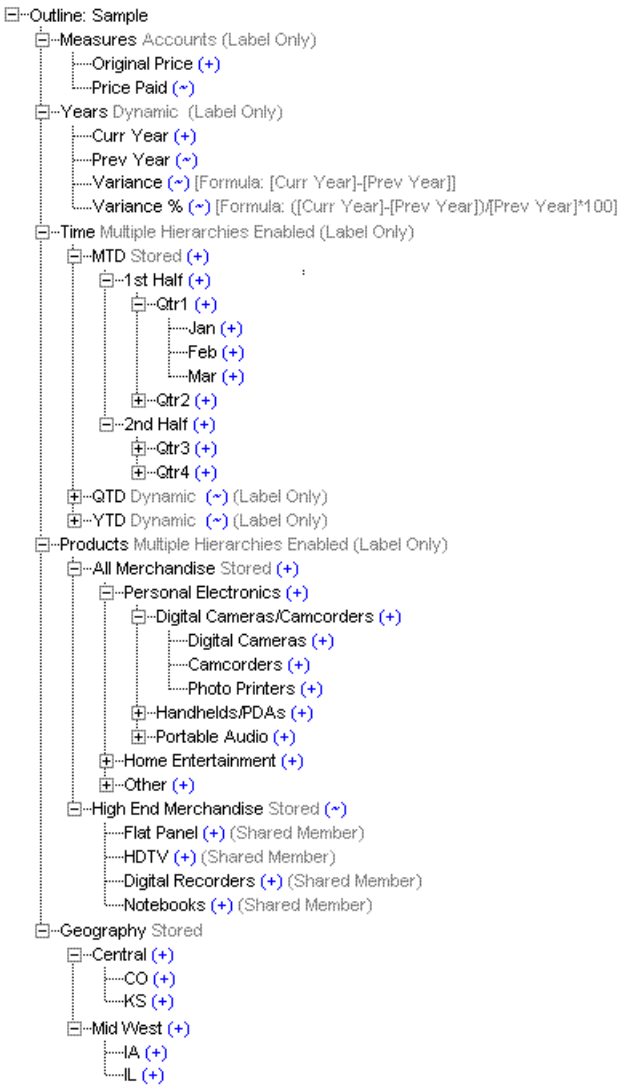


図 163 の簡易化した集約ストレージ・アウトラインは、完全には展開されていません。メンバー名の左側にあるプラス記号(+)のノードは、そのメンバーが表示されていない子を持つことを示しています。

集約ストレージ・データベースの準備

サブトピック

- 集約ストレージ・データベースでの次元構築
- 集約ストレージ・データベースでの排他操作について
- 集約ストレージ・データベースへのデータのロード
- 集約ストレージ・データベースからのデータの消去
- 集約ストレージ・アプリケーションのコピー
- データ・ロードと次元構築の結合

この項のトピックでは、集約ストレージ・データベースとブロック・ストレージ・データベース間の、次元構築およびデータ・ロード・プロセスに関する相違点に

ついて説明します。データ・ロード、次元構築およびルール・ファイルの概念と手順に精通している必要があります。

データ・ソースを使用してアウトラインの変更やデータ値のロードを行う方法は、第 17 章「データ・ロードおよび次元構築の理解」および第 20 章「データ・ロードまたは次元構築の実行およびデバッグ」を参照してください。

構築可能な集約ストレージ・アウトラインの最大サイズの詳細は、1025 ページの「アウトラインのページング制限」を参照してください。

集約ストレージ・データベースでの次元構築

アウトラインに対する集約ストレージの次元構築の変更により、次元構築の終了時にすべての集約ビューまたはすべてのデータ値が、データベースから消去されることがあります。1131 ページの「集約ストレージ・データベースの再構築」で、アウトライン変更の結果について説明しています。

複数のデータ・ソースを使用して次元を構築する場合は、増分次元構築を実行することで処理時間を節約できます。増分次元構築により、すべてのデータ・ソースの処理が完了するまで再構築を延期させることができます。増分次元構築の詳細は、319 ページの「再構築が延期された次元構築の実行」を参照してください。

ブロック・ストレージ・アウトラインと集約ストレージ・アウトラインの間でのアウトラインの特徴の相違点は、データ・ソースとルール・ファイルに影響します。たとえば、次元を疎として定義するか密として定義するかは、集約ストレージ・アウトラインには関係がありません。

集約ストレージ次元構築のルール・ファイルに関する相違点

サブトピック

集約ストレージ・アウトラインの構築用のルール・ファイルでは、集約ストレージ・アウトラインに適用されるアウトライン・プロパティとフィールド・タイプのみを定義する必要があります。表 192 を参照してください。

ブロック・ストレージ・アウトラインを集約ストレージに変換した後、それらを集約ストレージ版のアウトラインに関連付けることで、ルール・ファイルを更新します。Oracle Essbase Administration Services Online Help の「エディタでのアウトラインの関連付け」を参照してください。

集約ストレージ次元構築のフィールド・タイプに関する相違点

解決順フィールド・タイプは、集約ストレージ・データベースのみで使用できます。

フィールドには、アウトライン内でメンバーが評価される順序を指定する数値(0-127)が含まれます。0 より小さい値、または 127 より大きい値は、それぞれ 0 と 127 にリセットされます。警告メッセージは表示されません。解決順プロパティの詳細は、1076 ページの「計算順序」を参照してください。

解決順の有効な構築方法:

- 世代
- レベル
- 親子参照

➤ フィールド情報を設定する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「フィールド・タイプの設定」を参照してください。

集約ストレージの「データ準備エディタ」ダイアログ・ボックス

集約ストレージ・データベースのルール・ファイルを編集している間、ブロック・ストレージ・データベースのみに適用される次元構築ルール・ファイル・オプションのいくつかが、「データ準備エディタ」ダイアログ・ボックスに表示されます。表 203 に、集約ストレージ・アウトラインに適用されないブロック・ストレージのルール・ファイル設定をリストします。これらのオプションのルール・ファイルのエントリは、ルール・ファイルが処理されるときは無視されます。

表 203　集約ストレージ次元構築ルール・ファイルの相違点

Administration Services インタフェースでのルール・ファイルの場所	集約ストレージ・データベースに適用されない次元構築ルール・ファイルのオプション
「次元構築の設定」ダイアログ・ボックス、「グローバル設定」タブ	「データの構成」オプション
「次元のプロパティ」ダイアログ・ボックス、「次元のプロパティ」タブ	「次元タイプ」オプション: 国 「2 パス計算」オプション 「データ・ストレージ」オプション: ● 動的計算および保管 ● 動的計算 「すべて」構成オプション
「次元のプロパティ」ダイアログ・ボックス、「会計次元」タブ	このタブのオプションは適用されません。
「フィールド・プロパティ」ダイアログ・ボックス、「次元構築プロパティ」タブ	フィールド・タイプのオプション: ● 通貨名 ● 通貨カテゴリ 通貨機能は、集約ストレージ・データベースには適用されません。

Oracle Essbase Administration Services Online Help を参照してください。

集約ストレージ次元構築のデータ・ソースに関する相違点

集約ストレージ・アウトラインを変更するためのデータ・ソースは、ブロック・ストレージ・アウトラインのみに適用されるフィールド値を含めることはできません。表 204 に、次元構築データ・ソースで集約ストレージ・データベース・アウトラインのメンバーのプロパティとして認識されるメンバー・コードをリスト

します。他のすべての集計コードは無視され、+ (加算)とみなされます。292 ページの「データ・ソースを使用したメンバー・プロパティの操作」を参照してください。

表 204　集約ストレージ・アウトラインのメンバーに有効な集約プロパティ

コード	説明
%	集計の現在の合計値のパーセンテージとして表します(動的階層のメンバーにのみ適用)
*	集計の現在の合計値で乗算します(動的階層のメンバーにのみ適用)
+	集計の現在の合計値に加算します(動的階層のメンバーにのみ適用)
-	集計の現在の合計値から減算します(動的階層のメンバーにのみ適用)
/	集計の現在の合計値で除算します(動的階層のメンバーにのみ適用)
~	集計から除外します(動的階層のメンバー、または保管階層内のラベルのみメンバーの下のメンバーのみに適用されます)
O	ラベルのみのタグ
N	データ共有不可
C	メンバーを保管階層の先頭に設定します(次元メンバーまたは世代 2 のメンバーにのみ適用)
D	メンバーを動的階層の先頭に設定します(次元メンバーまたは世代 2 のメンバーにのみ適用)
H	次元メンバーで複数階層を使用可能にします(次元メンバーのみに適用)

集約ストレージ・アウトラインでは、通貨名および通貨カテゴリのフィールド・タイプはサポートされていません。

集約ストレージ・アウトラインでは、式は MDX 数値式と同じフォーマットで指定する必要があります。1028 ページの「集約ストレージ・アウトラインでの式の作成」を参照してください。

集約ストレージ・データベースでの代替階層の構築

集約ストレージ・アウトラインの共有メンバーを構築するには、「autoconfigure」設定を Administration Services の「次元構築の設定」ダイアログ・ボックスで選択してから、Essbase で共有を許可するよう設定されていることを確認します(「共有しない」を Administration Services の「次元構築の設定」ダイアログ・ボックスで選択解除するか、「データ共有を許可しない」プロパティをルール・ファイル内で除外します)。autoconfigure が選択されていて、共有が使用可能な場合は、Essbase によって、新しい親の下に重複メンバーが自動的に共有メンバーとして作成されます。352 ページの「ルール・ファイルを使用した共有メンバーの作成」を参照してください。

注： 重複する世代(DUPGEN)の方法を使用して集約ストレージ・アウトラインに代替階層を構築する場合は、制限があります。表 195 を参照してください。

注意 集約ストレージ・データベースの代替階層では、属性をレベル-0 のメンバーとのみ関連付けできます。

集約ストレージ・データベースでの排他操作について

集約ストレージ・データベースでは、複数の排他操作を同一データベース上で同時に行うことはできません。ある排他操作の実行中に別の排他操作を同時に実行しようとすると、最初の操作の進行中は別の操作を開始できないことを示すメッセージが表示され、2 番目の操作は拒否されます。たとえば、集約ストレージ・データベースで部分データ消去の実行時に、そのデータベースで別の部分データ消去を同時に実行することは、その 2 つの操作がデータベースの別の領域を消去している場合であっても、できません。ほとんどの排他操作は相互に排他的ですが、例外がいくつかあります。

排他操作と例外:

- エクスポート

  エクスポートは読取り専用操作であるため、エクスポート操作は同時に複数実行できます。エクスポート操作は、いずれも排他操作である、構築集約およびバックアップとして実行できます; ただし、エクスポートは、他の排他操作とは互換性がありません。
- 集約の構築
- バックアップ(データベースの読取り専用アーカイブ・モードへの設定)
- データのロード(ロード・バッファのコンテンツのデータベースへのコミット)

  集約ストレージ・ロード・バッファの作成と、ロード・バッファへのデータのロードは排他操作ではありません。これらの操作は他のいかなる操作とも同時に実行できます。ただし、ロード・バッファ内のデータのデータベースへのコミットは排他操作です。
- スプレッドシート送信操作(セルの更新など)

  送信操作が実行されているときに別の排他操作を試行すると、送信操作の終了を待機してから新しい操作が続行されます。送信操作は常に小さく、時間がかからない(5 秒未満)ため、送信操作と互換性がない場合でも、新しい操作がエラーになることはありません。つまり、相互に互換性がないことが原因で操作が拒否されることがなく、スプレッドシート送信操作を同時に実行できます。

  注: 送信操作の実行中に複数の排他操作を試行した場合、送信操作の完了後に新しい排他操作が実行される順序はランダムです; 順序は、新しい排他操作が試行された順番には基づきません。たとえば、送信操作の実行中に、タイプの異なる排他操作を試行すると、送信操作の終了を待機してから新しい排他操作が続行されます。一方、別のユーザーが他の送信操作を試行した場合、それらの送信操作が後から試行されたとしても、送信操作はその他の排他操作より前に実行されます。そのため、実行を待機している送信操作が 1 つでも存在すると、排他操作が無制限に待機する可能性があります。

- スライスのマージ
- データ消去操作(完全、集約のみ、部分)

クエリーは、すべての排他操作と並行して実行できます。ただし、操作によって、データベースのデータに追加、変更または削除が行われた場合、クエリーが変更を認識すると、操作の最後に次のシーケンスが実行されます:

1. 新規クエリーが一時的に開始を阻止されます(クエリーが待機する)。
2. 既存のクエリーが実行を終了します。
3. 排他操作からのデータ変更が、データベースにコミットされます。
4. 待機しているクエリーが開始します。

クエリーは、集約ストレージ・キューブ上のデータを変更する操作が原因で拒否または取り消されることはありません。

集約ストレージ・データベースへのデータのロード

サブトピック

- データ・ロード・バッファの使用によるデータの増分ロード
- データ・ロード・バッファのリソース使用率の制御
- データ・ロード・バッファ・プロパティの設定
- セルの競合の解決
- 複数のデータ・ロードの並行での実行
- 集約ストレージ・データベースのデータ・ロード・バッファのリスト
- データ・スライスの作成
- 増分データ・スライスのマージ
- データベースまたは増分データ・スライスのコンテンツの置換
- 増分データ・スライス統計の表示
- 増分データ・ロードのディスク・スペースの管理
- Smart View の使用
- 集約ストレージのデータ・ロードでのデータ・ソースに関する相違点
- 集約ストレージのデータ・ロードでのルール・ファイルに関する相違点

データのロードの概要は、第 20 章「データ・ロードまたは次元構築の実行およびデバッグ」を参照してください。

集約ストレージ・データベースにより、最大 1000 万を超えるのメンバーが含まれる非常に大きな次元の分析が容易になります。このような大きいデータベースへのデータ値のロードを効率的にサポートするため、Essbase では次のことが可能です:

- 一時的な集約ストレージのデータ・ロード・バッファによって、複数のデータ・ソースを処理できる
- データ・ロード・バッファで使用されるリソースのパーセンテージを制御できる

- 集約ストレージ・データベースに、データの複数のスライスを含めることができる(データベースに対するクエリーが各スライスにアクセスして、データ・セルのすべてを収集する)
- 増分データのサイズに比例した時間の長さで完了する、増分データ・ロード・プロセスを指定する

➤ 集約ストレージ・データベースに値をロードするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**集約ストレージ・データベースのデータ・ロードまたは次元構築の実行**	Oracle Essbase Administration Services Online Help
MaxL	**alter database** **import data**	『Oracle Essbase テクニカル・リファレンス』
ESSCMD	IMPORT LOADDB	『Oracle Essbase テクニカル・リファレンス』

注： 値が集約によって値がすでに計算されて保管されている場合は、データ値が変更されると、Essbase はより高いレベルの保管済の値を自動的に更新します。追加の計算手順は必要ありません。集約の存在とサイズは、データ・ロードの実行にかかる時間に影響する場合があります。1081 ページの「集約」を参照してください。

注： データをデータベースにロードしているときは、データをエクスポートできません。

データ・ロード・バッファの使用によるデータの増分ロード

MaxL の **import database data** ステートメントを使用して単一のデータ・ソースからデータ値をロードする場合には、集約ストレージ・データ・ロード・バッファは関係しません。

MaxL の複数の **import database data** ステートメントを使用してデータ値を集約ストレージ・データベースにロードする場合は、最初に一時データ・ロード・バッファに値をロードし、すべてのデータ・ソースが読み取られた後でストレージに最終的な書込みを行うことで、パフォーマンスを大幅に向上させることができます。

集約ストレージのデータ・ロード・バッファでは、Essbase によって、すべてのデータ・ソースが読み取られた後で値がソートされ、コミットされます。指定のデータ・セルに複数の(または重複する)レコードが検出される場合、それらの値は累計されます(1062 ページの「セルの競合の解決」を参照)。Essbase では、その後で、累計された値が格納され、データベース内の既存のデータ値の置換、加算または除算が行われます。集約ストレージのデータ・ロード・バッファの使用により、データ・ロードの全体的なパフォーマンスを大幅に向上させることができます。

注： 集約ストレージのデータ・ロード・バッファを使用している場合は、データ・バッファのコンテンツをデータベースにロードする際に、データ・ソースのセット全体に対して値の置換、加算または除算の選択が指定されます。

集約の作成やスライスのマージはリソースを非常に多く使用するため、データ・ロード・バッファがメモリー内に存在している間、これらの操作はできません。ただし、他のデータ・ロード・バッファにデータをロードし、データベース上でクエリーやその他の操作を実行することは可能です。データ・セット全体がデータベースにコミットされて、集約が作成されるまで、クエリーが短時間待機する場合があります。

データ・ロード・バッファは、バッファのコンテンツがデータベースにコミットされるまで、またはバッファが破棄されたときにはアプリケーションが再起動されるまで、メモリー内に存在します。コミット操作が失敗した場合でも、バッファは破棄され、データはデータベースにロードされません。(データ・ロード・バッファは、MaxL の **alter database** ステートメントを使用して手動で破棄できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。)

注： バッファのコンテンツをコミットする前にアプリケーションを停止すると、バッファは破棄されます。このような場合は、アプリケーションを再起動した後、新しいバッファを初期化してそれにデータをロードできます。

➤ 集約ストレージ・データベースのデータ・ロード・バッファを使用するには:

1. データ・ロード・バッファを準備します。このデータ・ロード・バッファには、MaxL の **alter database** ステートメントを使用して集約ストレージのデータ・ロード・バッファを初期化することで、データ値がソートされて累計されます。例:

```
    alter database ASOsamp.Sample
 initialize load_buffer with buffer_id 1;
```

2. MaxL の **import database** ステートメントを使用して、データ・ソースからデータ・ロード・バッファにデータをロードします。複数のステートメントを使用して、複数のデータ・ソースからデータをロードします。`.xls` ファイル、テキスト・ファイルおよび SQL リレーショナル・ソースなどのデータ・ソースの組合せを含めることができます。データ・ソースにルール・ファイルが必要な場合は、ルール・ファイルを指定します。

   次の例では、3 つのデータ・ソース(そのうちの 1 つはルール・ファイルを使用する)を同じデータ・ロード・バッファにロードします:

```
    import database ASOsamp.Sample data
 from server data_file 'file_1.txt'
 to load_buffer with buffer_id 1
 on error abort;
import database ASOsamp.Sample data
 from server data_file 'file_2'
 using server rules_file 'rule'
 to load_buffer with buffer_id 1;
```

```
 on error abort;
import database ASOsamp.Sample data
 from server excel data_file 'file_3.xls'
 to load_buffer with buffer_id 1
 on error abort;
```

データを同時に複数のロード・バッファにロードする方法は、1063 ページの「複数のデータ・ロードの並行での実行」を参照してください。

3. MaxL の **import database** ステートメントを使用して、データ・ロード・バッファのコンテンツをデータベースにコミットします。例:

```
    import database ASOsamp.Sample data
 from load_buffer with buffer_id 1;
```

1 つの MaxL ステートメントで複数のデータ・ロード・バッファのコンテンツをデータベースにコミットする方法は、1063 ページの「複数のデータ・ロードの並行での実行」を参照してください。

次の増分データ・ロードの例では、新しいデータ値が既存の値と交差しないときに最適なパフォーマンスが得られます:

1. ignore_missing_values および ignore_zero_values プロパティを使用して、単一のデータ・ロード・バッファを作成します。例:

```
    alter database ASOsamp.Sample
 initialize load_buffer with buffer_id 1
 property ignore_missing_values, ignore_zero_values;
```

データベースが更新されている間、そのデータベースでデータ送信要求を使用可能にする必要がある場合は、**resource_usage** 文法で 80%の設定でデータ・ロード・バッファを初期化します。例:

```
    alter database ASOsamp.Sample
 initialize load_buffer with buffer_id 1
 resource_usage 0.8 property
 ignore_missing_values, ignore_zero_values;
```

2. データをバッファにロードします。例:

```
    import database ASOsamp.Sample data
 from server data_file 'file_1.txt'
 to load_buffer with buffer_id 1
 on error abort;
import database ASOsamp.Sample data
 from server data_file 'file_2'
 to load_buffer with buffer_id 1;
 on error abort;
```

3. スライスを作成し、値を追加することで、データ・ロード・バッファのコンテンツをデータベースにコミットします。例:

```
    import database ASOsamp.Sample data
from load_buffer with buffer_id 1
add values create slice;
```

データ・ロード・バッファのリソース使用率の制御

増分データ・ロードの実行時、Essbase では、データのソートのために集約ストレージ・キャッシュが使用されます。データ・ロード・バッファで使用できるキャッシュの量は、パーセンテージを指定することで制御できます。パーセンテージは.01 以上、1.0 以下の数値で指定します。パーセンテージで意味があるのは、小数点以下の 2 桁のみです。たとえば、0.029 は 0.02 と解釈されます。デフォルトでは、データ・ロード・バッファのリソース使用率は 1.0 に設定され、データベース上に作成されるすべてのデータ・ロード・バッファの合計使用率が 1.0 を超えることはできません。たとえば、サイズ 0.9 のバッファが存在する場合、サイズが 0.1 より大きい別のバッファを作成できません。

注： 送信操作により、サイズ 0.2 のロード・バッファが内部的に作成されます。したがって、デフォルトのサイズ 1.0 のロード・バッファでは、データ・ロード・バッファのリソースが不十分なため、送信操作は失敗します。

バッファの使用が許可されるリソースの量を設定するには、**resource_usage** 文法で MaxL の **alter database** ステートメントを使用します。

たとえば、resource_usage を合計キャッシュの 50%に設定するには、次のステートメントを使用します:

```
  alter database ASOsamp.Sample
initialize load_buffer with buffer_id 1
resource_usage .5;
```

同時送信操作を実行することをプランしている場合は、ASOLOADBUFFERWAIT 構成設定、および **wait_for_resources** 文法で MaxL の **alter database** ステートメントを使用します。ASOLOADBUFFERWAIT は、**wait_for_resources** オプションを使用して集約ストレージ・データ・ロード・バッファを作成する際や、割当て、カスタム計算、ロックおよび送信操作を行う際に適用されます。

データ・ロード・バッファ・プロパティの設定

データを増分的にロードするときには、データ・ロード・バッファにデータをロードするときに、ソース・データ内の欠落した値とゼロ値がどのように処理されるかを指定できます。

重複セルのセルの競合を解決するため、ロード・バッファに最後にロードされたセルを使用するかどうかを指定できます。

データ・ロード・バッファ・プロパティ:

- ignore_missing_values: 入力データ・ストリーム内の#MI 値は無視されます
- ignore_zero_values: 入力データ・ストリーム内のゼロ値は無視されます
- aggregate_use_last: ロード・バッファに最後にロードされたセルの値を使用することで、重複するセルを組み合せます

注： 集約ストレージ・データベースにテキスト値と日付値をロードするときは、無効な集約を除去するため、aggregate_use_last オプションを使用します。その他のガイドラインは、197 ページの「テキスト・メジャーと日付メジャーのロード、消去およびエクスポート」を参照してください。

コマンドで複数のプロパティを使用し、競合がある場合は、最後にリストされたプロパティが優先されます。

データ・ロード・バッファ・プロパティを設定するには、**property** 文法で MaxL の **alter database** ステートメントを使用します。

例:

```
  alter database ASOsamp.Sample
initialize load_buffer with buffer_id 1
property ignore_missing_values, ignore_zero_values;
```

管理サービス・コンソールの「データ・ロード」ダイアログ・ボックスで、最後にロードされたセルの値を使用することで、欠落した値とゼロ値を無視するオプションや、重複セルを集約するオプションを設定できます。Oracle Essbase Administration Services Online Help の「「データ・ロード」ダイアログ・ボックス」を参照してください。

セルの競合の解決

デフォルトでは、同一キーを持つセルが同じデータ・ロード・バッファにロードされると、Essbase で、値をまとめて追加することによってセルの競合が解決されます。

ロード・バッファに最後にロードされたセルの値を受け入れることによって、重複するセルを組み合せるデータ・ロード・バッファを作成するには、**aggregate_use_last** 文法で MaxL の **alter database** ステートメントを使用します。

例:

```
  alter database ASOsamp.Sample
initialize load_buffer with buffer_id 1
property aggregate_use_last;
```

注： この **aggregate_use_last** 文法でデータ・ロード・バッファを使用すると、重複キーが存在しない場合でも、データ・ロードにはきわめて時間がかかります。

複数のデータ・ロードの並行での実行

集約ストレージ・データベースには、複数のデータ・ロード・バッファが存在できます。時間を節約するため、データを複数のデータ・ロード・バッファに同時にロードできます。

データベース上では、常に 1 つのデータ・ロード・コミット操作しか有効にできませんが、同じコミット操作の中で複数のデータ・ロード・バッファをコミットすると、バッファを個別にコミットするより処理が速くなります。

注： 管理サービス・コンソールを使用してデータを集約ストレージ・データベースにロードするときは、1 つのデータ・ロード・バッファのみ使用されます。

同時に複数のデータ・ロード・バッファにデータをロードするには、別々の MaxL シェル・セッションを使用します。たとえば、1 つの MaxL シェル・セッションで、ID が 1 のバッファにデータをロードします:

```
   alter database ASOsamp.Sample
  initialize load_buffer with buffer_id 1 resource_usage 0.5;
import database ASOsamp.Sample data
  from data_file "dataload1.txt"
  to load_buffer with buffer_id 1
  on error abort;
```

同時に、もう 1 つの MaxL シェル・セッションで、ID が 2 のバッファにデータをロードします:

```
   alter database ASOsamp.Sample
  initialize load_buffer with buffer_id 2 resource_usage 0.5;
import database ASOsamp.Sample data
  from data_file "dataload2.txt"
  to load_buffer with buffer_id 2
  on error abort;
```

データがデータ・ロード・バッファに完全にロードされたら、バッファ ID のカンマ区切りのリストを使用することで、1 つの MaxL ステートメントを使用して両方のバッファのコンテンツをデータベースにコミットします:

たとえば、次のステートメントでは、バッファ 1 および 2 のコンテンツがロードされます:

```
   import database ASOsamp.Sample data
  from load_buffer with buffer_id 1, 2;
```

注： 集約ストレージ・データベースに SQL データをロードするときは、データの並行ロードに、最大 8 つのルール・ファイルを使用できます。この機能は、ここで説明したプロセスとは異なります。複数の SQL データのロードを並行して実行する場合は、**using multiple rules_file** 文法で MaxL の 1 つの **import database** ステートメントを使用します。Essbase によって、複数の一時集約ストレージのデータ・ロード・バッファ(ルール・ファイルごとに 1 つ)が初期化され、1 回の操作ですべてのバッファのコンテンツがデータベースにコミットされます。『Oracle Essbase SQL インタフェース・ガイド』を参照してください。

集約ストレージ・データベースのデータ・ロード・バッファのリスト

集約ストレージ・データベースには、複数のデータ・ロード・バッファが存在できます。集約ストレージ・データベースに存在するデータ・ロード・バッファのリストと説明については、次のように **list load_buffers** 文法で MaxL の **query database** ステートメントを使用してください:

```
query database
appname.dbname
 list load_buffers;
```

このステートメントでは、既存の各データ・ロード・バッファに関する次の情報が戻されます:

表 205　データ・ロード・バッファ情報

フィールド	説明
buffer_id	データ・ロード・バッファの ID (1 から 4,294,967,296 の間の数)。
internal	データ・ロード・バッファが Essbase によって内部で作成された(TRUE)か、またはユーザーによって作成された(FALSE)かを指定するブール。
active	データ・ロード・バッファがデータ・ロード操作によって現在使用中であるかどうかを指定するブール。
resource_usage	データ・ロード・バッファによる使用が許可されている集約ストレージ・キャッシュのパーセンテージ(.01 から 1.0 までの数値)。
aggregation method	バッファ内の同じセルに対して複数の値を結合するために使用されるメソッドの 1 つ: ● AGGREGATE_SUM: 同じセルに対して複数の値がバッファに含まれる場合に値が追加されます。 ● AGGREGATE_USE_LAST: ロード・バッファに最後にロードされたセルの値を使用して重複セルが結合されます。
ignore_missings	入力データ・ストリームで#MI 値を無視するかどうかを指定するブール。
ignore_zeros	入力データ・ストリームでゼロを無視するかどうかを指定するブール。

データ・スライスの作成

データ・ロード・バッファを集約ストレージ・データベースに増分的にコミットすれば、スライスを作成できます。新しいスライスをデータベースにロードした後、新しいデータがクエリーに表示される前に、Essbase によって、スライス上の必要なすべてのビュー(集約ビューなど)が作成されます。

データ・スライスの作成は、増分データ・ロードのパフォーマンスを向上させるため、有用です。増分データ・ロードにかかる時間は、新規データの量に比例します。データベースのサイズは要因になりません。

データ・スライスを作成するには、**create slice** 文法で MaxL の **import database** ステートメントを使用します。

たとえば、値(デフォルト)を上書きすることでスライスを作成するには、次のステートメントを使用します:

```
  import database ASOsamp.Sample data
from load_buffer with buffer_id 1
override values create slice;
```

注： スライスを作成するときに上書き値を使用すると、#MISSING 値はゼロと置き換えられます。このオプションを使用すると、add values オプションや subtract values オプションを使用するよりもかなり時間がかかります。

管理サービス・コンソールでは、「データ・ロード」ダイアログ・ボックスでスライスを作成するオプションを設定できます。Oracle Essbase Administration Services Online Help の「「データ・ロード」ダイアログ・ボックス」を参照してください。

増分データ・スライスのマージ

メインのデータベース・スライスを変更しないまま、すべての増分データ・スライスをメインのデータベース・スライスにマージしたり、すべての増分データ・スライスを単一のデータ・スライスにマージしたりできます。スライスをマージするには、データをロードするための権限と同じ権限(管理者権限またはデータベース・マネージャ権限)を持つ必要があります。

新しいデータ・スライスのデータがデータ・ロード・バッファ内に存在すると、Essbase によって、増分データ・スライスがスキャンされ、それらのデータを新しいスライスと自動的にマージするかどうかが検討されます。自動マージの対象になるには、スライスが 100,000 セル未満か、新しいスライスのサイズの 2 倍より小さいものである必要があります。5,000,000 セルより大きいスライスが自動的にマージされることはありません。たとえば、新しいスライスに 300,000 個のセルが含まれている場合、90,000 個のセルと 500,000 個のセルが含まれている増分データ・スライスは、自動的に新しいセルとマージされます。700,000 個のセルが含まれて入る増分データ・スライスは、自動的にはマージされません。

データベースに新しい入力ビューが書き込まれた後、Essbase でスライスの集約ビューが作成されます。新しいスライス用に作成されるビューは、メインのデータベース・スライス上に存在するビューのサブセットです。

注： マージの実行中にはデータをエクスポートできません。

logical clear region 操作を使用する領域からデータを消去し、結果として消去したセルの値がゼロになる場合は、マージ操作中にゼロ値のセルを削除するよう選択できます。1070 ページの「集約ストレージ・データベースの特定の領域からのデータの消去」を参照してください。

マージ操作を実行するには、**merge** 構文で MaxL の alter database ステートメントを使用します。

たとえば、すべての増分データ・スライスをメインのデータベース・スライスにマージするには、次のステートメントを使用します:

```
  alter database ASOsamp.Sample
merge all data;
```

すべての増分データ・スライスをメインのデータベース・スライスにマージし、ゼロ値のセルを削除するには、次のステートメントを使用します:

```
  alter database ASOsamp.Sample
merge all data remove_zero_cells;
```

すべての増分データ・スライスを単一のデータ・スライスにマージするには、次のステートメントを使用します:

```
  alter database ASOsamp.Sample
merge incremental data;
```

管理サービス・コンソールでは、データ・スライスをマージできます。Oracle Essbase Administration Services Online Help の「増分データ・スライスのマージ(集約ストレージ・データベース)」を参照してください。

注： 集約ストレージ・アプリケーションをコピーする前に、すべての増分データ・スライスをメイン・データベース・スライスにマージする必要があります。マージされていない増分データ・スライスのデータはコピーされません。1065 ページの「増分データ・スライスのマージ」を参照してください。

データベースまたは増分データ・スライスのコンテンツの置換

完全に再ロードするには十分に小さくて、データの待ち時間も短いデータ・セットの場合は、Essbase で、現在のデータベースのコンテンツを削除し、データベースを指定したデータ・ロード・バッファのコンテンツと置き換えることができます。このアトミックな置換機能では、サービスを中断することなく、データベースの古いコンテンツのクエリーが新しいコンテンツに移行されます。新たにロードされたデータ・セットは、置換されたデータ・セット用に存在していたものと同じビューのセットを作成するために集約されます。

Essbase では、データベース内のすべての増分データ・スライスのコンテンツの微細な置換も可能です。データを、更新されることのない比較的大きい静的なデータ・セットと、個々の更新を識別するのは難しくても揮発性のデータ・セットに限定されている比較的小さい揮発性のデータ・セットに分けることができる状況を考えてみてください。たとえば、大きい静的なデータ・セットが直前の 3 年間の履歴トランザクション・データで構成されていても、過去 2 か月のトランザクション・データについては、ユーザーがソース・データベース内のトランザクションの特徴を変更できるとします。これらの変更の追跡は、途方もなく複雑になる可能性があります。この場合、静的データ・セットをデータベース内のメイン・スライスとしてロードし、揮発性のデータ・セットを 1 つ以上の増分スライスとしてロードできます。

Essbase では、すべての増分データ・スライスの現在のコンテンツが削除され、(**add values** 文法を使用して)指定されたデータ・ロード・バッファを含む新しいスライスが作成されます。新たにロードされたデータ・セットは、メイン・スライス上に存在するビューのセットに基づいて、集約ビューで拡張されます。

注： **override** 文法を使用するには、最適なパフォーマンスを得るために ignore_missing_values プロパティでデータ・ロード・バッファを作成します。さらに、静的なデータ・セットと揮発性のデータ・セットの間に競合がないことを確認する必要があります(たとえば、同じセルに対して各データ・セット内に値が存在することはできません)。

データベース内のデータベースのコンテンツまたは増分データ・スライスを置換するには、**override** 文法で MaxL の **import database** ステートメントを使用します。

たとえば、データベースのコンテンツを置換するには、次のステートメントを使用します:

```
  import database ASOsamp.Sample data
from load_buffer with buffer_id 1
override all data;
```

すべての増分データ・スライスのコンテンツを新しいスライスと置き換えるには、次のステートメントを使用します:

```
  import database ASOsamp.Sample data
from load_buffer with buffer_id 1
override incremental data;
```

注： 上書き置換が失敗すると、Essbase では古いデータ・セットを引き続き使用します。

管理サービス・コンソールの「データ・ロード」ダイアログ・ボックスで、データベースまたはデータ・スライスのコンテンツを置換するオプションを設定できます。Oracle Essbase Administration Services Online Help の「データベースまたは

データ・スライス・コンテンツの置換(集約ストレージ・データベース)」を参照してください。

増分データ・スライス統計の表示

Essbase では、増分データ・スライスのサイズと数、および増分データ・スライスのクエリーにかかるコストの統計を提供します。

すべての増分データ・スライスにアクセスするためのクエリーにかかる時間は、パーセンテージ(.01 以上、1.0 以下)で表されます。データベースにメインのスライスと複数の増分データ・スライスがある場合、クエリー統計 0.66 は、クエリー時間の 3 分の 2 が増分データ・スライスのクエリーに費やされ、3 分の 1 がメインのデータ・スライスのクエリーに費やされたことを意味します。増分データ・スライスのクエリーのコストが高すぎる場合は、スライスをマージできます。

増分データ・スライス統計を表示する方法は、Oracle Essbase Administration Services Online Help の「集約ストレージ統計の表示」を参照してください。

増分データ・ロードのディスク・スペースの管理

集約ストレージ・データベースでの増分データ・ロードには、現在のデータ・ファイルのサイズの最大 2 倍のディスク・スペースが使用される場合があります。データベースの大きさが 2 GB を超える場合は、デフォルトのテーブルスペースの最大ファイル・サイズが 2 GB を超えないように設定することで、ディスク・スペースの使用率を削減できます。1128 ページの「テーブルスペースでの処理」を参照してください。

➤ デフォルトのテーブルスペースの最大ファイル・サイズを設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	テーブルスペースの管理	Oracle Essbase Administration Services Online Help
MaxL	**alter tablespace**	『Oracle Essbase テクニカル・リファレンス』

Smart View の使用

Smart View では、submit コマンドは、増分データ・ロード機能を **override** 文法で使用することと同じです。

送信操作を実行している間、ロック、ロック解除および取得とロックの新しい要求は、送信操作が完了するまで待機します。

Oracle Hyperion Smart View for Office User's Guide を参照してください。

集約ストレージのデータ・ロードでのデータ・ソースに関する相違点

集約ストレージ・データベースに値をロードするためにデータ・ソース・レコードを処理する際、Essbase ではメンバーに式が存在しないレベル 0 の次元の交差のみを対象にしてソース・データ・レコードが処理されます。

次の例は、レベル 0 の交差のみのレコードを含むデータ・ソースを示しています。最後のフィールドにはデータ値が含まれていて、その他のフィールドは、それらの各次元のレベル 0 のメンバーです。

```
   Jan, Curr Year, Digital Cameras, CO, Original Price, 10784
Jan, Prev Year, Camcorders, CO, Original Price, 13573
```

Essbase は上位レベルのメンバーを指定するレコードを無視し、データ・ロードの完了時にスキップされたレコードの件数を表示します。

たとえば、メンバー Mid West はレベル 1 のメンバーであるため、次のレコードはスキップされます:

```
   Jan, Curr Year, Digital Cameras, Mid West, Original Price, 121301
```

Essbase サーバーでは、データベースに値をコミットする前にレコードの読取りとソートが内部的に行われるため、データ・ソースをソートする必要はありません。

集約ストレージのデータ・ロードでのルール・ファイルに関する相違点

集約ストレージ・データベースに値をロードするためのルール・ファイルの指定は、集約ストレージのデータ・ロード・プロセスに反映されます。

ブロック・ストレージのデータ・ロードでは、既存の値に上書きするか、既存の値にデータ・ソースの値を足すか、既存の値から差し引くかを、ルール・ファイルを使用してデータ・ソースごとに選択します。

集約ストレージ・データ・ロード・バッファを使用した集約ストレージのデータ・ロードでは、データ・ロード・バッファに収集されたすべてのデータ・ロード・ソースに対して選択した操作が行われてから、データベースにロードされます。

ブロック・ストレージ・アウトラインと集約ストレージ・アウトラインに定義されたルール・ファイルを使用するには、まずルール・ファイルを集約ストレージ・アウトラインに関連付けます。Oracle Essbase Administration Services Online Help の「エディタでのアウトラインの関連付け」を参照してください。

集約ストレージ・データベースからのデータの消去

サブトピック

集約ストレージ・データベースから、データを選択して消去するか、すべてのデータを消去できます。

1089 ページの「集約の消去」も参照してください。

集約ストレージ・データベースの特定の領域からのデータの消去

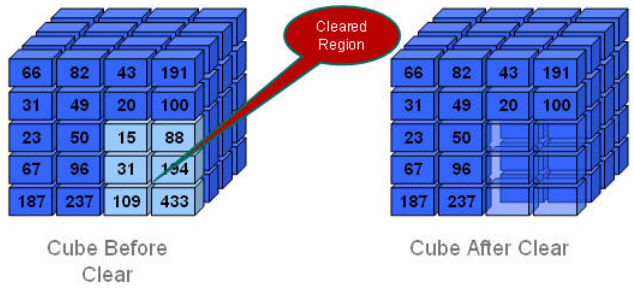
集約ストレージ・データベース内の指定した領域からはデータを消去し、他の領域にあるデータを保持できます。この機能は、揮発性のデータ(過去 1 か月間に対応するデータなど)を削除しても、履歴データは保持したいときには便利です。データを消去するには、データ・マネージャ権限または管理者権限を持っている必要があります。

領域からのデータの消去方法:

- 物理

  指定した領域内の入力セルは、図 164 に示すように、集約ストレージ・データベースから物理的に削除されます。

  図 164　データの領域の物理的な消去



  データベース内に複数のデータ・スライスがある場合は、物理消去領域操作によって、すべてのデータ・スライスがメインのデータ・スライスに自動的にマージされます。指定した領域のデータが消去されると、Essbase では、消去領域操作が行われるまでメインのデータ・スライス内に存在していたすべての集約ビューが生成されます。

  データを物理的に消去するプロセスが完了するには、消去されるデータのサイズではなく、入力データのサイズに比例した長さの時間がかかります。したがって、この方法は、データの大きいスライスを削除する場合のみ使用することをお薦めします。

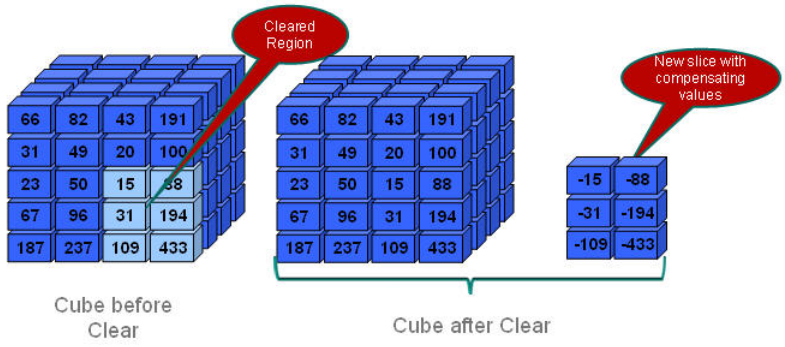
データを物理的に消去するには、MaxL の **alter database** ステートメントを **clear data in region** 文法と **physical** キーワードを指定して使用します:

```
alter database
appname
.
dbname
 clear data in region 'MDX set expression' physical;
```

- 論理

  指定した領域内の入力セルは、図 165 に示すように、負の補正値(消去するセルをゼロの値にする値)を含む新しいデータ・スライスに書き込まれます。

図 165　データ領域の論理的な消去



論理消去領域操作では、ゼロ値を持つデータのみ、自動的にメインのデータ・スライスにマージされます。データベース内の他のデータ・スライスはマージされません。指定した領域のデータが消去された後、Essbase では新しいデータ・スライスのみの集約ビューが生成されます。

データを論理的に消去するプロセスは、消去されるデータのサイズに比例する時間の長さで完了します。このオプションでは、相殺セルが作成されるので、データベースのサイズが増加します。

データを論理的に消去するには、MaxL の **alter database** ステートメントを **clear data in region** 文法で、**physical** キーワードを指定せずに使用します:

```
alter database
appname
.
dbname
 clear data in region 'MDX set expression';
```

論理的に消去された領域に対するクエリーでは、#MISSING 値ではなくゼロ値が戻されます。空のセルの#MISSING 値に依存する式の更新が必要な場合があります。

値がゼロのセルを削除するには、**alter database** MaxL ステートメントを **merge** 文法と **remove_zero_cells** キーワードとともに使用します。『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： 同じ領域で 2 回目の論理消去領域操作を実行することはお薦めしません。これは、2 回目の操作では、最初の操作で作成された補正セルが消去されず、新しい補正セルが作成されないためです。

消去する領域を指定するときは、次のガイドラインに従ってください:

- 領域は対称である必要があります。
  - {(Jan, Budget)}は、「Jan」のすべての「Budget」データを消去する有効な対称の領域です。
  - {(Jan, Forecast1),(Feb, Forecast2)}は、2 つの非対称の領域(「Jan, Forecast1」と「Feb, Forecast2」)で構成されているため、無効な領域です。
- 領域指定内の次元の個々のメンバーは、保管済メンバーである必要があります。
- 領域内のメンバーは、次のものにすることはできません:
  - 動的メンバー(暗黙的または明示的な MDX 式を持つメンバー)
  - 属性次元からのメンバー

    セルを属性によって消去する必要がある場合は、Attribute MDX 関数を使用します。
- 領域内のメンバーは、保管階層の上位レベルのメンバーにできます。これは、複数のレベル 0 のメンバーを指定する場合に便利な方法です。

  たとえば、Qrt1 を指定できます。これは、Jan、Feb および Mar(Qrt1 のレベル 0 の子)を指定することと同じです:

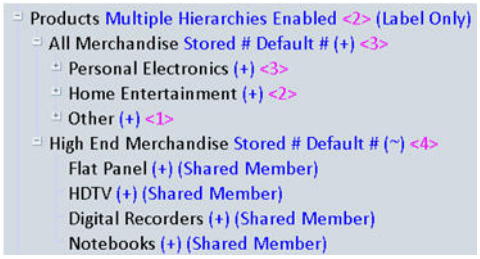
次の 2 つの MaxL ステートメントでは、同じ結果が生成されます:

```
alter database
appname
.
dbname
 clear data in region '{Qtr1}';
```

```
alter database
appname
.
dbname
 clear data in region '{Jan, Feb, Mar}';
```

- (物理的なデータの消去のみ)領域内のメンバーは、代替階層の上位レベル・メンバーにできます。

  たとえば、High End Merchandise を指定できます。これは、Flat Panel、HDTV、Digital Recorders および Notebooks (High End Merchandise の共有のレベル 0 の子)を指定することと同じです:



  次の 2 つの MaxL ステートメントでは、同じ結果が生成されます:

```
alter database
appname
.
dbname
 clear data in region '{High End Merchandise}';


alter database
appname
.
dbname
 clear data in region '{[Flat Panel],[HDTV],[Digital Recorders],
[Notebooks]}';
```

  データを論理的に消去するときに、代替階層のメンバーを指定するには、Descendants MDX 関数を使用します。

  注： 領域に代替階層の上位レベルのメンバーが含まれている場合は、パフォーマンスが低下する可能性があります。このような場合は、レベル 0 のメンバーのみを使用することを検討してください。

- MDX のセット式は一重引用符で囲む必要があります。

たとえば、「Forecast1」および「Forecast2」のシナリオのすべての 1 月のデータを消去するには、次のステートメントを使用します:

```
  alter database ASOsamp.Sample clear data in region 'CrossJoin({Jan},{Forecast1,
Forecast2})';
```

消去領域操作の間は、データベースを更新する操作(データのロード、データ・スライスのマージ、または別の領域からのデータの消去など)は実行できず、データをエクスポートすることもできません。データベースのクエリーは可能ですが、クエリーの結果は、消去領域操作の前のデータ・セットに基づいたものになります。

clear data in region 文法では、データベース全体からデータを消去できません。1074 ページの「集約ストレージ・データベースからのすべてのデータの消去」を参照してください。

集約ストレージ・データベースからのすべてのデータの消去

集約ストレージ・データベースからのすべてのデータの消去は、ブロック・ストレージ・データベースの場合と同じです。データベース全体を消去するには、次のいずれかを使用します:

- MaxL の **alter database** ステートメント(**reset** 文法):

```
alter database
appname.dbname
 reset;
```

- 管理サービス・コンソール

  Oracle Essbase Administration Services Online Help の「データの消去」を参照してください。

集約ストレージ・アプリケーションのコピー

集約ストレージ・アプリケーションですべてのデータをコピーするには、すべての増分データ・スライスをメインのデータベース・スライスにマージする必要があります。マージされていない増分データ・スライスのデータはコピーされません。

1065 ページの「増分データ・スライスのマージ」を参照してください。

データ・ロードと次元構築の結合

集約ストレージのデータ・ロード・バッファを使用するときは、データ・ソースとルール・ファイルを組み合せて、メンバーをアウトラインに追加したり、レベル 0 のセルにデータ値をロードしたりできます。ユーザーが指定するファイルの順序とは無関係に、Essbase によってアウトラインに変更が加えられ、その後データ値がロードされます。

注： 集約ストレージ・データベースの次元構築とデータ・ロードは、Administration Services では 1 つの操作で結合できますが、次元構築操作とデータ・ロード操作は別々に行うことをお薦めします。これは、次元構築によってはデータが消去されることがあり、それによってデータが失われてしまう可能性があるためです。1053 ページの「集約ストレージ・データベースでの次元構築」を参照してください。

集約ストレージ・データベースの計算

サブトピック

集約ストレージ・データベースの値は、アウトライン構造と MDX 式によって計算されます。データ・ロードが完了すると、集約ストレージ・データベースの計算に必要なすべての情報が使用可能になります。取得要求が出されると、レベル 0 のメンバーにロードされた値を集計し、式を計算することによって、Essbase サーバーで必要な値が計算されます。取得のために計算された値は格納されません。

取得パフォーマンスを向上させるため、Essbase では、値を集約し、それらの値を前もって格納できます。ただし、すべての値を集約して格納すると、ストレージ用のディスク・スペースを必要とするプロセスが長々と続く可能性があります。Essbase には、時間とストレージのリソースのバランスをとる、高機能な集約プロセスがあります。1079 ページの「集約ストレージ・データベースの集約」を参照してください。

取得のために集約ストレージ・データベースを準備するには、アウトラインを作成し、レベル 0 の値をロードします。次に、追加の値を集約して格納することでデータベースを計算し、残りの値は取得時に計算されるようにします。

注： 特殊な計算やデータの依存性を取り扱う計算スクリプトがデータベースに必要な場合は、データベースの設計を変更するか、ブロック・ストレージ・データベースを使用することを検討してください。

表 193 を参照してください。

データ値に影響するアウトラインの要因

集約ストレージ・アウトラインの階層構造により、値のロール・アップ方法が決定されます。レベル 0 のメンバーの値はレベル 1 のメンバーの値にロール・アップし、レベル 1 のメンバーの値はレベル 2 のメンバーの値にロール・アップする、などのようになります。

動的階層のメンバーに割り当てられた集計演算子では、ロールアップで使用される次の演算子が定義されます: 加算(+)、減算(-)、乗算(*)、除算(/)、パーセント(%)、非演算(~)および集計なし(^)。演算子の説明は、表 16 を参照してください。

注： 保管階層のメンバーでは、加算(+)または非集計演算子(~)のみを使用できます。

より複雑な演算では、動的階層のメンバーに MDX 式を指定できます。MDX 式は MDX 数値式と同じフォーマットで記述されます。1028 ページの「集約ストレージ・アウトラインでの式の作成」を参照してください。

集約ストレージ・データベースに適用されないブロック・ストレージの計算機能

ブロック・ストレージ・データベースの次の機能は、集約ストレージ・データベースには適用されません:

- 計算スクリプトの計算
- 動的計算メンバーと動的計算および保管メンバーのストレージのプロパティ
- ブロック・ストレージ式の構文と、式に事前定義された Essbase 関数
- カスタム定義計算関数およびカスタム定義計算マクロ
- 集約ストレージの動的階層のメンバーを除く次元のメンバー式
- レベル 0 より上のメンバーの交差にプレロードされた値
- 2 パス計算タグ
- 高機能計算などのブロック・ストレージのパフォーマンスに関する機能

計算順序

集約ストレージの計算順序とブロック・ストレージの計算順序は異なります。集約ストレージ・データベースでは、Essbase によりデータが次の順序で計算されます:

1. **保管階層のメンバーと属性次元の集約。** メンバーと次元が集約される順序は内部的に最適化され、データベース・アウトラインの性質と既存の集約に応じて変わります。集約は付加的であるため、Essbase が次元とメンバーを集約する順序は結果には影響しません。

   集約ストレージ・データベースの内部の集約順序は予測できないため、内在している丸め誤差も予測できません。これらの丸め誤差は、コンピュータ計算では予想される動作であり、関係しているデータ値についてはごくわずかです。
2. **動的階層次元のメンバーと式の計算。** メンバーと式が評価される順序は、各メンバーまたは次元に設定できる解決順プロパティによって定義されます。計算順序は計算結果に影響を及ぼす場合があります。

解決順のプロパティ

解決順の概念は、クエリー実行に適用されます。セルが多次元クエリーで評価されるとき、計算が解決される順序はあいまいな場合があります。このあいまいさをなくすため、解決順のプロパティを使用して、必要な計算の優先度を指定できます。

注： 各メンバーの解決順は、解決順のプロパティをメンバー・レベルまたは次元レベルで設定することによって指定することをお薦めします。解決順が指定されていない式を持たないメンバーは、その次元の解決順を継承します。解決順が指定されていない式を持つメンバーは、ゼロの解決順を持ちます。

➤ メンバーまたは次元の解決順を指定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**集約ストレージ・データベースでのメンバーと次元の計算順序の指定**	Oracle Essbase Administration Services Online Help
MaxL	MDX クエリーの WITH セクションの **solve_order** パラメータ	『Oracle Essbase テクニカル・リファレンス』(「MDX」、「文法規則」、「WITH 指定」を参照)

解決順のプロパティの値により、Essbase で式が計算される優先度が決まります。解決順が指定されているメンバーに対する式は、最も低い解決順から最も高い解決順の順序で計算されます(1077 ページの「解決順のプロパティの使用例」を参照)。0 から 127 の範囲で解決順序を指定できます。デフォルトは 0 です。

解決順は、メンバー・レベルまたは次元レベルで指定できます。Essbase では、計算の優先順位は次の情報を使用して決定されます:

1. メンバーの解決順
2. 次元の解決順(メンバーの解決順を指定しない式を持たないメンバーは、その次元の解決順を継承します。メンバーの解決順を指定しない式を持つメンバーは、ゼロの解決順を持ちます。)

   複数のメンバーが同じ解決順を持つ場合、メンバーは、その次元がデータベース・アウトライン内で出現する順序の逆順で評価されます。アウトライン内で後の方に出現するメンバーを優先します。

   タイの状況の計算順序は、ブロック・ストレージ・データベースの MDX クエリーで定義された計算済メンバーとは異なります。『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： メンバー式が別のメンバーの値に依存する場合、その式を持つメンバーは、依存しているメンバーよりも高い解決順を持つ必要があります。たとえば、図 167 の ASOsamp.Sample データベース・アウトラインでは、「Avg Units/Transaction」は「Units」および「Transactions」の値に依存します。「Avg Units/Transaction」は、「Units」および「Transactions」よりも高い解決順を持つ必要があります。

解決順のプロパティの使用例

次の例は、ASOSamp.Sample データベースをベースにしています。クエリー結果のあいまいさをなくすため、この例では、解決順のプロパティを使用して必要な計算の優先度を指定しています。

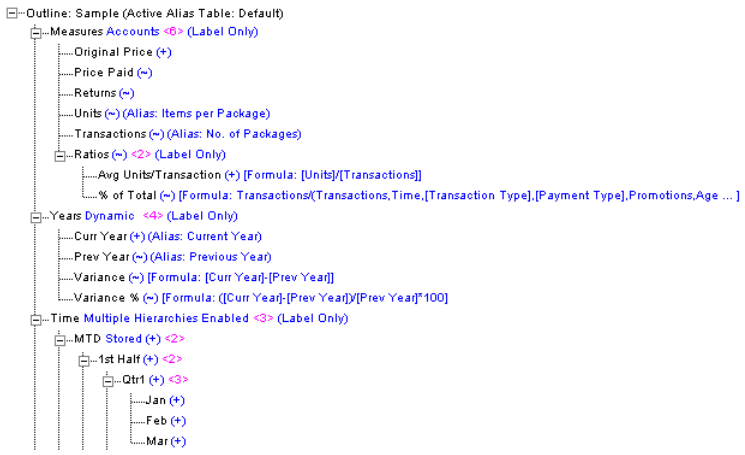
図 166 のスプレッドシートのクエリーでは、今年の 1 月と去年の 1 月の売上数量と取引件数のデータが取得されます。「Variance」メンバーは、今年と去年の間の差異を示します。「Avg Units/Transaction」メンバーは、取引ごとの売上数量の比率を示します。

図 166　2 つの比率の差異を示す ASOSamp.Sample データベースのスプレッドシート・クエリーの結果(C12)

	A	B	C
1			
2			
3			Jan
4	Curr Year	Units	42228
5		Transactions	44500
6		Avg Units/Transaction	0.94894382
7	Prev Year	Units	31643
8		Transactions	33160
9		Avg Units/Transaction	0.954252111
10	Variance	Units	10585
11		Transactions	11340
12		Avg Units/Transaction	-0.005308291
13	Variance %	Units	33.45131625
14		Transactions	34.19782871
15		Avg Units/Transaction	-0.556277601
16			

図 167 は、これらのメンバーのデータベース・アウトラインと、「Variance」メンバーと「Avg Units/Transaction」メンバーに適用される式を示しています。

図 167　「Measures」、「Years」および「Time」次元が含まれている ASOSamp.Sample データベース



取引ごとの平均数量の差異(図 166 のセル C12)を計算すると、結果は 2 つの比率の差異であることも、2 つの差異の比率であることもあります。結果は、Essbase で「Variance」に対する式か「Avg Units/Transaction」に対する式のどちらが優先されるかによって決まります。

解決順のプロパティの値は、データベース・アウトライン内のメンバーに添付され、Essbase で式が評価される優先度を決定します。解決順の高いメンバーに対する式が優先されます。

この例では、「Variance」メンバーが「Avg Units/Transaction」メンバーより高い解決順を持つ場合、「Variance」メンバーに対する式が優先し、その結果は 2 つの比率の差異です。これは ASOsamp.Sample データベースの場合であり、「Variance」メンバーの解決順が 20 で、「Avg Units/Transaction」メンバーの解決順が 10 のためです。「Variance」メンバーの方が解決順が高いため、「Variance」に対する式が優先されます。図 166 のクエリーのセル C12 の結果は、表 206 に示されているように、2 つの比率の差異です:

表 206　2 つの比率の差異を求めるように解決順のプロパティを指定

メンバー	解決順	式	「Variance」と「Avg Units/Transaction」の交差の結果(図 166 のセル C12)
分散	20	Curr Year - Prev Year	今年の 1 取引当たりの平均数量- 去年の 1 取引当たりの平均数量 0.94894382 (セル C6) - 0.954252111 (セル C9) = -0.005308291 (セル C12)
Avg Units/ Transaction	10	Units/ Transactions	

かわりに、ASOSamp.Sample データベースを変更し、「Avg Units/Transaction」メンバーに「Variance」メンバーよりも高い解決順を指定すると、「Avg Units/Transaction」メンバーが優先され、結果は、表 207 と図 168 に示されているように、2 つの差異の比率になります:

表 207　2 つの差異の比率を求めるように解決順のプロパティを指定

メンバー	解決順	式	「Variance」と「Avg Units/Transaction」の交差の結果(図 168 のセル C12)
分散	10	Curr Year - Prev Year	今年と去年の数量の差異/取引数の差異 10585 (セル C10) / 11340 (セル C11) = 0.933421517 (セル C12)
Avg Units/ Transaction	20	Units/Transactions	

図 168　2 つの差異の比率を示す ASOSamp.Sample データベースのスプレッドシート・クエリーの結果(C12)

	A	B	C
1			
2			
3			Jan
4	Curr Year	Units	42228
5		Transactions	44500
6		Avg Units/Transaction	0.94894382
7	Prev Year	Units	31643
8		Transactions	33160
9		Avg Units/Transaction	0.954252111
10	Variance	Units	10585
11		Transactions	11340
12		Avg Units/Transaction	0.933421517
13	Variance %	Units	33.45131625
14		Transactions	34.19782871
15		Avg Units/Transaction	0.978170764
16			

集約ストレージ・データベースの集約

集約ストレージ・データベースでは、データ値がアウトラインのレベル 0 のセルにロードされた後、別個の計算ステップは必要ありません。ユーザーは、データベース内の任意のポイントから、現在の取得のみについて、動的に集約される値を取得して表示できます。集約ストレージ・データベースはブロック・ストレージ・データベースより小さいため、データ値をすばやく取得できます。

データベースが増大するにつれて、取得では、クエリーの計算の要件を満たすためにより多くのデータ値を処理する必要があります。取得をより速くするため、Essbase では、データ値を事前計算し、それらの値を集約に格納できます。データベースのサイズが 100 万の集約セルに近づいた場合は、集約の実行を積極的に検討してください。データベース使用状況と使用環境によっては、小さいデータベースを同様に事前計算することによって、パフォーマンスを向上できます。集約ス

トレージ・データベースの計算には、管理サービス・コンソールまたは MaxL のいずれかを使用できます。

集約に関連する用語の理解

次のトピックでは、集約ストレージ・データベースの計算の説明に不可欠な用語について説明します。

集約セル

次元全体のレベル 0 の交差のセルのうち、式を持たないセルを、入力セルといいます。データ値はそれらのセルにロードできます。会計次元または動的階層のメンバーが含まれている上位レベルのセルは、常に取得時に計算されます。次元全体の他のすべての上位レベルの交差は、集約セルです。たとえば、図 163 では、「Price Paid」>「Curr Year」>「1st Half」>「Portable Audio」>「CO」は集約セルです。「Original Price」>「Curr Year」>「Jan」>「Camcorders」>「CO」は、もう 1 つの集約セルです。集約セルの値は、下位レベルの値からロール・アップされる必要があります。

集約セルの値は要求ごとに計算されますが、事前に計算してディスクに保管しておくこともできます。

集約ビュー

Essbase がどの集約セルを事前計算して保管するかを定義するときは、集約ビューを操作します。集約ビューは集約セルの集合です。この集合は、各次元内のメンバーのレベルに基づきます。

たとえば、図 163 のアウトラインの 1 つの集約ビューについて考えてください。この集約ビューには、次の次元レベルの集約セルが含まれています:

- 「Measures」次元、レベル 0
- 「Years」次元、レベル 0
- 「Time」次元、階層 0 のレベル 1
- 「Product」次元、階層 0 のレベル 2
- 「Geography」次元、レベル 0

この例の集約ビューは 0, 0, 1/0, 2/0, 0 と表されます。

それぞれの次元は、左から右に、アウトライン内でのその順序で表されます。次元に階層が含まれている場合、注釈によってその階層内でのメンバー・レベルが示されます。次元内の階層は、階層 0 からトップダウンで番号が付けられます。

0, 0, 1/0, 2/0, 0 集約ビューには、次のメンバーの交差を含む集約セルが含まれています:

```
   Original Price, Curr Year, Qtr1, Personal Electronics, CO
Original Price, Curr Year, Qtr1, Personal Electronics, KS
Original Price, Curr Year, Qtr1, Home Entertainment,  CO
Original Price, Curr Year, Qtr1, Home Entertainment,  KS
```

```
Original Price, Curr Year, Qtr2, Personal Electronics, CO
Original Price, Curr Year, Qtr2, Personal Electronics, KS
Original Price, Curr Year, Qtr2, Home Entertainment,  CO
Original Price, Curr Year, Qtr2, Home Entertainment,  KS
Original Price, Curr Year, Qtr3, Personal Electronics, CO
Original Price, Curr Year, Qtr3, Personal Electronics, KS
Original Price, Curr Year, Qtr3, Home Entertainment,  CO
Original Price, Curr Year, Qtr3, Home Entertainment,  KS
Original Price, Curr Year, Qtr4, Personal Electronics, CO
Original Price, Curr Year, Qtr4, Personal Electronics, KS
Original Price, Curr Year, Qtr4, Home Entertainment,  CO
Original Price, Curr Year, Qtr4, Home Entertainment,  KS
Original Price, Prev Year, Qtr1, Personal Electronics, CO
Original Price, Prev Year, Qtr1, Personal Electronics, KS
Original Price, Prev Year, Qtr1, Home Entertainment,  CO
Original Price, Prev Year, Qtr1, Home Entertainment,  KS
and so on...
```

集約

集約とは、アウトライン階層に基づいた、レベル 0 のデータ値の集計のことです。集約には 1 つ以上の集約ビューが含まれます。Essbase には、ロール・アップされる集約ビューを選択し、それらを集約してから、選択したビューにセルの値を保管するという、高機能な集約プロセスがあります。集約に、データ・ロードによって変更されるレベル 0 の値に依存する集約セルが含まれる場合、上位レベルの値は、データ・ロード・プロセスの最後に自動的に更新されます。

集約という用語は、この集約プロセスと、このプロセスの結果として保管される値のセットの両方に使用されます。

集約スクリプト

集約スクリプトのそれぞれは、生成される集約ビューの特定の選択を定義するファイルです。Administration Services の集約設計ウィザードを使用して、集約スクリプトを作成し、それらのスクリプトを.csc 拡張子を持つテキスト・ファイルとしてデータベース・ディレクトリに保管できます。1087 ページの「集約スクリプトの操作」を参照してください。

データベースの集約の実行

集約の実行には、Administration Services の集約設計ウィザードまたは MaxL ステートメントのいずれかを使用できます。集約プロセスには 2 つの段階があります:

- 集約ビューの選択。
- 選択された集約ビューの値の計算と格納。この段階は、集約の生成とも呼ばれます。

集約ビューの選択段階で、様々な組合せの集約ビューの計算と格納が、集約のクエリー応答時間にどのように影響するかが Essbase で分析されます。分析に対する入力として、物理的ストレージとパフォーマンスの要件を定義できます。また、

データの使用状況の追跡や分析プロセスに対する情報の入力も可能です。1085 ページの「使用状況に基づいたビューの選択」を参照してください。

集約ビューの有用性とリソース要件に基づいて、Essbase で集約ビューのリストが作成されます。このリストには、各ビューとともに、その集約ビューとその上にリストされる他のすべての集約ビューの保管時に適用される、ストレージおよびパフォーマンス情報が含まれています。リストされたビューの集約、リストされたビューのサブセットの選択と集約、または別の入力基準による選択プロセスの再実行を選択できます。また、新しい選択基準に基づいた新しいビューの生成を、集約に追加することもできます。1083 ページの「集約ビューの選択の調整」を参照してください。

選択を生成するかどうかにかかわらず、集約ビューの選択は集約スクリプトとして保存できます。集約スクリプトには柔軟性があり、同じ選択が再度必要になった場合に選択プロセスをバイパスできるため、時間を節約できます。1087 ページの「集約スクリプトの操作」を参照してください。

選択プロセスが終了すると、選択された集約ビューは集約を生成するときに計算されます。

次のプロセスに従って集約を定義および生成することをお薦めします:

1. アウトラインの作成または変更後、データ値をロードします。
2. デフォルト集約を実行します。

   オプション: 格納停止ポイントを指定します。
3. 推奨される集約ビューを生成し、デフォルトの選択を集約スクリプトに保存します。
4. 集約が設定されているクエリーのタイプを実行します。
5. クエリー時間や集約時間が長すぎる場合は、集約の微調整を検討してください。1083 ページの「集約ビューの選択の調整」を参照してください。
6. **オプション:** 集約の選択を集約スクリプトとして保存します。

   1087 ページの「集約スクリプトの操作」を参照してください。

➤ データベース集約の選択または生成を実行するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	取得時間を改善するための集約の計算	Oracle Essbase Administration Services Online Help
MaxL	**execute aggregate process** **execute aggregate selection** **execute aggregate build**	『Oracle Essbase テクニカル・リファレンス』

注: 集約設計ウィザードを使用すれば、集約プロセスをバックグラウンドで実行できます。Administration Services のプロセスをバックグラウンドで実行するときは、バックグラウンド・プロセスを実行しながら、同時に他のアクティビティのために管理サービス・コンソールの使用を続けることができます。

1089 ページの「集約のパフォーマンスの最適化」も参照してください。

集約ビューの選択の調整

Essbase で提案される集約ビューのデフォルトの選択では、非常に優れたパフォーマンスが得られます。ただし、選択リスト内のすべての集約ビューを受け入れると、最適なパフォーマンスは保証されません。デフォルトの選択では、Essbase が保管階層を分析し、どの集約セルでも取得される可能性が等しいと仮定します。Essbase では、クエリーの時点で使用可能なメモリーの容量などの外的要因には対応できません。使用可能なメモリーは、取得時のキャッシュ・メモリーの定義や、他の同時プロセスで必要なメモリーなどの要因の影響を受ける可能性があります。

どのデータが最もクエリーされるかを追跡し、その結果と代替ビューを集約ビューの選択プロセスに組み込むことをお薦めします。1085 ページの「使用状況に基づいたビューの選択」を参照してください。

集約を調整してテストする場合は、次の点を考慮してください:

- 取得パフォーマンスの向上を図った結果、ディスク・ストレージのコストと集約の生成にかかる時間が増加することがあります。
- クエリーを追跡した結果として、提案された集約ビューのセットになり、その場合は一部のクエリーが他のクエリーよりも高いパフォーマンスが得られるということがあります。提案された集約ビューを選択すると、クエリー・タイプと頻度が追跡期間中に実行されるクエリーのタイプと頻度に近いかぎり、あるクエリーのパフォーマンス時間は著しく向上し、他のクエリーではほとんど向上が見られない(ただし、悪くなることはありません)ことがあります。
- 集約を最適化するには、調整プロセスを繰り返し行う必要がある場合があります。

集約ビューのサイズを見積るには、`essbase.cfg` の ASOSAMPLESIZEPERCENT 構成設定を使用すると、Essbase で使用されるサンプル・セルの数を指定できます。サンプル・サイズは、入力レベル・データのパーセンテージで指定します。デフォルト(最小)のサンプル・サイズは 100 万(1,000,000)セルです。『Oracle Essbase テクニカル・リファレンス』を参照してください。

Essbase では、データベースに適したバランスの集約ビューを選択して保管する際に役立つ情報が提供されます。この情報を、データベースの取得要件と環境についてわかっている事柄と比較検討してください。次の情報は、集約のための集約ビューの選択に役立ちます:

- ストレージの最大値の必要条件

  集約ビュー選択時には、次の 2 通りの方法でストレージの上限を指定できます:

  - 集約の選択を開始するときには、最大保管停止値を指定します。集約ビューは、指定されたストレージの上限に達するまで、または選択するビューがなくなるまで選択されます。

    Administration Services では、ユーザーが指定する数値はストレージの容量(MB)です。MaxL では、この数値は、レベル 0 の保管される値のサイズの

係数倍です。Oracle Essbase Administration Services Online Help および『Oracle Essbase テクニカル・リファレンス』を参照してください。

- データベースの各分析の後、Essbase はレベル 0 の入力セル・ビューに関する情報と、その後に提案された集約ビューのリストを表示します。集約ビューごとに、その集約ビューが含まれるストレージ番号と、その集約ビューが依存する他のすべての集約ビューが表示されます。集約に含める集約ビューを選択するときは、このストレージ番号を考慮に入れることができます。

- クエリー・コストの相対的なパフォーマンスの向上

  集約ビューごとに表示されるクエリー・コスト番号により、関連する集約ビューから値を取得するための平均取得時間が推定されます。デフォルトのビュー選択では、可能なすべてのクエリーの平均としてのコストが見積られます。クエリーの追跡を使用する場合、見積られるコストは追跡されるすべてのクエリーの平均です。特定の集約ビューのコスト番号は、異なる選択リストでは異なることがあります。たとえば、集約ビュー 0, 0, 1/0, 2/0, 0 は、デフォルトの選択リストでは、分析で追跡したクエリーが含まれる選択に示されるものとは異なるクエリー・コストを示すことがあります。

  改善率をパーセンテージで計算するには、集約ストレージのクエリー・コスト値をレベル 0 の入力セルのみを保存する場合のクエリー・コスト値で除算します。

- 使用状況の追跡

  集約ビューの選択を実行する前に、クエリーの追跡をオンにして、最も頻繁に取得されるデータを確認できます。一定期間のデータベースのアクティビティの後、Essbase で集約分析プロセスに使用状況統計を組み込むことができます。1085 ページの「使用状況に基づいたビューの選択」を参照してください。

- 集約時間

  選択プロセスの完了後の集約実行にかかる時間は、集約ビューが生成されるたびに増加します。実際の集約時間を確認するには、集約を実行する必要があります。

次のプロセスに従って、集約を調整することをお薦めします:

1. 1081 ページの「データベースの集約の実行」で説明しているデフォルトの集約を実行します。
2. 集約スクリプトにデフォルトの選択を保存します。1087 ページの「集約スクリプトの操作」を参照してください。
3. クエリーの追跡をオンにします。1085 ページの「使用状況に基づいたビューの選択」を参照してください。
4. データベースに対して、ユーザーに通常のクエリーを実行してもらいます。または、集約が設定されているデータベースに対してバッチ・クエリーの操作を実行します。すべてのクエリー・ツールによるクエリーが追跡されます。
5. データ取得要件の捕捉に十分な時間が経過したら、追跡したデータを含む別の集約を実行します。

6. 保管候補の集約ビューのリストを分析し、システム・リソースと取得パフォーマンスのバランスが最適な集約ビューを選択します。
7. 選択した集約ビューを生成し、必要に応じて、選択した集約ビューを集約スクリプトに保存します。
8. 集約スクリプトと様々な選択条件についてこのプロセスを繰り返し、環境に最適な集約ビューを選択します。

注： レポートの生成やユーザー・クエリーなど、異なるデータベース取得状況で最適な集約を選択するには、調整プロセスを繰り返し、それぞれの状況に応じた集約スクリプトを作成することが必要となる場合があります。1087 ページの「集約スクリプトの操作」を参照してください。

使用状況に基づいたビューの選択

Essbase では、データベースに対する取得統計を捕捉できます。その後、それらの統計を使用して、自社の取得パターンに合った集約を構築できます。また、Essbase では、使用状況の情報が集約ビューの選択プロセスで使用されるときに、データベースの分析に代替階層も含めます。

定期的なレポートのためのデータベース使用状況は、進行中のユーザー取得の場合とは異なることがあります。様々な取得の状況を最適化するには、状況による使用パターンの追跡と状況ごとの集約スクリプトの作成を検討してください。

集約の選択プロセスを開始する前に、クエリーの追跡がオンになっていて、使用状況の捕捉に十分な時間が経過していることを確認してください。

➤ データ・クエリー・パターンを追跡するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**集約ビュー選択のクエリー・データ追跡**	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』

クエリーの追跡では、クエリー使用状況の情報がメモリー内に保持されます。次のいずれかの操作を実行すると、クエリー使用状況の情報が消去されます。

- データのロードまたは消去
- 集約の生成または消去
- クエリーの追跡のオフ設定

クエリーの追跡は、オフにするか、アプリケーションを停止するか、アウトラインを変更するまでオンのままです。

ユーザー定義ビューの選択の理解

デフォルトでは、Essbase は、内部的なメカニズムを使用して集約の作成方法を決定します。ユーザー定義のビュー選択には、デフォルトのビュー選択とクエリー・データに基づいたビュー選択に影響を与える方法があります。1085 ページの「使用状況に基づいたビューの選択」を参照してください。

管理者は、保管階層にビュー選択プロパティを適用して、Essbase に特定のレベルの集約が選択されないようにすることもできます。

注： セカンダリ階層は、共有または属性階層のいずれかです。

プロパティ	効果
デフォルト	プライマリ階層については、Essbase によってすべてのレベルが検討されます。代替ロールアップが使用可能な場合を除き、セカンダリ階層は集約されません。
すべてのレベルを検討	集約の可能性のある候補として、階層のすべてのレベルが検討されます。これはプライマリ階層ではデフォルトですが、セカンダリ階層ではデフォルトではありません。
集約しない	この階層の集約は行われません。Essbase で選択されるすべてのビューは、入力レベルです。
最下位レベルのみ検討	セカンダリ階層にのみ適用されます。Essbase では、この階層の最下位レベルのみの集約が検討されます。
上位レベルのみ検討	プライマリ階層にのみ適用されます。この階層の最上位レベルのみの集約が検討されます。
中間レベルに集約しない	プライマリ階層に適用されます。最上位レベルと最下位レベルのみを選択します。

注： 最下位レベルの属性次元は、ゼロ レベルの属性メンバーで構成されます。セカンダリ階層が共有メンバーを使用して形成されている場合、最下位レベルは共有メンバーの直属の親を構成します。

Essbase では、選択されたビュー選択プロパティを満たすビューのみが検討されます。

デフォルトのプロパティを変更するには、データベースの主要なクエリー・パターンに精通している必要があります。特定のビューが選択されないようにすると、それらのビューに対するクエリーは遅くなりますが、他のクエリーの速度は上がります。同様に、セカンダリ階層で「すべてのレベルを検討」を有効にすると、その階層に対するクエリーは速くなり、他のクエリーは遅くなることがあります。

➤ ビュー選択プロパティを定義するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	ビュー選択プロパティの設定	Oracle Essbase Administration Services Online Help
MaxL	**execute aggregate process** および **execute aggregate selection**	『Oracle Essbase テクニカル・リファレンス』

クエリー・ヒント

クエリー・ヒントは、実行される可能性のあるクエリーのタイプを Essbase に指定します。たとえば、時間次元の最下位レベルのクエリーを指定するには、1 月のように、最下位レベルの時間で 1 つのメンバーを指定できます。これにより、

Essbase に任意のメンバーが最下位レベルの時間でクエリーされる可能性があることが指定されます。Essbase により、ビューを選択する際のクエリー・ヒントの優先度で指定されたメンバーが指定され、共通のクエリーが最適化されます。

次元にメンバーが指定されていない場合は、その次元のメンバーがクエリーで使用される可能性があることを意味します。たとえば、Sample.Basic で(Sales, 100, East)のクエリー・ヒントを使用することは、レベル 1 の市場のレベル 1 の製品の利益率メジャーが、除外された年次元およびシナリオ次元とは無関係に、共通のタイプのクエリーであることを意味します。

使用状況に基づいたビューの選択は、クエリー・ヒントに優先します。1085 ページの「使用状況に基づいたビューの選択」を参照してください。ユーザー定義ビューの選択は、両者の間に競合がある場合は、クエリー・ヒントに優先します。1085 ページの「ユーザー定義ビューの選択の理解」を参照してください。

アスタリスク「*」は、次元にメンバーが選択されていないことを示します。

ヒント	使用事例
(Jan, *, *, *, *,)	最下位レベルの時間に対する頻繁なクエリー。
(Qtr1, *, *, *, Cola)	大抵の場合、クエリーには、最下位レベルの製品と四半期レベルの時間のメンバーが含まれます。
(Jan, *, *, NY, *) (Qtr1, *, *, NY, *)	「Market」の州レベルでの分析は、「Quarterly」レベル以下であり続ける傾向があります。

同じアウトライン内に多数のヒントが存在可能です。少なくとも 1 つのヒントに適合するビューは、適合するものがないビューよりも重要視されます。

クエリー・ヒントに、動的メンバー、ラベルのみメンバーまたは共有メンバーを含めることはできません。

➤ クエリー・ヒントを適用するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	クエリー・ヒント	Oracle Essbase Administration Services Online Help

集約スクリプトの操作

各集約スクリプトは、データベースに対する特定の集約ビューの選択を表しています。

集約スクリプトによって時間を節約できます。たとえば、新しいデータ値をロードした後、別の集約ビューの選択を実行する必要はありません。集約スクリプトに保管されている選択を使用して集約を生成することで、集約プロセスの速度を上げることができます。

集約スクリプトによって柔軟性を得ることもできます。集約スクリプトには、異なる取得状況ごとに最適化された集約ビューの選択を保存できます。たとえば、月末レポートの取得を最適化するためのスクリプトや、日次レポートの取得を最適化するためのスクリプトを保存して使用できます。

データベースの集約スクリプトに含まれている選択がそのデータベースには無効な場合、そのスクリプトは無効になります。集約スクリプトは、集約の作成時に作成します。集約スクリプト・ファイルは手動で変更しないでください。手動で変更すると、予測できない結果が生じることがあります。集約および集約スクリプトを作成する必要があるタイミングの詳細は、1089 ページの「集約の置換」を参照してください。

集約スクリプトの作成

保存されている集約スクリプトを使用すれば、全集約プロセスを分割できます。集約は、その集約の集約ビューが選択されているときとは異なる時点で生成できます。集約スクリプトには、集約ビューの選択段階で導出される情報が含まれます。

➤ 集約スクリプトを作成するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**取得時間を改善するための集約の計算** **集約設計ウィザード**	Oracle Essbase Administration Services Online Help
MaxL	**query database** **execute aggregate selection**	『Oracle Essbase テクニカル・リファレンス』

集約スクリプトは、拡張子が.csc のテキスト・ファイルとしてデータベース・ディレクトリに保管され、アウトライン内の次元レベルの構造が変更されていないかぎり有効です。集約の選択が有効なタイミングの詳細は、1089 ページの「集約の置換」を参照してください。無効な集約スクリプト・ファイルの潜在的なクラッタを回避するには、集約設計ウィザードを使用します。それらのファイルがもう有用ではない場合は、集約スクリプトを手動で削除します。

集約スクリプトの実行

集約スクリプトを実行すると、その中に指定された集約ビューが生成されます。複数の集約スクリプトを作成できますが、一度に生成できる集約は 1 つのみです。

➤ 集約スクリプトを実行するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**取得時間を改善するための集約の計算** **集約設計ウィザード**	Oracle Essbase Administration Services Online Help
MaxL	**execute aggregate build**	『Oracle Essbase テクニカル・リファレンス』

注： 集約設計ウィザードで既存の集約スクリプトのリストが表示される場合は、データベース・ディレクトリ内の.csc 拡張子を持つすべてのファイルがリストされます。実行できるのは、有効な集約スクリプト・ファイルのみです。

集約のパフォーマンスの最適化

集約のパフォーマンス時間を可能なかぎり改善するため、CALCPARALLEL 構成設定を使用してスレッド数を増やすことができます。各スレッドは集約ビューの構築タスクを取り扱うため、ユーザーが定義するスレッド数で、同時に構築できる集約ビューの数が設定されます。シングル CPU のコンピュータでも、CALCPARALLEL 構成設定を大きくすると、データベースがメモリーには大きすぎるといった、構築する集約が I/O バウンドである場合のパフォーマンスが向上することがあります。スレッドは集約ストレージ・キャッシュを共有する必要があるため、定義されたスレッド数をサポートするには、このキャッシュ・サイズを十分に大きくする必要があります。それ以外の場合は、スレッド数を大きくすると、集約のパフォーマンス時間が低下することがあります。

注： 小規模なデータベースについては、Essbase9.3.1 以降のバージョンにおける集約ビューの構築のパフォーマンスは、9.3.1 より前のバージョンの Essbase よりも遅くなる可能性があります。ただし、数億セルより大きいデータベースについては、Essbase 9.3.1 は、複数のプロセッサを装備し、CALCPARALLEL 構成の設定が適切に選択されているコンピュータ上で特に優れたパフォーマンスを発揮します。

『Oracle Essbase テクニカル・リファレンス』を参照してください。

集約の消去

ディスクからの集約の消去は、手動で行うほうがよい場合もあります。たとえば、ディスク・スペースをディスクの使用度が高い操作に使用できるようにする場合です。集約を消去すると、レベル 0 の値を除くすべてのデータがデータベースから消去され、他の用途で使用できるようにディスク領域が解放されます。集約が消去された後、クエリーにより、レベル 0 の値から動的に取得された値が計算されます。

Essbase サーバーで集約されたデータが自動的に消去されるタイミングについては、表 192 の「データベースの再構築」を参照してください。

➤ 集約を消去するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	**集約ストレージ・データベースからのデータの消去**	Oracle Essbase Administration Services Online Help
MaxL	**alter database**	『Oracle Essbase テクニカル・リファレンス』

集約の置換

集約は、既存の集約を消去し、集約ビューの別の選択を生成することで置換できます。新しい集約ビューの選択および生成プロセスを実行することも、集約スクリプトを実行することもできます。次のような状況では、集約の置換を検討してください:

- 集約ビューの選択を微調整してパフォーマンスを向上させる場合。1083 ページの「集約ビューの選択の調整」を参照してください。
- データベースからデータを取得する状況(レポートの生成やユーザーのクエリーなど)に応じて、最適な集約を作成する場合。
- データベース・サイズが著しく増大した後で集約を最適化する場合。データベースのサイズが徐々に大きくなると、集約の効率が低くなることがあります。パフォーマンスの低下が顕著な場合や、データベース・サイズが元のサイズの約 150%まで増加した場合は、集約の置換を検討してください。
- 使用可能なメモリーやディスク・リソースが変更された場合など、環境の変化にあわせて集約を最適化する場合。

次元内のレベル数が変更されたり、アウトラインに 1 つ以上の次元が追加または削除された場合には、集約および関連する集約スクリプトを置換する必要があります。1081 ページの「データベースの集約の実行」および 1087 ページの「集約スクリプトの操作」を参照してください。

集約ストレージの会計次元のタイム・バランスおよびフロー・メトリック計算の実行

この項のトピックでは、集約ストレージ・データベースでのタイム・バランスおよびフロー・メトリック計算の実行方法について説明します。

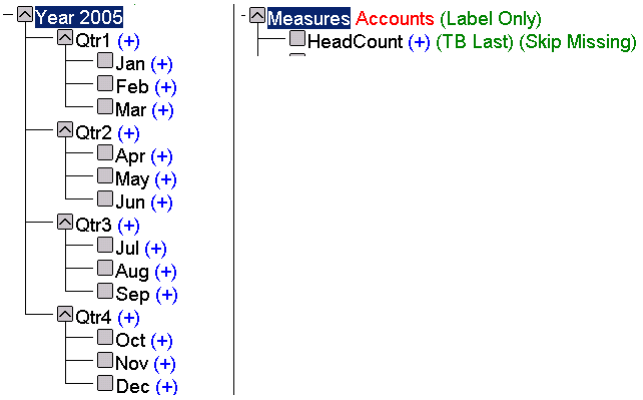
集約ストレージの会計次元でのタイム・バランス・タグの使用

集約ストレージの会計次元にタイム・バランス・プロパティを設定することで、時間次元に沿った組込みの計算を実行できます。これにより、タイムバランス機能を実行するためのメンバー式使用の時間とパフォーマンスのオーバーヘッドが削減されます。

保管済または式を持つ会計次元メンバーでは、次のタイムバランス・プロパティがサポートされています:

- TB 期首、TB 期末、TB 平均
- スキップしない、#Missing のスキップ

人的資源アプリケーションの「Headcount」などの保管済メジャーについて考えてください。年-四半期-月階層では、「Headcount」のデータは月レベルでロードされます。



目的の年ごとまたは四半期ごとの「Headcount」の値は、その月の合計ではありません。そうではなく、期間内に記録された最後の値である必要があります。

SKIPMISSING で「Headcount」に TB 期末のタグを付けることは、「Year 2005」の場合は、その値がその月の従業員数の空でない最後の値であることを意味します。「Dec」に欠落していない「Headcount」値がある場合は、その値が戻されます。そうでない場合は、「Nov」の値がチェックされ、欠落していない場合はこの値が戻されます。

式を持つメンバーにタイム・バランス・タグがある場合、この式はレベル 0 の時間メンバーに対してのみ実行され、時間次元はタイム・バランス・タグに従って集約されます。

タイム・バランス・タグには、時間次元に沿った組込みの計算があります。期間累計やローリング平均など、式を使用するその他の時間ベースの計算を実行するには、TimeView という次元を作成し、その次元にすべての時間ベースの式を記述できます。これを実行することで、他の時間ベースの計算が実行不可になることなく、タイム・バランス計算の機能を使用できます。

集約ストレージの会計次元でのフロー・タグの使用

フロー・タグは、会計次元メンバーを持つ式に割り当てることができます。

次の例で、フロー・メトリックで解決される問題について説明します。12 か月すべてに「売上」と「追加」のデータがあることが前提です。各月の初めの在庫を読み込むために集約を実行しようとしています。

表 208　在庫計算

	売上	追加	在庫
1 月	5	1	50
2 月	6	3	46
3 月	4	2	43
4 月	7	0	41
...			

期首在庫メンバーを計算するために、そのメンバーで MDX 式を使用する場合もあります。フロー・メトリックを使用しないで各月の期首在庫を取得するには、

計算機エンジンで MDX 式を幾何学級数的に繰り返すことが必要な場合があります。

```
   Inventory = SUM(MemberRange(Jan:Time.CurrentMember), (Additions - Sales)) +
Beg_Inventory
```

図で示した例を最適化するには、在庫メンバーに式(**Addition - Sales**)を割り当ててから、メンバーにフローとタグ付けします。

代替階層の制限

代替階層が集約ストレージの時間次元で使用される場合は、会計次元でフロー・タグと TB タグを使用するときに、次の制限が適用されます:

1. 代替時間階層間の共有レベルは、レベル 0 である必要があります。
2. 代替時間階層間の共有レベルでのメンバーの順序は、すべての代替階層内で同じである必要があります。

タイム・バランス・タグが付けられたメジャーの集約

MDX の **Aggregate** 関数は、タイム・バランス・タグが付けられたメジャーの集約に使用できます。『Oracle Essbase テクニカル・リファレンス』を参照してください。

集約ストレージ・データベースのタイム・バランス・メジャーにおける属性計算の影響

次のような場合、次の計算ロジックが適用されます

1. 集約ストレージ・アウトラインに、1 つ以上の属性を持つ日時次元またはリンク属性次元が含まれている。
2. タイム・バランス・タグが付いたメジャーでクエリーを実行する。

上のケースが両方とも当てはまる場合は、セルを評価するために MDX の Aggregate()セマンティクスが使用されます。

たとえば、次のようなシナリオを考えてみます:

- 「Year」は日レベルの階層を持つ日時次元である。
- 「Holiday」は「Year」の属性次元であり、それぞれの日付に Holiday_TRUE または Holiday_FALSE とタグ付けされている。
- 「Opening Inventory」に TBFirst とタグ付けされている。

(Year, Holiday_TRUE, [Opening Inventory])の値は MDX のロジックに従って評価されます:

```
   Aggregate( {Set of dates that are holidays in Year}, [Opening Inventory])
```

集約ストレージ・データの取得

サブトピック

この項のトピックでは、Essbase で集約ストレージ・データベースからデータを取得する方法について説明します。

表 196 も参照してください。

属性計算の取得

集約ストレージ・アプリケーションでサポートされるのは、属性計算次元の合計メンバーのみです。属性計算次元から「Min」や「Avg」などの他のメンバー名を指定すると、エラーが戻されます。177 ページの「属性計算次元の理解」を参照してください。

集約ストレージ・データベースをサポートしている取得ツール

Essbase には、集約ストレージ・データベースからデータを取得できる、次のプログラムおよびツールが用意されています:

- MDX クエリー
- レポート・ライター(管理サービス・コンソールまたは MaxL の **export data using report_file** MaxL ステートメントから実行)

注： ブロック・ストレージ専用の機能(レポート・ライターの SPARSE コマンドなど)をサポートするコマンドは、集約ストレージ・データベースでは使用できません。MDX クエリーでは、集約ストレージの機能が完全にサポートされています。

64 集約ストレージ・データベースでのカスタム計算および割当ての実行

この章の内容

この章の情報は、集約ストレージ・データベースのみに適用されます。

ブロック・ストレージ・データベースにおけるカスタム計算の実行の詳細は、次のトピックを参照してください:

- 第 29 章「ブロック・ストレージ・データベース用の計算スクリプトの作成」
- 第 30 章「ブロック・ストレージ・データベース用の計算スクリプトの例の確認」

ブロック・ストレージ・データベースにおける割当ての実行の詳細は、次のトピックを参照してください:

- 395 ページの「値の割当て」
- 513 ページの「製品をまたがるコストの割当て」
- 514 ページの「次元内での値の割当て」
- 516 ページの「複数の次元での値の割当て」

この章で説明する API と構成設定の詳細は、『Oracle Essbase API リファレンス』と『Oracle Essbase テクニカル・リファレンス』をそれぞれ参照してください。

集約ストレージ・データベースでのカスタム計算および割当ての実行

Essbase は、集約ストレージ・データベースでカスタム計算(1096 ページの「集約ストレージ・データベースでのカスタム計算」を参照)と割当て(1101 ページの「集約ストレージ・データベースでの割当て」を参照)を実行するための、個別の

API 関数および構造を提供します。ただし、どちらの機能にも次の共通する機能性があります:

- 集約ストレージ・データ・ロード・バッファ

  1125 ページの「カスタム計算および割当てのためのデータ・ロード・バッファの理解」を参照してください。
- 貸方および借方処理

  1126 ページの「カスタム計算および割当てのための貸方および借方処理の理解」を参照してください。
- オフセット処理

  1125 ページの「カスタム計算および割当てのためのオフセット処理の理解」を参照してください。
- カスタム計算および割当ての実行で使用される、MDX 式の構文を検証するための API 関数(EssVerifyMDXExpression)。

  『Oracle Essbase API リファレンス』を参照してください。

集約ストレージ・データベースでのカスタム計算

カスタム計算は集約ストレージ・データベースで継続的に発生する計算の実行を有効化することで、Essbase の分析機能を拡張します。

ターゲットのレベル 0 セルを更新する、集約ストレージ・データベースのカスタム計算を記述できます。カスタム計算スクリプトは MDX で表現されます。

カスタム計算を使用することで、総勘定元帳の残高勘定科目における基礎計算を行い、Essbase 集約ストレージ・データベースのターゲットのレベル 0 メンバーに結果を書き込むことができます。残高勘定科目または固定金額で計算を実行し、すべての会計期間で繰り返すようにスケジューリングできます。

集約ストレージ・データベースでのカスタム計算は、データベースが総勘定元帳レポートに使用される場合に有用で、ここでは複式簿記が有効になります。資産および支出などの借方項目は、株式および収益などの貸方項目と貸借一致する必要があります。

カスタム計算を作成して実行するには、次のワークフローを使用します:

- MDX で表現される計算スクリプトを作成します。

  1098 ページの「カスタム計算の記述」を参照してください。
- 計算が実行されるデータベース内の領域を選択します。ターゲットおよび POV(視点)のパラメータを使用して、領域を実行時に指定します。
- 借方および貸方処理を使用する場合、アウトラインの借方および貸方メンバーを選択して正の値と負の値を記述します。これらのパラメータを実行時に指定します。

  1126 ページの「カスタム計算および割当てのための貸方および借方処理の理解」を参照してください。

- オフセット項目を使用する場合、オフセット項目を作成する領域を選択します。MDX タプルを使用して、このパラメータを実行時に指定します。オフセットが指定されていない、または空の場合、オフセット計算は実行されません。

  注： 総勘定元帳の記帳では、オフセット項目は元帳の反対側の相殺メジャーとなります。たとえば、January の$100 という貸方では、元帳の借方側に$100 のオフセットが追加される可能性があるため、元帳はその金額の今後の支出への準備として貸借一致する場合があります。

  1125 ページの「カスタム計算および割当てのためのオフセット処理の理解」を参照してください。

- Essbase API を使用してカスタム計算スクリプトを実行します。

  1098 ページの「カスタム計算の実行」を参照してください。

カスタム計算基準のリスト

カスタム計算機能は指定された様々な基準に基づいています。表 209 に、カスタム計算の定義に使用される基準の簡単な説明を示します。次に進む前にこれらの用語を確認してください。

表 209　カスタム計算基準の説明

基準	説明
POV	カスタム計算が実行されるコンテキストを記述する、データベース内の対称領域。 属性メンバーは、この引数では使用できません。
計算スクリプト	MDX で表現される計算スクリプト。 属性メンバーは、式の左側では使用できません。
ターゲット	MDX で表現されるタプル引数は、計算結果が書き込まれるデータベース内の領域を定義します。この引数は各式の左側およびオフセットと組み合せることで、結果およびオフセット値が書き込まれる場所を決定します。 属性メンバーは、この引数では使用できません。
(オプション) オフセット	各ソース金額のオフセット値が書き込まれる、データベース内の場所。 属性メンバーは、この引数では使用できません。
(オプション) 貸方および借方メンバー	複式簿記では、1 つの取引に対する仕訳の貸借一致。両方とも MDX メンバー式です。借方メンバーは正の結果値が書き込まれるメンバーを示し、貸方メンバーは負の結果値が書き込まれるメンバーを示します。 属性メンバーは、この引数では使用できません。
ソース領域	スクリプト内の式によって参照される、キューブの領域を指定する MDX セット式。

カスタム計算の記述

カスタム計算スクリプトは、.csc 拡張子を付けて作成および格納するファイルです。MDX の 1 つまたは一連のタプル表現ペアでカスタム計算スクリプトを作成し、セミコロンで終了します。構文は次のとおりです:

```
tuple := numeric_value_expression;
```

tuple は 1 つ以上のメンバーの MDX 指定で、その中のメンバー同士は同じ次元に属すことができません。タプルは等式の左側にあり、カスタム計算の結果が書き込まれる場所を判断する主要要因となります。

タプル式ではメンバー名のみを使用できます。カスタム計算スクリプトでは、メンバー関数の使用はサポートされていません。

注： 結果のターゲットを判断する二次要因はターゲット・パラメータで、三次要因は POV パラメータです。2 番目および 3 番目のパラメータは、計算スクリプトの要素としてではなく、計算の実行時に指定します。

numeric_value_expression は、数字や算術演算子などの簡単な MDX 数値式です。この表現は等式の右側にある必要があります。算術演算子のみが実行されます。算術演算子(AND、OR または IF ステートメントなど)が使用されていない場合は、エラーが返されます。

数値式ではメンバー名を使用できますが、カスタム計算スクリプトではメンバー関数の使用はサポートされていません。

属性メンバーは、カスタム計算スクリプトの等式の左側で使用できません。

ソース領域も定義する必要があり、これが Essbase のパフォーマンス・ヒントとして機能します。Essbase は、ソース領域に指定されたデータをプリフェッチし、それを使用してスクリプトで指定されている計算を実行します。

カスタム計算スクリプトの例、およびソース領域の定義の詳細は、1099 ページの「カスタム計算の使用例」を参照してください。

カスタム計算の実行

カスタム計算は、MaxL の **execute calculation** ステートメントを使用して実行できます(集約ストレージ・バージョン)。

Essbase API を使用してカスタム計算スクリプトを実行できます。EssPerformCustomCalcASO を使用して、集約ストレージ・データベースでカスタム計算を実行または確認します。カスタム計算の情報を ESS_PERF_CUSTCALC_T 構造体に指定します。

また、Oracle Hyperion Calculation Manager を使用することで、カスタム計算を設計し、それを実行のため Enterprise Scheduling Services に配置できます。

カスタム計算の使用例

次のような次元のアウトラインを想定します:

- **Company:** CompanyA、CompanyB、およびその他の子が含まれます。
- **Department:** 101、102、103 などの番号付けされた部門が含まれます。
- **Account:** 勘定科目 5740 が賃貸料の勘定科目で、SQFT が各部門の平方フィート数を記録するために使用される統計勘定科目です。
- **Scenario:** Actual メンバーにはデータが転記され、Allocation メンバーには割当ておよびカスタム計算が格納されます。Scenario メンバーは、Actual および Allocation の子メンバーを集約する親です。
- **Year:** 月および四半期で構成される時間次元。
- **Geography:** 州および都市で構成される次元。
- **AmountType:** Debit がターゲットで、Credit がオフセットです。
- **Project:** Proj1、Proj2 などのプロジェクトが含まれる次元。

POV は MDX セット式で、カスタム計算が実行される場所を示します。次のように指定されます:

```
CrossJoin( { ( [Company], [101], [Jan], [Scenario] ) },
  Descendants( Geography, Geography.Levels(0)) )
```

DebitMember は MDX メンバー式で、正の結果値が書き込まれる対象の借方メンバーを示します。`[BeginningBalance_Debit]`のように指定されます。

CreditMember は MDX メンバー式で、負の結果値およびオフセット結果値が書き込まれる対象の貸方メンバーを示します。`[BeginningBalance_Credit]`のように指定されます。

注： オフセットは、すべての結果値の合計が負数の場合は借方メンバーに書き込まれます。

オフセットは MDX タプル式で、オフセット項目が作成される場所を示します。`([Account_NA], [Project_NA])`のように指定されます。

オフセット式はターゲットおよび POV と組み合せることで、オフセット項目が作成される場所を決定します。次元が重複する場合、オフセットの場所を解決するための順序はオフセット、ターゲット、POV の順序になります。

ターゲットは MDX タプル式で、カスタム計算の結果を書き込む場所を示します。`(Allocation)`のように指定されます。

ターゲット式は POV と、カスタム計算スクリプト内の各行の左側と組み合せることで、結果が書き込まれる場所を決定します。次元が重複する場合、ターゲットの場所を解決するための順序は等式の左側、ターゲット、POV の順序になりま

す。この例では、POV で指定された Scenario メンバーよりターゲットが優先されるため、結果は Allocation メンバーに書き込まれます。

カスタム計算スクリプトの例を次に示します:

```
   (AccountA,Proj1) := 100;
([AccountB], [Proj1]) := ([AccountB], [Proj1]) * 1.1;
(AccountC,Proj1) :=
    ((AccountB,Proj1,2007) + (AccountB, Proj1)) / 2;
(AccountA,Proj2) :=
    ((AccountD,Proj1) +
     (AccountB,Proj2)) / 2;
```

POV の各組合せの場合、

- 現在の POV の組合せのコンテキストで、計算スクリプトが実行されます。
- 1 つのオフセット値がターゲットの場所に書き込まれます。

注： 各式(計算スクリプト内の行)は順番にではなく、同時に実行されます。したがって、1 つの式の結果は次に続く式では使用できません。

ソース領域を定義するには、カスタム計算スクリプトを確認し、等式の右辺でどのメンバーが参照されるかを決定します。ソース領域には、少なくとも、カスタム計算スクリプトの割当てステートメントの右辺にあるすべてのメンバーが含まれる必要があります。

ソース領域を単一の MDX セットとして定義します。等式の右辺のメンバーが複数のディメンションに属している場合は、CrossJoin を使用して 2 つのセットからこのセットを作成できます。CrossJoin では 2 つのセットのみが受け入れられるため、ネストした CrossJoin の使用が必要な場合があります。

上記のカスタム計算スクリプトのソース領域は次のとおりです:

```
Crossjoin(
{[AccountB], [AccountD]},
 Crossjoin(
  {[Proj1], [Proj2]}, {[2007]}
 )
)
```

スクリプトに割り当てられていないソース領域にメンバーを含める必要はありません。たとえば、スクリプトで使用されていないソース領域に`[AccountC]`を追加した場合は、無視されて、パフォーマンスがわずかに低下する場合があります。

ソース領域の数値を考慮する必要はありません。たとえば、カスタム計算スクリプト内の次の割当てでは、ソース領域に何も追加す必要はありません: `([Bud Var]):=10`

集約ストレージ・データベースでの割当て

割当ては、収益またはコストを配分する予算設定プロセスで使用されます。

割当て機能では、指定されたソース金額を集約ストレージ・データベース内のターゲット範囲のセルに割当てできます。指定された基準に基づいてソース金額をターゲットに比例的に割り当てることも、ソース金額をターゲットに均等に分散することも可能です。

MaxL の **execute allocation** ステートメントを使用して集約ストレージ割当てを実行できます。

また、集約ストレージ割当ては、EssPerformAllocationAso API 関数を使用し、ESS_PERF_ALLOC_T API 構造で割当てに関する情報を指定することで実行することもできます。

単一割当ては、EssPerformAllocationAso API の単一呼出しに対応しており、固有の POV (視点)、範囲、金額、基準、ターゲット、およびオフセット(オプション)があります。これらのパラメータの異なる値のセットで割当てを実行するには、個別の API を順次呼び出す必要があります。

割当ては割当てエンジンで実行されてから、Essbase で作成された一時データ・ロード・バッファを使用することで、集約ストレージ・データベースに戻って書き込まれます。データ・ロード・バッファの一般的な情報は、1057 ページの「集約ストレージ・データベースへのデータのロード」を参照してください。割当ておよびカスタム計算に固有の情報は、1125 ページの「カスタム計算および割当てのためのデータ・ロード・バッファの理解」を参照してください。

割当て基準のリスト

割当ては指定された様々な基準に基づいています。表 210 に、割当ての定義に使用される基準の簡単な説明を示します。次に進む前にこれらの用語を確認してください。

表 210　割当て基準の説明

基準	説明	参照
POV	割当てが実行されるコンテキストを記述する、データベース内の対称領域	1104 ページの「POV の設定」
範囲	割当て値が計算されて書き込まれる、データベース内の対称領域	1104 ページの「範囲の設定」
(オプション)除外範囲	割当て値を書き込まない、範囲内の場所	1104 ページの「範囲の設定」
金額	割り当てられる金額	1105 ページの「金額の設定」
(オプション)金額コンテキスト	金額の追加コンテキストまたは特殊性	1105 ページの「金額の設定」
(オプション)金額の時間スパン	金額に想定される 1 つ以上の期間	1105 ページの「金額の設定」

基準	説明	参照
(オプション)ゼロの金額オプション	ゼロまたは#MISSING の金額値の処理	1105 ページの「金額の設定」
基準	範囲と組み合せることで、金額の割当て方法を決定する基準値の場所を定義します	1108 ページの「基準の設定」
(オプション)基準の時間スパン	基準に想定される 1 つ以上の期間	1108 ページの「基準の設定」
(基準の時間スパンが設定されている場合は必須) 基準の時間スパン・オプション	基準の時間スパン全体で基準を計算するためのメソッド: ● 結合 ● 分割	1108 ページの「基準の設定」
ゼロの基準オプション	ゼロの基準値の処理	1108 ページの「基準の設定」
(オプション)負の基準オプション	負の基準値の処理	1108 ページの「基準の設定」
ターゲット	範囲と組み合せることで、割当て値が書き込まれるデータベース内の領域を定義します	1109 ページの「ターゲットの設定」
(オプション)ターゲットの時間スパン	ターゲットに想定される 1 つ以上の期間	1109 ページの「ターゲットの設定」
(ターゲットの時間スパンが設定されている場合は必須) ターゲットの時間スパン・オプション	ターゲットの時間スパン全体で値を割り当てるためのメソッド: ● 金額の除算 ● 金額の繰返し	1109 ページの「ターゲットの設定」
割当てメソッド	金額を割り当てるためのメソッド: ● シェア: 金額を基準値に比例的に割り当てます ● 分散: 金額を均等に割り当てます	1109 ページの「割当てメソッドの設定」
(オプション)分散スキップ・オプション	分散割当てメソッドの場合、ゼロ、#MISSING または負の範囲にある基準値をスキップするかどうか	1109 ページの「割当てメソッドの設定」
丸めメソッド	割当て値を丸めるかどうか。 丸めることを選択した場合、丸め誤差を処理するメソッドを指定します: ● 丸め誤差の無視 ● 丸め誤差合計を追加する対象: ○ 最大割当て値 ○ 最小割当て値 ○ 特定の場所	1112 ページの「丸めメソッドの設定」
(オプション)丸め桁数	割当て値が丸められる小数点以下の桁数: ● 最も近い整数 ● 指定した小数点以下の桁数 ● 10 の累乗	1112 ページの「丸めメソッドの設定」

基準	説明	参照
(丸めメソッドが特定の場所に設定されている場合は必須) 丸め先ロケーション	丸め誤差合計を追加する対象の場所	1112 ページの「丸めメソッドの設定」
(オプション)オフセット	各ソース金額のオフセット値が書き込まれる、データベース内の場所	1113 ページの「オフセットの設定」
(オプション)貸方および借方メンバー	複式簿記では、1 つの取引に対する仕訳の貸借一致	1113 ページの「割当ての均衡化」

領域の理解

Essbase は割当ての実行時に、データベースの各種の領域を使用します。各領域は、領域で定義された各次元の 1 つ以上のメンバーで構成されます。

表 211　割当てで使用される領域のリスト

領域名	領域定義	説明
ソース	(POV X amount [X amount context] X [amount time span])	割り当てられる金額値が含まれる領域。 ソース領域とターゲット領域は重複できません。
ターゲット	(POV X target X debit member/credit member X range X [target time span])	割当て値が書き込まれる場所が含まれる領域。 ソース領域とターゲット領域は重複できません。 ターゲット領域は割当ての実行前に空である必要はありません。Essbase は空でないセルを割当てデータまたはゼロで上書きします。#MISSING というセルの場合、Essbase が割当てデータをこのセルに書き込まないかぎり、セルは#MISSING のままになります。
基準	(POV X basis X range X [basis time span])	ソース金額の割当て方法の決定に使用される基準値が含まれる領域。 基準は POV の部分を上書きする場合があります。
オフセット	(POV X offset X debitMember/ creditMember)	オフセット値が書き込まれる場所が含まれる領域。

割当て基準の指定

割当てパラメータ値は、次の方法で表記できます:

- MDX メンバー式
- MDX セット式
- MDX タプル式(同じ次元からの 2 つのメンバーは存在できない)
- 定数

割当てパラメータ値の表記方法の詳細は、『Oracle Essbase API リファレンス』を参照してください。

共有メンバーの使用

共有メンバーが割当てパラメータで指定されていると、Essbase は割当てを実行する前に、共有メンバーをその元のメンバーにマッピングします。

重複メンバーの使用

メンバー名が繰り返されている、またはメンバーおよびその共有メンバーが両方指定されているなどの理由で、重複メンバーが割当てパラメータで指定されていると、Essbase は重複メンバーを削除して警告を発行します。

POV の設定

POV は、割当てが実行されるコンテキストを記述するデータベース内の対称領域を指定します。POV はレベル 0 メンバーのみで構成できます。POV で定義された次元は他のパラメータで使用できませんが、基準および基準の時間スパンは例外です。

割当ては、POV セット内のあらゆるメンバーの組合せに対して繰り返されます。POV の組合せの数は、複数のメンバーが存在する次元のメンバー数の積です。(メンバーが 1 つだけの次元は、組合せの数の計算に使用されません。)

たとえば、POV が 2 つの次元(CostCenter および Project)で構成され、割当てが 2 つのコスト・センター(CostCenter1 および CostCenter2)と 3 つのプロジェクト(Project1、Project2 および Project3)で行われると仮定します。POV の組合せの数は 6 になります:

```
   Project1,CostCenter1
Project1,CostCenter2
Project2,CostCenter1
Project2,CostCenter2
Project3,CostCenter1
Project3,CostCenter2
```

割当ては、割当てコンテキストが各組合せに正常に設定されるまで 6 回繰り返されます。

割当ての基準として想定される値は POV の組合せによって異なります。1108 ページの「基準の設定」を参照してください。

注： 期間が POV で指定されている場合、金額の時間スパンとターゲットの時間スパンの各オプションは使用できません。1105 ページの「金額の設定」と 1109 ページの「ターゲットの設定」をそれぞれ参照してください。

範囲の設定

範囲は、割当て値が計算されて書き込まれる、データベース内の対称領域を指定します。

割当て値を範囲内の特定のセルに書き込まない場合は、除外範囲パラメータを使用して範囲の対称サブセットを表記します。範囲のサブセットを除外している場合でも、Essbase は範囲内のすべてのセルを使用して割当て値を計算します。

範囲からセルを除外すると、割当て値の合計が金額の値を下回る場合があります。

次の例では、範囲が 6 つのメンバーの組合せで構成され、金額が 6 で、割当てメソッドが分散のため、Essbase が範囲全体で金額を均等に割り当てます。割当て分散金額は 1 です(6/6 = 1)。

表 212 に示すように、Essbase は範囲の各セルに 1 を書き込みます。

表 212　例: 範囲の各メンバーに対する金額の割当て

	CostCtr1	CostCtr2
Project1	1	1
Project2	1	1
Project3	1	1

表 213 に示すように、除外された範囲が(Project2,CostCtr2)のメンバーの組合せに設定されている場合、Essbase は割当て分散金額をそのセルに書き込みません。したがって、割当て値の合計(5)は金額(6)を下回ります。割当てプロセス後の除外されたセルの値は、#MISSING またはゼロになります。

表 213　例: 範囲の一部のメンバーのみに対する金額の割当て

	CostCtr1	CostCtr2
Project1	1	1
Project2	1	
Project3	1	1

範囲と除外された範囲は、レベル 0 のメンバーのみで構成できます。

金額の設定

金額は割当てのソースを指定します。ターゲット領域のセルに金額値が割り当てられます。金額は、上位レベル・メンバーまたはレベル 0 メンバーで構成が可能で、数値式、タプルまたは定数として表すことができます。

金額を表す式によって特定の要件が決まります:

- 数値式:
  - 式の全メンバーが同じ次元に属している必要があります。
  - タプルは式で使用できません。
  - 演算式(+、-、/および*)のみを式で使用できます。
  - MDX 関数(Avg や Parent など)は使用できません。

  例:

```
(Acc_1000 + Acc_2000)/2
```

```
AccA + AcctB
```

```
Balance * 1.1
```

- タプル:
  - タプルは、POV で指定されていないすべての次元の 1 つのメンバーを使用する必要があります。
  - 金額コンテキストを空にする必要があります。

  例:

```
(Balance,Cost_Center_00,Project_00)
```

```
(Balance,Cost_Center_00,Actual)
```

- 定数:
  - 金額コンテキストを空にする必要があります。
  - 金額の時間スパンを空にする必要があります。

  例:

```
100
```

金額を詳細に定義するためにこれらのパラメータを使用できます:

- (**オプション**)金額コンテキストは、金額の追加的なコンテキストまたは特殊性を指定します。金額コンテキストは、上位レベル・メンバーまたはレベル 0 メンバーで構成が可能で、タプルとして表すことができます。金額コンテキストを指定することで、POV で指定されていない次元のメンバーを含めることができます。

  金額コンテキストを使用する際、これらの要件が金額および金額コンテキストに適用されます:

  - パラメータは同じ次元内のメンバーを参照できません。
  - パラメータはすべて、POV で指定されていないすべての次元のメンバーを使用する必要があります。
- (**オプション**)金額の時間スパンは、金額に想定される 1 つ以上の期間を指定します。指定された期間の金額値が集約され、集約された金額値が割り当てられます。期間は時間次元のレベル 0 メンバーである必要があります。

金額が演算式を使用して指定され、金額の時間スパンが使用される際、金額の時間スパンは、金額内のあらゆる式、または金額で使用されるあらゆる式メンバーよりも優先されます。たとえば、表 214 に示すように、金額が Dept_A/Dept_B として指定され、金額の時間スパンが各部門について Jan、Feb、Mar および Aprに設定されると仮定します。POV に対して割り当てられる金額は、Dept_A の金額時間スパン値(10)を Dept_B の金額の時間スパン値(20)で割ることで計算され、0.5 になります。

表 214　例: 金額の時間スパンは金額内の式よりも優先される

金額の時間スパンのメンバー	Dept_A	Dept_B
Jan	1	2
Feb	2	4
Mar	3	6
Apr	4	8
合計	10	20

- (**オプション**)ゼロの金額オプションは、値がゼロまたは#MISSING の場合に金額を処理する方法を指定します。ゼロ値の割当て(デフォルト)、次のゼロ以外または#MISSING 以外の金額値へのスキップ、または割当て操作全体の取消しを選択できます。

次の例に示すように、金額コンテキストと金額の時間スパンを使用して同じ結果を導き出すことができます。金額は Dept_A の値ですが、金額の時間スパンは Dept_A の Jan、Feb、Mar および Apr の月のみに焦点を当てるために使用されます。表 215 に示すように、金額の時間スパンに含まれるメンバーの集約値(10)は、範囲内のセル全体に割り当てられる金額値です。

表 215　例: 金額の時間スパン

金額の時間スパンのメンバー	Dept_A
Jan	1
Feb	2
Mar	3
Apr	4
合計	10

表 216 に示すように、Jan + Feb + Mar + Apr の演算式として金額を指定し、金額コンテキストを Dept_A として設定することで、同じ金額値を導き出すことができます:

表 216　例: 金額コンテキスト

金額コンテキスト	Jan	Feb	3 月	4 月	合計
Dept_A	1	2	3	4	10

基準の設定

(**オプション**)基準は、範囲と組み合せることで、金額の割当て方法を決定する基準値の場所を定義します。基準は上位レベル・メンバーまたはレベル 0 メンバーで構成できます。

基準を詳細に定義するためにこれらのパラメータを使用できます:

- (**オプション**)基準の時間スパンは、基準に想定される 1 つ以上の期間を指定します。期間は時間次元のレベル 0 メンバーである必要があります。
- (基準の時間スパンが設定されている場合は必須)基準の時間スパン・オプションは、基準の時間スパンで指定された期間に対して、基準が計算される方法を指定します。各期間で個別に基準値を使用する(分割)か、または基準の時間スパンで指定された期間全体の基準値の合計を使用する(結合)ことを選択できます。
  - 基準の時間スパンが複数の期間を指定し、ターゲットの時間スパンが 1 つの期間を指定する、または空の場合、基準の時間スパン・オプションを結合に設定する必要があります。Essbase はターゲットの時間スパン・オプションを無視します。
  - 基準の時間スパンおよびターゲットの時間スパンが複数の期間を指定し、基準の時間スパン・オプションを分割に設定する場合、基準の時間スパンおよびターゲットの時間スパンで指定される期間は同一である必要があります。Essbase はターゲットの時間スパン・オプションを無視します。

  1113 ページの「基準およびターゲットの時間スパンの設定の理解」を参照してください。
- ゼロの基準オプションは、ゼロの基準値を処理する方法を指定します。次のゼロ以外または#MISSING 以外の金額値へのスキップ、または割当て操作全体の取消しを選択できます。Essbase は、割当てメソッドに基づいてゼロの基準オプションを処理します。1109 ページの「割当てメソッドの設定」を参照してください。
- (**オプション**)負の基準オプションは、負の基準値を処理する方法を指定します。負の基準オプションに使用できるオプションは、使用される割当てメソッドによって異なります。1109 ページの「割当てメソッドの設定」を参照してください。

注：　基準は、分散割当てメソッドを使用中で、どの分割スキップ・オプションも設定していない場合は無視されます。

ターゲットの設定

ターゲットは、範囲と組み合せることで、割当て値が書き込まれるデータベース内の領域を定義します。ターゲットはレベル 0 メンバーのみで構成できます。

ターゲットを詳細に定義するためにこれらのパラメータを使用できます:

- (**オプション**)ターゲットの時間スパンは、ターゲットに想定される 1 つ以上の期間を指定します。期間は時間次元のレベル 0 メンバーである必要があります。
- (ターゲットの時間スパンが設定されている場合は必須) ターゲットの時間スパン・オプションは、ターゲットの時間スパンで指定された期間全体で値を割り当てるメソッドを指定します。金額値を除算するか、または指定された期間全体で金額値を繰り返すことを選択できます。
    - 基準の時間スパンが複数の期間を指定し、ターゲットの時間スパンが 1 つの期間を指定する、または空の場合、基準の時間スパン・オプションを結合に設定する必要があります。Essbase はターゲットの時間スパン・オプションを無視します。
    - 基準の時間スパンおよびターゲットの時間スパンが複数の期間を指定し、基準の時間スパン・オプションを分割に設定する場合、基準の時間スパンおよびターゲットの時間スパンで指定される期間は同一である必要があります。Essbase はターゲットの時間スパン・オプションを無視します。

1113 ページの「基準およびターゲットの時間スパンの設定の理解」を参照してください。

割当てメソッドの設定

割当てメソッドは、金額を均等に割り当てるか、比例的に割り当てるかを指定します。

- シェア・メソッドは、範囲の現在のメンバーの基準値(basis_mbr_value)を範囲全体の基準の合計(basis_range_sum)で除算することで、ある割合の金額(alloc_share_amt)を割り当てます。割当て金額は、範囲の有効な基準値の数値に基づきます。割当てシェア金額を計算するためのアルゴリズムは次のとおりです:

    ```
    alloc_share_amt = (basis_mbr_value/basis_range_sum) * amount
    ```

    基準値と Essbase のアクション:

    - ゼロ: Essbase はゼロを対応するターゲット・セルに書き込みます。

      すべての基準値の合計がゼロの場合(division-by-zero エラーが発生する可能性があります)、Essbase はゼロの基準オプション設定を使用します。1108 ページの「基準の設定」を参照してください。
    - #MISSING: Essbase はターゲット・セルを#MISSING のままにするか、ターゲット・セルにすでに値がある場合は、既存の値をゼロで上書きします。

- 負数: Essbase は負の基準オプション設定を使用します。負の基準値の使用(デフォルト)、次の金額値へのスキップ(データは現在の金額値に割り当てられません。Essbase は次の POV の組合せにスキップします)、または操作全体の取消しを選択できます。

次の例はシェア割当てメソッドを示しています。両方の例とも、割り当てる金額は 10 です。

表 217 では、金額(10)はビルの賃貸料を表し、基準が範囲内の各部門の人数を表すと仮定します。Essbase は、#MISSING 以外の人数を示す部門(Dept_A から Dept_D)の基準値を使用して、割当てシェア金額を計算し、これが賃貸料割当てになります。

Dept_A の賃貸料割当ては、Dept_A の基準値(3)が、範囲全体の有効な基準値の合計(3 + 2 = 5)で除算され、金額(10)で乗算されたものです: 3/5 * 10 = 6。Dept_D の場合、賃貸料割当ては 2/5 * 10 = 4 です。範囲内のターゲット・セルの合計は 10 と等しくなります。

表 217　シェア割当てメソッドの例

範囲のメンバー	基準(人数)	ターゲット(賃貸料割当て)
Dept_A	3	6
Dept_B		
Dept_C	0	0
Dept_D	2	4

表 218 では、すべての基準値がシェア割当て金額の計算に想定されると仮定します。Mbr1 の割当ては、Mbr1 の基準値(3)が、範囲全体の有効な基準値の合計(3 + -1 + 2 = 4)で除算され、金額(10)で乗算されたものです: 3/4 * 10 = 7.5。Mbr3 の場合、割当ては-1/4 * 10 = -2.5 です。Mbr4 の場合、割当ては 2/4 *10 = 5 です。範囲内のターゲット・セルの合計は 10 と等しくなります。

表 218　シェア割当てメソッドの例: 負の基準オプション - デフォルト

範囲のメンバー	基準	ターゲット
Mbr1	3	7.5
Mbr2	#MISSING	
Mbr3	-1	-2.5
Mbr4	2	5.0

- 分散メソッドは、金額を範囲全体で均等に割り当てます(`alloc_spread_amt`)。金額の除算に使用される数値、すなわち、割当て分散金額が書き込まれる予定のターゲット・セルの数は、範囲内の有効な基準値の合計(`#_valid_basis_values`)に基づきます。割当て分散金額を計算するためのアルゴリズムは次のとおりです:

```
alloc_spread_amt = amount/#_valid_basis_values
```

分散割当てメソッドを使用する際、オプションで分散スキップ・オプション・パラメータを使用すると、範囲内のゼロ、#MISSING または負数の基準値をすべてスキップできます。複数のオプションを指定できます。

基準値と Essbase のアクション:

- ゼロ: Essbase はゼロを対応するターゲット・セルに書き込みます。

  分散スキップ・オプションがゼロをスキップするように設定されている場合、データは割り当てられません。

- #MISSING: Essbase はターゲット・セルを#MISSING のままにするか、ターゲット・セルにすでに値がある場合は、Essbase は既存の値をゼロで上書きします。

  分散スキップ・オプションが#MISSING をスキップするように設定されている場合、データは割り当てられません。

- 負数: Essbase は負の基準オプション設定を使用します(これは、負数をスキップする分散スキップ・オプション設定よりも優先されます)。次のいずれかのアクションを選択できます:
  - 負の基準値の使用(デフォルト)
  - 次の金額値へのスキップ(データは現在の金額値に割り当てられません)
  - 負数の絶対値の使用
  - $MISSING として負数を処理(値はターゲット・セルに割り当てられません)
  - ゼロとして負数を処理(ゼロがターゲット・セルに割り当てられます)
  - 操作全体の取消し

すべての基準値がスキップされている場合(割当てゼロの分母を形成します)、Essbase はゼロ基準オプション設定を使用します。1108 ページの「基準の設定」を参照してください。

次の例は分散割当てメソッドを示しています。両方の例とも、割り当てる金額は 10 です。

表 219 では、分散スキップ・オプション・パラメータが指定されていないと仮定します。したがって、Essbase は範囲内の 4 つすべての基準メンバーを想定します。Essbase は金額(10)を、範囲内の有効な基準メンバーの数(4)で除算し、その値(2.5)を範囲内の各ターゲット・セルに分散します: 10/4 = 2.5。

表 219　分散割当てメソッドの例: 基準値をスキップしない

範囲のメンバー	基準	ターゲット
Mbr1	2	2.5
Mbr2	#MISSING	2.5

範囲のメンバー	基準	ターゲット
Mbr3	3	2.5
Mbr4	-6	2.5

表 220 では、分散スキップ・オプション・パラメータが#MISSING および負数を無視するように設定されていると仮定します。したがって、Essbase は正の値を持つ 2 つの基準メンバー(Mbr1 および Mbr3)のみを想定します。Essbase は金額(10)を、範囲内の有効な基準メンバーの数(2)で除算し、その値(5)を Mbr1 および Mbr3 のターゲット・セルに分散します: 10/2 = 5。

表 220　分散割当てメソッドの例: #MISSING および負の基準値のスキップ

範囲のメンバー	基準	ターゲット
Mbr1	2	5
Mbr2	#MISSING	
Mbr3	3	5
Mbr4	-6	

丸めメソッドの設定

丸めメソッドは、割当て値を丸めるかどうかを指定します(デフォルトは、丸めない)。

値を丸めることを選択した場合、丸めメソッドは丸め誤差の処理方法を指定します。丸め誤差を無視するか、すべての割当て値を丸めて丸め誤差合計を最大割当て値、最小割当て値または特定のセルに追加することを選択できます。丸め誤差を最大割当て値または最小割当て値に追加することを選択し、複数の最大割当て値または最小割当て値が存在する場合、Essbase は丸め誤差を追加する対象となる、最大値または最小値の 1 つを選択します。

割当て値を丸めることを選択した場合、丸めメソッドを詳細に定義するためにこれらのパラメータを使用できます:

- **(割当て値を丸める場合は必須)**丸め桁数は、割当て値が丸められる小数点以下桁数を指定します。最も近い整数(デフォルト)、指定した小数点以下の桁数、または 10 の累乗に丸めることを選択できます。

  丸め桁数は-100 から 100 までの数値で、整数、MDX 数値式またはタプルとして表現できます。

  丸め桁数の設定が割当て値の通貨に基づく場合は、MDX 数値式の使用が便利です。たとえばデータベースに、POV の一部である Currency という次元と、割当て値の通貨に基づいて割当て値を丸める方法を指定する、NumCurrencyDigits という関連付けられた属性次元が含まれると仮定します。丸め桁数を次のように表現できます:

  ```
  Currency
  ```

```
.currentMember.
NumCurrencyDigits
```

注： 丸めメソッドに、割当て値を丸めないことを選択した場合、丸め桁数のパラメータは空である必要があります。

- **(丸めメソッドが特定の場所に設定されている場合は必須)**丸め先ロケーションは、丸め誤差合計を追加する対象のセルを指定します。タプルとして表現され、セルは範囲内にあり範囲と同じ次元性を持つ必要があります。丸め先ロケーションはレベル 0 のメンバーのみで構成できます。

注： 丸めメソッドに、特定のセルに丸めること以外のオプションを選択した場合、丸め先ロケーションのパラメータは空である必要があります。

オフセットの設定

(オプション)オフセットは、各ソース金額のオフセット値が書き込まれる、データベース内の場所を指定します。

オフセットは割当ておよびカスタム計算で同様に動作します。1125 ページの「カスタム計算および割当てのためのオフセット処理の理解」を参照してください。

割当ての均衡化

(オプション) debitMember と creditMember は、レベル 0 のメンバーのみで構成できます。

借方メンバーおよび貸方メンバーは同じ次元の 2 つの異なるメンバーである必要があります。

debitMember と creditMember は割当ておよびカスタム計算で同様に動作します。1126 ページの「カスタム計算および割当てのための貸方および借方処理の理解」を参照してください。

基準およびターゲットの時間スパンの設定の理解

基準およびターゲットの時間スパンに指定されるメンバーの数は、表 221 で説明するように、Essbase で基準の時間スパン・オプションおよびターゲットの時間スパン・オプションがそれぞれ処理される方法に影響します。基準またはターゲットの時間スパンが空である、あるいは単一の期間に設定されている状況では、Essbase は、基準またはターゲットの時間スパン・オプションにそれぞれ設定した可能性のあるあらゆる設定を無視します。基準またはターゲットの時間スパンの一方または両方が複数の期間に設定されている状況では、Essbase は、基準またはターゲットの時間スパン・オプションに特定の設定を必要とします。

表 221　サマリー: 基準およびターゲットの時間スパンと、基準およびターゲットの時間スパン・オプション

基準の時間スパン	ターゲットの時間スパン	基準の時間スパン・オプション	ターゲットの時間スパン・オプション	参照
空または単一のメンバー	空または単一のメンバー	無視される	無視される	1114 ページの「例 1: 基準およびターゲットの時間スパン - 空または単一のメンバー」
空または単一のメンバー	複数のメンバー	無視される	除算または繰返し	1114 ページの「例 2: 基準の時間スパン - 空または単一のメンバー、ターゲットの時間スパン - 複数のメンバー」
複数のメンバー	空または単一のメンバー	結合	無視される	1116 ページの「例 3: 基準の時間スパン - 複数のメンバー、ターゲットの時間スパン - 空または単一のメンバー」
複数のメンバー	複数のメンバー	分割	無視される	1117 ページの「例 4: 基準およびターゲットの時間スパン - 複数のメンバー、基準の時間スパン・オプション - 分割」
複数のメンバー	複数のメンバー	結合	除算または繰返し	1118 ページの「例 5: 基準およびターゲットの時間スパン - 複数のメンバー、基準の時間スパン・オプション - 結合」

例 1: 基準およびターゲットの時間スパン - 空または単一のメンバー

この例では、基準およびターゲットの時間スパンが未設定か、単一の期間のみに設定されています。Essbase は、基準またはターゲットの時間スパン・オプションに選択した可能性のあるあらゆる設定を無視します。

例 2: 基準の時間スパン - 空または単一のメンバー、ターゲットの時間スパン - 複数のメンバー

この例では、基準の時間スパンが未設定で、複数の期間がターゲットの時間スパンに指定されています。基準の時間スパン・オプションは無視されます。ターゲットの時間スパン・オプションには、divide または repeat を選択できます。

金額が 1000 であると仮定します。表 222 には、各部門の基準(Dept_1 = 1)と範囲の合計基準(21)が示されます:

表 222　例 2: 基準値

	範囲						合計
	Dept_1	Dept_2	Dept_3	Dept_4	Dept_5	Dept_6	
基準	1[1]	2	3	4	5	6	21[2]

[1] メンバーの基準

[2] 範囲の合計基準

ターゲットの時間スパン・オプションの設定によって、割当ての計算方法が決まります。

- **指定されたターゲットの時間スパン期間全体で割当て金額を繰り返します:**

  このシナリオでは、Essbase は単一期間の割当てを実行し、割当て金額値をターゲットの時間スパン内のすべてのメンバーにコピーします。

  Essbase は次のアルゴリズムを使用します:

  ```
  alloc_amt = (basis_mbr_value/basis_total_range) * amount
  ```

  表 223 に示すように、Dec 07、Dept_1 の場合、メンバー基準値(1)は範囲全体の合計基準(21)で除算され、結果(0.04762)が金額(1000)で乗算されます: (1/21) * 1000 = 47.62。Essbase は 47.62 を Jan 08、Feb 08、Mar 08 および Apr 08 のセルにコピーします。Essbase は引き続き、各部門に対する Dec 07 の割当てを実行します。各ターゲットの時間スパンについて、範囲全体の割当て値の合計は金額(1000)と等しくなります。

表 223　例 2: 繰返しに設定されたターゲットの時間スパン・オプションを使用した割当て

	範囲						合計
ターゲットの時間スパンのメンバー	**Dept_1**	**Dept_2**	**Dept_3**	**Dept_4**	**Dept_5**	**Dept_6**	
Dec 07	47.62	95.24	142.86	190.48	238.10	285.71	1000
Jan 08	47.62	95.24	142.86	190.48	238.10	285.71	1000
Feb 08	47.62	95.24	142.86	190.48	238.10	285.71	1000
Mar 08	47.62	95.24	142.86	190.48	238.10	285.71	1000
Apr 08	47.62	95.24	142.86	190.48	238.10	285.71	1000
							5000[1]

[1] 合計割当て値

合計割当て金額は、元の金額値(1000)にターゲットの時間スパン・メンバーの数(5)を乗算したものです: 1000 * 5 = 5000。

- **指定されたターゲットの時間スパン期間全体で割当て金額を分割します**

  このシナリオでは、Essbase は 1 つの期間の割当てを実行し、割当て金額をターゲットの時間スパン内のすべてのメンバーで均等に分割します。

  Essbase は次のアルゴリズムを使用します:

  ```
  alloc_amt = ((basis_mbr_value/basis_total_range) * amount)/
  #_target_time_span_periods
  ```

  表 224 に示すように、Dec 07、Dept_1 の場合、Essbase は、繰返しのターゲットの時間スパン・オプション・シナリオで説明されたものと同じ計算を実行して、47.62 を導き出します。ただし、この金額は Dept_1 に対する 5 つすべてのターゲットの時間スパン期間で均等に分割されるため、9.52 が各ターゲット・セルに書き込まれます: 47.62/5 = 9.52。Essbase は引き続き、各部門に対す

る割当てを実行します。各ターゲットの時間スパンについて、範囲全体の割当て値の合計は(200)と等しくなります。

表 224　例 2: 分割に設定されたターゲットの時間スパン・オプションを使用した割当て

	範囲						合計
ターゲットの時間スパンのメンバー	**Dept_1**	**Dept_2**	**Dept_3**	**Dept_4**	**Dept_5**	**Dept_6**	
Dec 07	9.52	19.05	28.57	38.10	47.62	57.14	200
Jan 08	9.52	19.05	28.57	38.10	47.62	57.14	200
Feb 08	9.52	19.05	28.57	38.10	47.62	57.14	200
Mar 08	9.52	19.05	28.57	38.10	47.62	57.14	200
Apr 08	9.52	19.05	28.57	38.10	47.62	57.14	200
							1000[1]

[1] 合計割当て値

範囲全体の合計割当て値は元の金額値(1000)です: 200 * 5 = 1000。

例 3: 基準の時間スパン - 複数のメンバー、ターゲットの時間スパン - 空または単一のメンバー

この例では、複数の期間が基準の時間スパンに指定されていますが、ターゲットの時間スパンは未設定です。ターゲットの時間スパン・オプションは無視されます。基準の時間スパン・オプションの有効な選択肢は、combine だけです。

金額が 1000 であると仮定します。表 225 に示すように、各部門に使用される基準は、基準の時間スパンに対する基準値(Dept_1 = 15)の合計です。範囲の合計基準はすべての部門の基準値の合計です(147):

表 225　例 3: 基準値

	範囲						合計
基準の時間スパンのメンバー	**Dept_1**	**Dept_2**	**Dept_3**	**Dept_4**	**Dept_5**	**Dept_6**	
Dec 07	1	2	3	4	5	6	
Jan 08	2	3	4	5	6	7	
Feb 08	3	4	5	0	7	8	
Mar 08	4	5	6	1	8	9	
Apr 08	5	6	7	2	9	10	
合計	15[1]	20	25	12	35	40	147[2]

[1] 基準の時間スパン期間全体で合計される、各範囲メンバーの基準

[2] 範囲の合計基準

割当ては結合という基準の時間スパン設定を使用して計算され、ここでは基準の時間スパン期間全体の基準値の合計が使用されます。

Essbase は各範囲メンバーに対して次のアルゴリズムを使用します:

```
alloc_amt = (sum_across_basis_time_span/basis_total_range) * amount
```

表 226 に示すように、ターゲットの時間スパンが複数の期間に設定されていないため、各部門の割当て値が 1 つのターゲットの場所に書き込まれます。Dept_1 の割当て金額の場合、基準の時間スパンの合計(15)は範囲の合計基準(147)で除算され、結果(0.10204)が金額(1000)で乗算されます: (15/147) * 1000 = 102.04。Essbase は引き続き、範囲の各部門に対する割当てを実行します。

表 226　例 3: 結合に設定された基準の時間スパンを使用した割当て

	範囲						合計
	Dept_1	Dept_2	Dept_3	Dept_4	Dept_5	Dept_6	
ターゲット	102.04	136.05	170.07	81.63	238.10	272.11	1000[1]

[1] 合計割当て値

範囲全体の合計割当て値は元の金額値(1000)です。

例 4: 基準およびターゲットの時間スパン - 複数のメンバー、基準の時間スパン・オプション - 分割

この例では、複数の期間が基準の時間スパンおよびターゲットの時間スパンに指定され、基準の時間スパン・オプションは split に設定されています。分割という基準の時間スパン・オプションを使用する際、基準の時間スパンおよびターゲットの時間スパンで指定される期間は、同一である必要があります。(基準の時間スパン・オプションを combine に設定する例については、1118 ページの「例 5: 基準およびターゲットの時間スパン - 複数のメンバー、基準の時間スパン・オプション - 結合」を参照してください。)

金額が 1000 であると仮定します。表 227 に示すように、範囲の合計基準は全部門の基準値の合計です(165):

表 227　例 4: 基準値

	範囲						合計
基準の時間スパンのメンバー	Dept_1	Dept_2	Dept_3	Dept_4	Dept_5	Dept_6	
Dec 07	1	2	3	4	5	6	21[1]
Jan 08	2	3	4	5	6	7	27
Feb 08	3	4	5	6	7	8	33
Mar 08	4	5	6	1	8	9	39
Apr 08	5	6	7	2	9	10	45
							165[2]

[1] 基準の時間スパンの各期間に対する合計基準

[2] 範囲の合計基準

割当ては分割という基準の時間スパン設定を使用して計算され、ここでは各期間に対する基準値が個々に使用されます。

Essbase は次のアルゴリズムを使用します:

```
alloc_amt = (basis_mbr_value/basis_total_range) * amount
```

表 228 に示すように、Dec 07、Dept_1 の場合、メンバー基準値(1)は範囲に対する合計基準(165)で除算され、結果(0.00606)が金額(1000)で乗算されます: (1/165) * 1000 = 6.06。Essbase は引き続き、各部門に対する各期間の割当てを実行します。

表 228　例 4: 分割に設定された基準の時間スパンを使用した割当て

	範囲						合計
ターゲットの時間スパンのメンバー	**Dept_1**	**Dept_2**	**Dept_3**	**Dept_4**	**Dept_5**	**Dept_6**	
Dec 07	6.06	12.12	18.18	24.24	30.30	36.36	127.27
Jan 08	12.12	18.18	24.24	30.30	36.36	42.42	163.64
Feb 08	18.18	24.24	30.30	36.36	42.42	48.48	200.00
Mar 08	24.24	30.30	36.36	42.42	48.48	54.55	236.36
Apr 08	30.30	36.36	42.42	48.48	54.55	60.61	272.73
							1000[1]

[1] 合計割当て値

範囲全体の合計割当て値は元の金額値(1000)です。

例 5: 基準およびターゲットの時間スパン - 複数のメンバー、基準の時間スパン・オプション - 結合

この例では、複数の期間が基準およびターゲットの時間スパンに指定されていますが、基準の時間スパン・オプションが combine に設定されているため、基準およびターゲットの時間スパンが同じメンバー・セットを含む必要はありません。(基準の時間スパン・オプションを split に設定する例については、1117 ページの「例 4: 基準およびターゲットの時間スパン - 複数のメンバー、基準の時間スパン・オプション - 分割」を参照してください。)

金額が 1000 であると仮定します。表 229 に示すように、各部門に使用される基準は、基準の時間スパン全体の基準値(Dept_1 = 10)の合計です。範囲の基準はすべての部門の基準値の合計です(113):

表 229　例 5: 基準値

	範囲						合計
基準の時間スパンのメンバー	**Dept_1**	**Dept_2**	**Dept_3**	**Dept_4**	**Dept_5**	**Dept_6**	
Dec 07	1	2	3	4	5	6	
Jan 08	2	3	4	5	6	7	
Feb 08	3	4	5	6	7	8	
Mar 08	4	5	6	0	8	9	
合計	10[1]	14	18	15	26	30	113[2]

[1] 基準の時間スパン期間全体で合計される、各範囲メンバーの基準

[2] 範囲の合計基準

ターゲットの時間スパン・オプションの設定によって、割当ての計算方法が決まります。

- **指定されたターゲットの時間期間全体で割当て金額を繰り返します:**

  このシナリオでは、Essbase は単一期間の割当てを実行し、割当て金額値をターゲットの時間スパン内のすべてのメンバーにコピーします。

  Essbase は各範囲メンバーに対して次のアルゴリズムを使用します:

```
alloc_amt = (sum_across_basis_time_span/basis_total_range) * amount
```

  表 230 に示すように、Dec 07、Dept_1 の場合、Dept_1 の基準(10)は範囲に対する合計基準(113)で除算され、結果(0.0885)が金額(1000)で乗算されます: (10/113) * 1000 = 88.50。Essbase は 88.50 を Jan 08、Feb 08、Mar 08 および Apr 08 のセルにコピーします。Essbase は引き続き、各部門に対する Dec 07 の割当てを実行します。各ターゲットの時間スパンについて、範囲全体の割当て値の合計は金額(1000)と等しくなります。

表 230　例 5: 繰返しに設定されたターゲットの時間スパン・オプションを使用した割当て

	範囲						合計
ターゲットの時間スパンのメンバー	**Dept_1**	**Dept_2**	**Dept_3**	**Dept_4**	**Dept_5**	**Dept_6**	
Dec 07	88.50	123.89	159.29	132.74	230.09	265.49	1000
Jan 08	88.50	123.89	159.29	132.74	230.09	265.49	1000
Feb 08	88.50	123.89	159.29	132.74	230.09	265.49	1000
Mar 08	88.50	123.89	159.29	132.74	230.09	265.49	1000
Apr 08	88.50	123.89	159.29	132.74	230.09	265.49	1000
							5000[1]

[1] 合計割当て値

合計割当て値は、元の金額値(1000)にターゲットの時間スパン・メンバーの数(5)を乗算したものです: 1000 * 5 = 5000。

- **指定されたターゲットの時間期間全体で割当て金額を除算します:**

  このシナリオでは、Essbase は単一期間の割当てを実行し、割当て金額をターゲットの時間スパン内のすべてのメンバーで均等に除算します。

  Essbase は次のアルゴリズムを使用します:

```
alloc_amt = ((basis_time_span/basis_total_range) * amount)/
#_target_time_span_periods
```

  表 231 に示すように、Essbase は、繰返しのターゲットの時間スパン・オプション・シナリオで説明されたものと同じ計算を実行して、88.50 を導き出します。ただし、この金額は Dept_1 に対する 5 つすべてのターゲットの時間スパン期間で均等に除算されるため、17.70 が各ターゲット・セルに書き込まれます: 88.50/5 = 17.70。Essbase は引き続き、各部門に対する割当てを実行します。各ターゲットの時間スパンについて、範囲全体の割当て値の合計は(200)と等しくなります。

表 231　例 5: 除算に設定されたターゲットの時間スパン・オプションを使用した割当て

	範囲						合計
ターゲットの時間スパンのメンバー	**Dept_1**	**Dept_2**	**Dept_3**	**Dept_4**	**Dept_5**	**Dept_6**	
Dec 07	17.70	24.78	31.86	26.55	46.02	53.10	200
Jan 08	17.70	24.78	31.86	26.55	46.02	53.10	200
Feb 08	17.70	24.78	31.86	26.55	46.02	53.10	200
Mar 08	17.70	24.78	31.86	26.55	46.02	53.10	200
Apr 08	17.70	24.78	31.86	26.55	46.02	53.10	200
							1000[1]

[1] 合計割当て値

範囲全体の合計割当て値は元の金額値(1000)です: 200 * 5 = 1000。

集約ストレージ割当ての例

次の例は、POV の組合せを変更すると、割当ての基準として想定されている値にどのように影響するかを示しています。この例では、シェア割当てメソッドを使用し、前年の総賃貸料を現行の年のすべてのコスト・センターに、各コスト・センターの人数に基づいて割り当てます。集約ストレージ・データベースに Departments、Time、CostCenter および Measures の 4 つの次元があり、割当て基準が表 232 のように指定されていると仮定します:

表 232　POV の例: 割当て基準

基準	定義
POV	Dept_A、Dept_B
金額	2007、CCNA、TotalRent 金額値を次のように仮定します: ● Dept_A = 1000 ● Dept_B = 2000
基準	Jan 2008、Head count
ターゲット	Jan 2008、RentalAllocation
範囲	CostCenter のレベル 0 の子孫 範囲が次のコスト・センターの値を求めると仮定します: ● CostCenter1 ● CostCenter2 ● CostCenter3 ● CostCenter4

割当てが各 POV の組合せに対して実行されます:

- Dept_A
- Dept_B

表 233 に示すように、各 POV の組合せには、割当ての計算で使用される固有の基準値のセットがあります: 範囲内の各コスト・センターの人数と Jan 2008 の合計の人数です:

表 233　POV の例: 各 POV の組合せに対する基準値

	メンバー基準値				範囲基準値
POV	**CostCenter1**	**CostCenter2**	**CostCenter3**	**CostCenter4**	**2008 の人数合計**
Dept_A	1	2	3	5	11
Dept_B	5	0		10	15

各 POV について、Essbase は各コスト・センターの人数(各メンバーの基準値)を、範囲の合計人数(範囲の基準値)で除算してから、その値を各部門の合計賃貸料(金額)で乗算します。たとえば、Dept_A、CostCenter1 の場合、メンバー基準値(1)が範囲の基準(11)で除算され、その結果(0.09090909)が金額(1000)で乗算されます: (1/11) * 1000 = 90.90909。Dept_B、CostCenter1 の場合、割当て金額は 666.6667 です: (5/15) * 2000 = 666.6667。表 234 に、各コスト・センターに対する割当てシェア金額を示します:

表 234　POV の例: 各 POV の組合せに対するターゲット値

	メンバー・ターゲット値				金額値
POV	**CostCenter1**	**CostCenter2**	**CostCenter3**	**CostCenter4**	**賃貸料割当て合計**
Dept_A	90.90909	181.8182	272.7273	454.5455	1000
Dept_B	666.6667	0		1333.333	2000

集約ストレージ割当ての使用例

この使用例の目的は、シェア割当てメソッドを使用することで、各部門が占有する平方フィート数に基づいて総月額賃貸料を部門全体で比例再配分することです。

次のような次元のアウトラインを想定します:

- **会社**: 複数の元帳が含まれます。賃貸料割当ては Vision US 元帳で行われます。
- **部門**: 次のメンバーが含まれます:
  - 100: Vision US の総月額賃貸料を$100,000 の中で蓄えておきます。この金額は部門 999 の子の間で比例割当てされます。
  - 999: 次の各部門の親です:
    - 101: 賃貸料割当ての 45%を配分されます
    - 102: 賃貸料割当ての 30%を配分されます
    - 103: 賃貸料割当ての 25%を配分されます
- **勘定科目**: 次のメンバーが含まれます:
  - 5740: 賃貸料の勘定科目
  - SQFT: 各部門の平方フィート数を記録するために使用される統計勘定科目
- **金額タイプ**: 次のメンバーの親である PeriodActivty が含まれます:
  - PeriodActivityDebit: ターゲットの場所
  - PeriodActivityCredit: オフセットの場所

賃貸料割当ては様々な方法で実行が可能で、個々の方法は同じ結果になります。2 つのシナリオが存在します。各シナリオについて、各パラメータが次のように定義されていると仮定します:

- **割当てメソッド**: シェア
- **範囲**: 部門 999 の子孫:
  - 101
  - 102
  - 103

  範囲内のセルは除外されません。
- **基準**: アクティビティの期間(月次)に対する各範囲メンバーの平方フィート数。

```
SQFT,PeriodActivity
```

- **ゼロ金額オプション**: (デフォルト)ゼロの金額値を割り当てます。
- **ゼロ基準オプション**: 基準値がゼロの場合、割当て操作を取り消します。
- **基準の時間スパン・オプション**: (デフォルト)分割、各期間の基準値を個別に使用します。
- **丸めメソッド**: 割当て値を最も近い 1,000 に丸めて、丸め誤差合計を部門 101 に追加します。
- **借方メンバー**: 合計が正数の場合は、値を PeriodActivityDebit に書き込みます。
- **貸方メンバー**: 値を PeriodActivityCredit に書き込みます。

シナリオ 1: 集約ストレージ割当て

シナリオ 1 では、各パラメータが次のように定義されていると仮定します:

- **POV**: 会社次元の 1 つのメンバー、Vision US で構成されます。

  1 つの次元の 1 つのメンバーのみが指定されているため、POV は変化しないので割当ては 1 回だけ実行されます。
- **金額**: 割当てのソース値は次の次元間メンバーが元になります:

  ```
  5740,100,Beginning Balance
  ```

- **ターゲット**: 割当て値を各部門の勘定科目 5740 に書き込みます。
- **オフセット**: オフセット項目をメンバー 5740,100 に書き込みます。

シナリオ 2: 集約ストレージ割当て

シナリオ 2 では、各パラメータが次のように定義されていると仮定します:

- **POV**: 2 つの次元のそれぞれ 1 つのメンバーで構成されます:
  - Vision US: 会社次元のメンバー
  - 5740: 金額次元のメンバー

    このシナリオでは、勘定科目 5740 は POV の一部です。基準では、勘定科目 5740 はメンバー SQFT で上書きされます。

  各次元の 1 つのメンバーのみが指定されているため、POV は変化しないので割当ては 1 回だけ実行されます。
- **金額**: 割当てのソース値は次の次元間メンバーが元になります:

  ```
  100,Beginning Balance
  ```

- **ターゲット**: 未設定です。POV、ターゲット、借方メンバー/貸方メンバー、および範囲の組合せはすべての次元のメンバーを使用するため、ターゲットは空にできます。
- **オフセット**: オフセット項目を部門 100 に書き込みます。

式の使用時におけるデータ不整合の回避

式メンバーは金額および基準では使用できますが、ターゲットでは使用できません。ただし、集約ストレージ・データベースではトランザクション・セマンティクスがサポートされていないため、金額または基準で式メンバーを使用する際にデータ不整合の問題が発生する可能性があります。

次の例では、ユーザー 1 が一連の部門の収益値を転記し、各部門の年次累計収益が割当ての基準となる、ボーナス金額の割当てをユーザー 2 が実行します。これらの操作が実行される順序が結果に影響を及ぼします:

- シナリオ 1: ユーザー 2 が割当てを実行する前にユーザー 1 が収益を転記します。

  割当て結果は更新された収益値に基づきます。
- シナリオ 2: ユーザー 1 が収益を転記する前にユーザー 2 が割当てを実行します。

  割当て結果は更新された収益値ではなく、以前の収益値に基づきます。
- シナリオ 3: ユーザー 1 による収益の転記とユーザー 2 による割当ての実行が同時に行われます。

  最初に開始されたユーザーの操作に応じて、割当て結果は更新済または以前の収益値に基づきます。

  金額または基準で式メンバーを使用する際は、これらの操作を同時に実行することはお薦めしません。

また、割当てのための年次累計収益の計算に MDX 式が使用されると仮定します。式の複雑性が結果に影響を及ぼす場合があります:

- シナリオ 4: 年次累計収益の式には 1 つの次元のメンバーが組み込まれ、+、-、/および*の演算式のみが使用されます。

  割当て結果は完全に、更新済または以前の収益値に基づきます。

  シナリオ 4 で説明したような、シンプルな MDX 式を使用することをお薦めします。
- シナリオ 5: 年次累計収益の式がシナリオ 4 の式よりもはるかに複雑です。

  割当て結果の一部は更新された収益値に基づいて、一部は以前の収益値に基づく可能性があります。

カスタム計算および割当てのためのデータ・ロード・バッファの理解

集約ストレージ・データベースで割当てまたはカスタム計算を実行する際、Essbase は一時的なデータ・ロード・バッファを使用します。データ・ロード・バッファを作成するには不十分なリソースが集約ストレージ・キャッシュにある場合、Essbase はリソースが使用可能になるまで待機します。

1 つの集約ストレージ・データベースに、複数のデータ・ロード・バッファが存在できます。Essbase が割当ておよびカスタム計算のために作成するデータ・ロード・バッファは、構成不可能です。ただし、データ・ロードおよび転記のために作成するデータ・ロード・バッファは構成できます。

割当てとカスタム計算を、データ・ロードおよび転記と同時に実行する場合は、データ・ロードおよび転記のために作成するデータ・ロード・バッファのリソース使用率を、最大の 0.8(80%)に設定します。リソース使用率を低く設定するほど、データ・ロードおよび転記と同時に実行可能な割当ておよびカスタム計算の数は増加します。また、ロード・バッファ操作を処理するため、リソースが使用可能になるのを Essbase が待機する時間も構成できます。

データ・ロード・バッファを構成するには、**initialize load_buffer** 文法および ASOLOADBUFFERWAIT 構成設定で MaxL の **alter database** ステートメントを使用します。

カスタム計算および割当てのためのオフセット処理の理解

総勘定元帳の記帳では、オフセット項目は、等しい値の取引からの元帳の反対側の相殺メジャーとなります。このドキュメントでは、オフセット項目はオフセットと呼ばれます。

オフセットの指定はオプションです。オフセットは、借方の合計と貸方の合計が等しくない場合に必要になる可能性があります。貸方および借方の合計が等しくない場合、元帳は未決済になります。このようなケースで、オフセットが元帳の貸借一致に使用されます。

たとえば、January の$100 という貸方では、元帳の借方側に$100 のオフセットが追加される可能性があるため、元帳はその金額の今後の支出への準備として貸借一致する場合があります。

オフセットはタプルの形式で指定する場所で、この場所にカスタム計算スクリプトの結果をオフセットする値が Essbase によって書き込まれます。

次の例では、POV が **Prod1**、**Prod2**、**AcctA**、**AcctB**、**Jan** と仮定します。

次のカスタム計算スクリプトの合計は 13 になります。

```
   mbr1 := 7;
mbr2 := -4;
mbr3 := 0;
mbr4 := 10;
```

したがって、オフセットが必要な場合、オフセットも 13 である必要があります。オフセットが"Offset_Member"というメンバーに書き込まれると仮定します。

```
        Debit   Credit
mbr1      7
mbr2           4
mbr3      0
mbr4     10
mbr_offset        13
Total     17     17
```

オフセットが使用される際、貸方および借方処理は逆順になります。オフセットが貸方および借方処理で使用される場合は次のような計算プロセスが実行されます。

1. 指定された POV について、計算スクリプトで記述された結果の合計(このケースでは 13)を取得します。
2. 合計が正数の場合は、ターゲット・データベースの貸方メンバーに書き込みます。
3. 合計が負数の場合は、正数に変更してからターゲット・データベースの借方メンバーに書き込みます。

カスタム計算および割当てのための貸方および借方処理の理解

Oracle General Ledger は複式簿記を使用します。ここではすべての取引に借方項目と貸方項目の 2 つの仕訳があります。

したがって、すべての取引に対して 2 つの勘定科目があり、列として表されます。2 つの勘定科目は貸借一致する必要があります。つまり、借方列の合計は貸方列の合計と等しくなります。

借方メンバーが指定可能で、これにはカスタム計算が正の結果値を書き込み、貸方メンバーも指定可能で、これにはカスタム計算が負の結果値およびオフセット結果値を書き込みます。借方メンバーおよび貸方メンバーは同じ次元の 2 つの異なるメンバーである必要があります。たとえば、"AmountType"という次元には"Credit"および"Debit"という 2 つのレベル 0 の子がある場合があります。

ターゲット・データベースのレベル 0 セルに正数が書き込まれるような計算結果になった場合は、常に、正の値が借方メンバーに書き込まれます。

ターゲット・データベースのレベル 0 セルに負数が書き込まれるような計算結果になった場合は、常に、記号が正に変更されて、貸方メンバーに書き込まれます。

65 集約ストレージ・アプリケーションとデータベースの管理

この章の内容

この章の情報は、集約ストレージ・データベースのみに適用され、ブロック・ストレージ・データベースとは関係がありません。

関連項目:

- 第 60 章「集約ストレージとブロック・ストレージの比較」
- 『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』(集約ストレージ・アプリケーションのバックアップの詳細)

集約ストレージのセキュリティ

集約の定義および実行には、計算権限(Administration Services)または実行(MaxL)権限かそれ以上の権限が必要です。データベース値を消去する次元構築には、書込み権限が必要です。

セキュリティ関連のトピックを参照してください:

- 669 ページの「Essbase ネイティブ・セキュリティ・モードについて」(『Oracle Essbase データベース管理者ガイド』)
- 「アプリケーションおよびデータベースに対するユーザー/グループ権限の管理」

  Oracle Essbase Administration Services Online Help の「カスタム定義関数の削除」を参照してください。
- 『Oracle Essbase テクニカル・リファレンス』の権限およびロールに関する項

集約ストレージ・アプリケーションのストレージ管理

集約ストレージ・アプリケーションでは、テーブルスペース・マネージャで、テーブルスペース定義を使用してディスク上のデータ・ストレージと作業領域を管理して、データの取得とストレージを制御します。

テーブルスペースでの処理

テーブルスペースは、データ・ファイルと作業ファイルのストレージおよび取得の最適化に役立ちます。テーブルスペースでは、集約ビューや集約などのデータ・アーティファクトをファイルにマップする場所の定義を行います。各アプリケーション・ディレクトリは、次の 4 つのテーブルスペース用のディレクトリに含まれています:

- `default` - データベースのデータ構造とデータベース値が格納されます(データのロード後は、このテーブルスペースの場所は変更できません。)
- `log` - デフォルトのテーブルスペース更新のバイナリ・トランザクション・ログが格納されます
- `metadata` - ファイルの場所、ファイル、およびデータベースに含まれるオブジェクトに関する情報が格納されます
- `temp` - データ・ロード、集約および取得などの操作中に使用される、一時ワークスペースを提供します

テーブルスペース名は、オペレーティング・システムにかかわらず、大文字と小文字が区別されます。metadata と log の場所やサイズは変更できません。default と temp では、複数の場所とサイズを指定でき、次のようなテーブルスペース・プロパティを定義できます:

- ディレクトリ・パスの場所
- それぞれの場所で使用する最大ディスク・スペース
- それぞれの場所に有効な最大ファイル・サイズ

注: テーブルスペース内に情報を保管するために使用されるファイルの場所が空の場合は、その場所を変更または削除できます。

テーブルスペース・マネージャによりディスク・スペースが固定サイズの増分で割り当てられるため、テーブルスペースの場所に最大ディスク・スペースを指定すると、終点が指定され、指定されたスペースは予約されません。

スペースが必要になると、Essbase でファイルの場所が(番号順に)チェックされ、スペースが見つかると書込みが開始されます。すべての割当てが使用されると、使用可能なスペースがなくなり、エラーが戻されます。データベース値が消去されると、テーブルスペース・ファイルが縮小されるため、ディスク・スペースが解放されます。必要のなくなった作業ファイルは削除され、他のプログラムでスペースを使用できるようになります。

ファイルに指定された最大サイズに基づいて、Essbase では複数のファイルが書き込まれます。たとえば、ess00001.dat、ess00002.dat、などのファイルです。データベース・ファイルを他のメディアにバック・アップする場合は、テーブルスペース・ファイルの最大サイズを、そのメディアで扱えるサイズよりも大きいサイズに設定しないでください。

テーブルスペースの定義

テーブルスペースの定義は、集約ストレージ・アプリケーションごとに行います。

➤ テーブルスペースを定義するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	テーブルスペースの管理	Oracle Essbase Administration Services Online Help
MaxL	**alter tablespace**	『Oracle Essbase テクニカル・リファレンス』

注： UNIX プラットフォームでは、最大 2 GB のファイル制限が適用されます。データ・ロード中や集約構築中に.dat ファイル制限を超えると、"Failed to extend file: file exceeds maximum file size for this system."というメッセージが表示されます。そのデータ・ファイルは Essbase により閉じられ、次のファイル(essn+1)が作成され、続行されます。

集約ストレージ・キャッシュの管理

集約ストレージ・キャッシュにより、データ・ロード、集約および取得中のメモリーの使用が容易になります。キャッシュ・メモリーのロック機能が使用されるのは、ブロック・ストレージ・アプリケーションでのみです。

集約ストレージ・アウトラインが開始されると、メモリー内の小さい領域が関連のアプリケーション用の集約ストレージ・キャッシュとして割り当てられます。追加のキャッシュ領域が必要になると、最大キャッシュ・サイズが使用されるか、オペレーティング・システムで追加の割当てが拒否されるまで、キャッシュ・サイズは増分的に増大します。

注： 集約キャッシュ・メモリーの割当ての拒否によって、既存のメモリーの使用増加が拒否されるわけではありません。

現在の集約キャッシュ・メモリーの割当てと、最大集約キャッシュ・サイズの設定を確認できます。設定を変更すると、メモリーの使用を最適化できる場合があります。デフォルトの最大キャッシュ・サイズである 32 MB は、最小の設定サイズです。入力レベル・データのサイズを使用すれば、キャッシュの最大サイズを増やす必要があるタイミングを決定できます。Administration Services および MaxL では、入力レベル・データのサイズが集約ストレージ・データベース・プロパティ(レベル 0 の値のサイズ)として表示されます。

32 MB のキャッシュ設定は、約 2 GB の入力レベル・データを含むデータベースをサポートします。入力レベル・データのサイズがある要素によって 2 GB を超えた場合、集約ストレージ・キャッシュはその要素の平方根によって増やすことができます。たとえば、入力レベル・データのサイズが 3 GB (2 GB * 1.5)の場合は、32 MB の集約ストレージ・キャッシュ・サイズに 1.5 の平方根を乗算し、集約キャッシュ・サイズをその結果である 39.04 MB に設定します。

集約生成のパフォーマンスについては、並列計算のためのスレッド数を考慮してください。集約生成プロセスでは、集約ストレージ・キャッシュを分割する複数のスレッドが使用されます。集約ストレージ・アプリケーションまたはデータベースの CALCPARALLEL 構成設定に指定されたスレッド数を増やすと、集約ストレージ・キャッシュ・サイズの増加が必要になることがあります。『Oracle Essbase テクニカル・リファレンス』の CALCPARALLEL 構成設定を参照してください。

注： スレッド数をプロセッサ数より大きい数に設定すると、集約ストレージ・アプリケーションのパフォーマンスが向上することがあります。

集約ストレージ・キャッシュの最大サイズは、必要なサイズを超える大きさにはしないでください。

➤ 集約ストレージ・キャッシュのサイズを設定するには、次のツールを使用します:

ツール	トピック	場所
Administration Services	集約ストレージ・キャッシュのサイズ設定	Oracle Essbase Administration Services Online Help
MaxL	**query application** **alter application**	『Oracle Essbase テクニカル・リファレンス』

注： 変更した集約ストレージ・キャッシュの設定は、アプリケーションを再起動したときに有効になります。

集約ストレージ・データベースに集約ビューを構築する際のパフォーマンスの向上

集約ストレージ・データベースに集約ビューを構築する際に、次のメッセージが表示される場合があります:

```
For better performance, increase the size of aggregate storage cache
```

このメッセージは、集約ストレージ・データベースの入力セル数が億を超えている場合に表示されることがあります。

集約構築のパフォーマンスを向上させるには、次の手順を実行します。

1. 集約ストレージ・キャッシュのサイズを、少なくとも 512MB に増やすか、入力データ・サイズの 20%に増やすか、いずれか小さい方にします。(キャッシュ設定がすでにこのサイズより大きい場合は、次の手順に進みます。)管理サービス・コンソール、または次の MaxL コマンドを使用できます:

```
alter application
appname
 set cache_size
x
MB
```

この設定は、アプリケーションを再起動すると有効になります。

2. 集約ストレージ・キャッシュを増やした後も、集約ビューの構築中にメッセージが表示される場合は、ASOSAMPLESIZEPERCENT 構成設定を使用します。
構文:

```
ASOSAMPLESIZEPERCENT [
appname
 [
dbname
]]
n
```

メッセージが表示されなくなるまで少しずつ n の値を増やすと、最適な集約パフォーマンスになります。データベースに含まれるものが次の場合:

- 入力セルが 2000 万の場合、5%から開始します
- セルが 1 億の場合、1%から開始します
- セルが 10 億を超える場合、0.1%から開始します

集約ビューの選択を解除します; その後、再度選択して構築し直します。メッセージが表示される場合は、設定を増やして再試行します。

集約ビュー構築のパフォーマンスは、メッセージが表示されなくなるまで向上しません。メッセージが表示されなくなり、パフォーマンスが最大になったら、設定の増加を終了します。

注: ASOSAMPLESIZEPERCENT を高く設定しすぎると、パフォーマンスは低下します。セルが 10 億を超えるデータベースに最適な設定は、ほとんどの場合 3%未満です。ASOSAMPLESIZEPERCENT の詳細は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

集約ストレージ・データベースの再構築

データベースの再構築は、階層に対する変更などの集約ストレージ・データベースのアウトライン変更によって、強制的に行われることがあります。階層は、最上位のメンバーとその子孫で構成されます。

- 動的階層には、1 つの保管レベルのみ含まれます。会計次元は動的階層です。
- 属性次元は 1 つの階層です。世代 1 は階層の最上位のメンバーです。
- 標準次元に複合階層使用可能とタグ付けされていない場合、その標準次元は 1 つの階層です。世代 1 は階層の最上位のメンバーです。

- 標準次元に複合階層使用可能とタグ付けされている場合、その標準次元には複数の階層が含まれます。世代 2 のメンバーは階層の最上位のメンバーです。たとえば、ASOSamp.Sample の「Products」次元には 2 つの階層が含まれます。最上位のメンバーは、世代 2 のメンバーである「All Merchandise」と「High End Merchandise」です。

影響を及ぼすアウトライン変更は、次のとおりです:

- 再構築の前にデータベースからデータを消去する必要があるかどうか
- アウトラインの再構築に必要な時間およびストレージ

集約ストレージ・データベースの再構築のレベル

データベースの再構築に必要な時間とストレージを最小限に抑えるには、データベース・アウトラインが頻繁に変更される場合は、アウトラインとアウトライン変更のタイプを分析します。

集約ストレージ・データベースの再構築のレベルを、(時間、ストレージおよびデータに関して)最も費用がかかるものから最もかからないものの順にリストします:

表 235　集約ストレージの再構築レベル

ユーザー・アウトラインの変更	Essbase - 再構築レベル	パフォーマンスへの影響
標準次元の追加、削除、移動	データと集約ビューの消去およびアウトライン全体の再構築の実行	非常に大きい ユーザーは、入力(レベル 0)データを再ロードし、集約ビューを選択して、データベース集約を再実行する必要があります。

ユーザー・アウトラインの変更	Essbase - 再構築レベル	パフォーマンスへの影響
• 階層を追加、削除、移動します。 • 階層内の保管されるレベル数を変更します。 参照: ❍ 1135 ページの「例: 階層内の保管されるレベル数の変更なし」 ❍ 1135 ページの「例: 階層内の保管されるレベル数の変更」 • 保管階層の最上位のメンバーを、ラベルのみから保管済、または保管済からラベルのみに変更します。 • 動的階層を保管階層に、または保管階層を動的階層に変更します。 • プライマリ階層または代替階層がそのプライマリまたは代替階層に一致するか一致しなくなるような、そのプライマリ階層または代替階層を変更します。 プライマリ階層のすべてのレベル 0 のメンバーは、すべての代替階層内で直接または間接的に表される必要があります(たとえば、子の合計である親はその子を表すことがあります)。プライマリ階層の最上位レベルは、各代替階層の最上位レベルと等しい必要があります。1136 ページの「例: 代替階層での変更」を参照してください。	集約ビューの消去、およびアウトライン全体の再構築の実行	非常に大きい ストレージ要件は、データベース・ファイル(.dat ファイル)のサイズの最大 3 倍までです。 ユーザーは集約ビューを選択してデータベース集約を再実行する必要があります。
他のカテゴリに含まれていない変更の実行(メンバーの削除や移動、または兄弟の中で最後ではないメンバーの追加など)	アウトライン全体の再構築の実行	大きい ストレージ要件は、データベース・ファイル(.dat ファイル)のサイズの最大 3 倍までです。
代替階層または属性次元に対する簡易再構築の変更の実行(下記参照)	属性次元や代替階層をベースとするすべての集約ビューの再構築	小さい ストレージ要件は、影響を受けるビューのサイズの最大 3 倍までです。このような集約ビューは、通常、使用状況に基づいてビューを選択するためにクエリーの追跡を使用した場合のみ存在します。1085 ページの「使用状況に基づいたビューの選択」を参照してください。

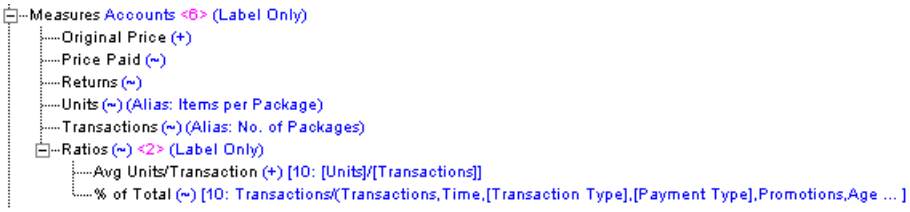
ユーザー・アウトラインの変更	Essbase - 再構築レベル	パフォーマンスへの影響
レベル 0 の保管済メンバーを持たない属性次元以外の次元(たとえば、すべてのレベル 0 のメンバーは、共有されているか式を持つ)で、階層内のレベル数を変更せず、2 の累乗の境界を交差しない子または子分岐を追加します。	アウトラインの簡易再構築の実行 **注：** 階層内のレベル数を変更すると、Essbase ではすべての集約ビューが消去され、アウトライン全体の再構築が実行されます。パフォーマンスの影響は非常に大きくなります。 階層内のレベル数が変更されず、子または子分岐が 2 の累乗の境界を交差すると、Essbase ではすべてのアウトラインの再構築が実行されます。パフォーマンスの影響は大きくなります。	非常に小さい
保管済のレベル 0 のメンバーを持つ属性次元以外の属性次元: ● 2 の累乗の境界を交差しない親の最後の子として子を追加します。(1、2、4、8、16 など)。たとえば、親メンバーに 3 つの子がある場合、4 番目の子を親の最後の子として追加できます。 ● 2 の累乗の境界を交差せず、階層内のレベル数を変更しない、既存の親の最後の子分岐としての子分岐を追加します。 例: ● メンバー名を変更します ● 式の変更 ● 別名の変更 ● 動的階層の集計演算子の変更(+から-への変更など)	アウトラインの簡易再構築を実行します。	非常に小さい
保管済のレベル 0 のメンバーを持つ属性次元以外の属性次元: ● 2 の累乗の境界を交差する子を親の最後の子として追加します。たとえば、親メンバーに 3 つの子があり、4 番目と 5 番目の子を追加する場合、5 番目の子は 2 の累乗の境界を交差します。1136 ページの「例: 子メンバーの追加」を参照してください。 ● 2 の累乗の境界を交差する既存の親の最後の子分岐として子分岐を追加するか、階層内のレベル数を変更し、すべてのアウトラインの再構築をトリガーするシナリオの場合は、1137 ページの「例: 子分岐の追加」を参照してください。	集約ビューの消去、およびアウトライン全体の再構築の実行	非常に大きい

アウトライン変更の例

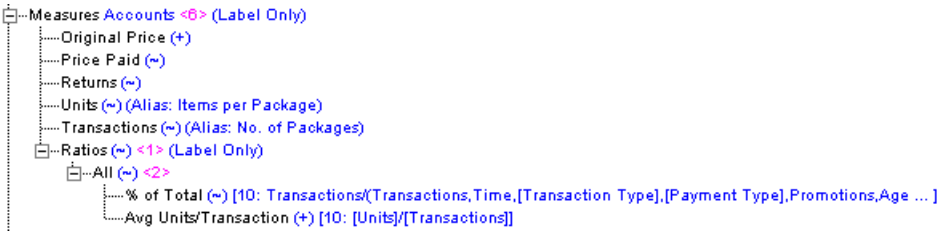
この項には、表 235 で説明している、より複雑なアウトライン変更の例が含まれます。

例: 階層内の保管されるレベル数の変更なし

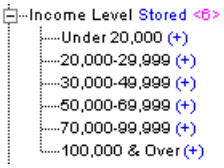
ASOSamp.Sample では、「Measures」次元に勘定科目とタグ付けされています。したがって、動的階層としての「Measures」に含まれる保管レベルは 1 つのみです。



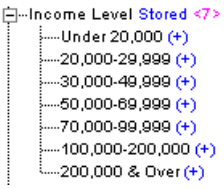
「Ratios」に子メンバー「All」を追加しても、「Measures」次元の保管されるレベル数は変更されません。アウトラインを保存すると、簡易再構築がトリガーされます。



ASOSamp.Sample では、「Income Level」は保管階層次元です。

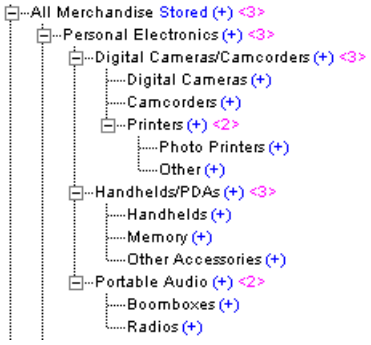


子メンバーを追加しても、階層内のレベル数(2)は変更されません。7 個目または 8 個目の子メンバーを最後に追加できますが、9 個目の子メンバーを追加すると、2 の累乗の境界を交差するため(1136 ページの「例: 子メンバーの追加」を参照)、アウトライン全体の再構築が必要になります。



例: 階層内の保管されるレベル数の変更

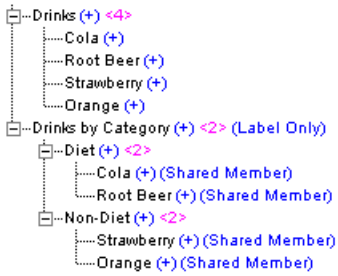
ASOSamp.Sample の製品次元で、「Photo Printers」を「Printers」に名前変更して子メンバーを追加すると、「All Merchandise」階層のレベル数が 4 から 5 に増えます。アウトラインを保存すると、Essbase ではすべての集約ビューが消去され、アウトライン全体の再構築が実行されます。



例: 代替階層での変更

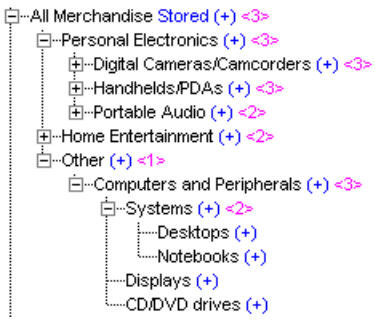
「Drinks by Category」の下の共有メンバー「Orange」を削除して、「Drinks」の下のその非共有メンバーを削除しない場合、代替階層「Drinks by Category」は「Drinks」階層のレプリカではなくなります。アウトラインを保存すると、Essbase ではすべての集約ビューが消去され、アウトライン全体の再構築が実行されます。

共有と非共有の「Orange」のメンバーを削除すると、代替階層「Drinks by Category」は引き続き「Drinks」階層のレプリカのままになります。アウトラインを保存すると、Essbase ではアウトライン全体の再構築が実行されますが、集約ビューは消去されません。



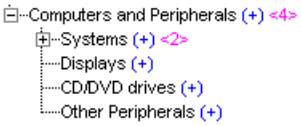
例: 子メンバーの追加

ASOsamp.Sample では、「All Merchandise」階層内の「Systems」の下に子メンバーを追加すると、「Systems」の下の子の数が 3 つに増え、2 の累乗の境界を交差します。アウトラインを保存すると、Essbase ではアウトライン全体の再構築が実行されます。



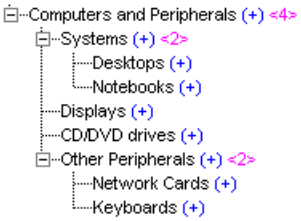
ただし、「Computers」と「Peripherals」の下に子メンバーを追加すると、「Computers」と「Peripherals」の下の子の数が 3 つから 4 つに増えます。4 番目の子は、既存のメンバーの後に追加する必要があり、これを追加しても 2 または 4

の境界は交差しません。この子は既存のメンバーの後に追加する必要があります。アウトラインを保存すると、Essbase では簡易再構築が実行されます。



例: 子分岐の追加

ASOSamp.Sample では、「All Merchandise」階層内の「Computers and Peripherals」の下に子分岐を追加すると、子の数が 4 つに増えます。この子は、既存のメンバーの後に追加する必要があり、これを追加しても 2 の累乗の境界は交差しません。「Other Peripherals」という新しいメンバーは、2 つの子を持ちます。「Systems」(「Other Peripherals」の兄弟)は、2 つの子を持ちます。子分岐を追加しても、同じレベルの兄弟メンバーの子では 2 の累乗の境界内のままになります。アウトラインを保存すると、Essbase では簡易再構築が実行されます。



3 つの子メンバーを持つ子分岐を追加すると、2 の累乗の境界が交差し、Essbase によるすべてのアウトラインの再構築が必要になる場合があります。ただし、「Systems」にすでに 3 つのメンバーがある場合、2 の累乗の境界は 4 になり、すべてのアウトラインを再構築せずに、最大 4 つの子を「Other Peripherals」に追加できます。

集約ストレージ・データベースのエクスポート

集約ストレージ・データベースに対する読取り権限がある場合は、レベル 0 のデータをデータベースから指定したテキスト・ファイルにエクスポートできます。このエクスポート・ファイルには、制御、アウトラインまたはセキュリティ情報ではなく、圧縮されていないデータのみが含まれます。データのエクスポート中、ユーザーは、Essbase サーバーに接続して、データベースで読取り専用の操作を実行できます。

アウトラインが変更されていない場合は、ルール・ファイルがなくてもエクスポートされたデータを再ロードできます。次の理由の場合は、データのエクスポートを検討してください:

- プラットフォーム間でデータを転送する場合
- エクスポートされたファイルを、バイナリではなくテキストのフォーマットで作成する場合
- バックアップを作成する場合

エクスポート

エクスポート・ファイルのデフォルトの場所は、ARBORPATH/app/です。代替の場所を指定できます。Oracle Essbase Administration Services Online Help または『Oracle Essbase テクニカル・リファレンス』を参照してください。

集約ストレージ・データベースのエクスポートには、次のような制限があります:

- エクスポートできるのは、レベル 0 のデータ(入力データ)のみです。
- 列のエクスポートは実行できません。縦欄式エクスポートでは、出力ファイルの各行に、各次元のメンバー名が表示されます(また、名前が行ごとに繰り返されることがあります)。

2 GB を超えるエクスポート・ファイルが作成されないようにするため、名前に番号の接尾辞(_1、_2 など)が含まれている複数のエクスポート・ファイルが Essbase によって作成されることがあります。たとえば、最初のファイル名が/home/exportfile.txt の場合、次のファイルは/home/exportfile_1.txt になります。

パフォーマンスの向上のため、データは並行してエクスポートできます。

➤ データをエクスポートするには、次のツールを使用します:

ツール	トピック	場所
Administration Services	データベースのエクスポート	Oracle Essbase Administration Services Online Help
MaxL	**export data**	『Oracle Essbase テクニカル・リファレンス』

第 XI 部

付録

付録の内容：

- 制限
- サンプル・アプリケーションの設定
- Essbase エラーのトラブルシューティング
- ディスク要件およびメモリー要件の見積り
- ESSCMD の使用
- Essbase アプリケーション、データベース、次元、メンバーおよび別名の命名規則
- 異なるセキュリティ・モデルにおけるセキュリティ関連操作の相違
- Essbase 32 ビットおよび 64 ビットの互換性

A 制限

この付録の内容

この付録では、「非 Unicode のアプリケーション」という用語は、非 Unicode モードのアプリケーションと、Unicode に対応していない Essbase サーバー上のアプリケーションを指します。第 43 章「Essbase の Unicode 実装の理解」を参照してください。

名前と関連アーティファクトの制限

表 236 に、名前と関連アーティファクトの制限をリストします。

表 236　名前および関連アーティファクト

アーティファクト	制限
別名	● 非 Unicode アプリケーションの制限: 80 バイト ● Unicode モード・アプリケーションの制限: 80 文字
別名テーブルの名前	● 非 Unicode アプリケーションの制限: 30 バイト ● Unicode モード・アプリケーションの制限: 30 文字
アプリケーション名	● 非 Unicode アプリケーションの制限: 8 バイト ● Unicode モード・アプリケーションの制限: 30 文字
アプリケーションの説明	● 非 Unicode アプリケーションの制限: 79 バイト ● Unicode モード・アプリケーションの制限: 80 文字

アーティファクト	制限
● カスタム定義関数名 ● カスタム定義マクロ名 ● カスタム定義関数の定義 ● カスタム定義マクロの定義	● 非 Unicode アプリケーションの制限: 127 バイト。 MaxL と API では、文字は 127 バイトまでに切り捨てられます。 ● Unicode モード・アプリケーションの制限: 128 文字。 MaxL と API では、文字は 128 文字までに切り捨てられます。 どちらの場合も、サーバーでは切捨ては行われません。切捨てが行われても、エラーは表示されません。
カスタム定義関数およびマクロのコメント	● 非 Unicode アプリケーションの制限: 255 バイト。 MaxL と API では、文字は 255 バイトまでに切り捨てられます。 ● Unicode モード・アプリケーションの制限: 256 文字。 MaxL と API では、文字は 256 文字までに切り捨てられます。 どちらの場合も、サーバーでは切捨ては行われません。切捨てが行われても、エラーは表示されません。
データベース名	● 非 Unicode アプリケーションの制限: 8 バイト ● Unicode モード・アプリケーションの制限: 30 文字
データベースの説明	● 非 Unicode アプリケーションの制限: 79 バイト ● Unicode モード・アプリケーションの制限: 80 文字
ディレクトリ・パス 例: `EPM_ORACLE_HOME/products/Essbase/EssbaseServer/bin`	● 非 Unicode アプリケーションの制限: 255 文字 ● Unicode モード・アプリケーションの制限: 1024 文字
環境変数名	● 非 Unicode アプリケーションの制限: 320 バイト ● Unicode モード・アプリケーションの制限: 320 文字
環境変数値	● 非 Unicode アプリケーションの制限: 256 バイト ● Unicode モード・アプリケーションの制限: 256 文字
Essbase サーバー名	● 非 Unicode アプリケーションの制限: 1024 バイト ● Unicode モード・アプリケーションの制限: 1024 文字
計算スクリプト、レポート・スクリプトおよびルール・ファイルのファイル名	● 非 Unicode アプリケーションの制限: 8 バイト ● Unicode モード・アプリケーションの制限: ○ パスに含める場合は、次の 2 つの値のいずれか小さい方になります: ❑ 1024 バイト ❑ オペレーティング・システムによって確定されている制限 ○ パスに含まれない場合、いくつかの MaxL ステートメントでのように、1024 バイトになります。
テキスト・ファイルおよび Excel ファイルのファイル名	非 Unicode アプリケーションの制限: 8 バイト
フィルタ名	● 非 Unicode アプリケーションの制限: 256 バイト ● Unicode モード・アプリケーションの制限: 256 文字

アーティファクト	制限
グループ名	● 非 Unicode アプリケーションの制限: 256 バイト ● Unicode モード・アプリケーションの制限: 256 文字
LRO セル・ノート	● 非 Unicode アプリケーションの制限: 600 バイト ● Unicode モード・アプリケーションの制限: 600 文字
LRO URL	512 文字(シングル・バイト文字のみ使用可)
URL およびファイルの LRO の説明	● 非 Unicode アプリケーションの制限: 80 バイト ● Unicode モード・アプリケーションの制限: 80 文字
メンバーのコメント・フィールド	● 非 Unicode アプリケーションの制限: 255 バイト ● Unicode モード・アプリケーションの制限: 256 文字
メンバー名	● 非 Unicode アプリケーションの制限: 80 バイト ● Unicode モード・アプリケーションの制限: 80 文字
修飾メンバー名の制限: 修飾メンバー名に指定できるレベル数	● 20 レベル
パスワード	● 非 Unicode アプリケーションの制限: 100 バイト ● Unicode モード・アプリケーションの制限: 100 文字
ランタイム代替変数名	320 バイト
ランタイム代替変数値	256 バイト
代替変数名	320 バイト
代替変数名	256 バイト
テキスト・リスト・オブジェクト内のテキスト値の最大数	1024
テキスト値の名前の最大長	80
テキスト値の ID の数値の範囲	-32768 から 32767
トリガー名	● 非 Unicode アプリケーションの制限: 30 バイト ● Unicode モード・アプリケーションの制限: 30 文字
UDA	● 非 Unicode アプリケーションの制限: 80 バイト ● Unicode モード・アプリケーションの制限: 80 文字
ユーザー名	● 非 Unicode アプリケーションの制限: 256 バイト ● Unicode モード・アプリケーションの制限: 256 文字

データ・ロードおよび次元構築の制限

表 237 に、データのロードと次元の構築に関連する制限をリストします。

表 237　データ・ロードおよび次元構築の制限

アーティファクト	制限
選択条件および除外条件	選択基準と除外条件を記述する文字の数: すべての基準の組合せは 32 KB に制限されている
データ・ロードまたは次元構築エラー・ログに書き込まれるエラー・メッセージの数(`essbase.cfg` の DATAERRORLIMIT)	デフォルトは 1000、最小数は 1、最大数は 65000
重複メンバー次元の構築時に許可される生成の最大数	20
増分次元ビルドのソース・ファイルの最大数	64

集約ストレージ・データベースの制限

表 238 に、集約ストレージ・データベースに関連する制限をリストします。

表 238　集約ストレージの制限

アーティファクト	制限
集約ストレージ・データベースの単一次元内の階層数	65535 (すべての保管階層、動的階層およびこの次元をベースとするすべての属性次元を含む)
集約ストレージ・アウトライン内のメンバー数	10,000,000 - 20,000,000 メンバー。使用可能なメモリーおよび他のメモリー要件に依存
集約ストレージ・データベースでのファイル位置サイズの最大	4,294,967,295 MB

アーティファクト	制限

集約ストレージ・アウトライン内に保管されている次元レベルの組合せの数	保管次元全体の次元レベルの要素の積は、2^{52} を上回ることはできません。これは 4,503,599,627,370,496 と等しくなります。
	アウトライン内に保管されている次元レベルを計算するには:
	1. 各次元の保管レベルの係数を決定します。
	2. 次元の係数を乗算します。たとえば、a * b * c * d などです。
	各次元の係数を決定するには、次のガイドラインに従います:
	● 動的次元の場合、係数は 1 です。
	● 保管次元の場合、最初の次元の係数で、ラベルのみ以外のレベルをカウントします。
	❍ 保管される次元に属性次元がある場合は、各属性次元内のレベルをカウントし、その数を次元の係数に加算します。次に、その次元のすべてのレベル 0 のメンバーに特定の属性次元からの属性がある場合は、これが行われる属性次元ごとに 1 を減算します。
	❍ 保管される次元に追加の保管階層がある場合は、各保管階層のラベルのみ以外のレベルの数をカウントし、その数を次元の係数に加算します。レベルが共有メンバーのみで構成される場合は、そのレベルをカウントしないでください。
	❍ 次元係数は、保管されている次元の動的階層の影響は受けません。
	たとえば、ASOsamp.Sample の 11 個の次元の係数を乗算すると、1 * 1 * 4 * 2 * 2 * 2 * 3 * 2 * 6 * 7 * 7 は 56,448 に等しくなります。
	次元レベルの要素を表示するには:
	● 管理サービス・コンソールで、データベースを右クリックし、「編集」、「プロパティ」の順に選択します。「統計」タブの「集約ストレージ統計」に、各次元の保管済レベルの数(つまり次元レベルの要素)が表示された、アウトライン内の各非属性次元がリストされます。例:
	`Dimension [Products] has [6] levels, bits used 7`
	● 次の MaxL コマンドを使用します:
	`query database appName . dbName list aggregate_storage runtime_info;`

非常に疎なデータ・セットを持つ集約ストレージ・データベースでクエリーできるセルの最大数	2^{64} この制限は、Essbase によってクエリーで処理されるセル数に基づいています。これは、すべての次元のメンバーの積であり、出力レポートに含まれているセルの数ではありません。 この制限は、クエリーのサイズと比較して相対的に小さいレポートを生成するために、欠落したデータを抑制するクライアント・インタフェース(Oracle Hyperion Financial Reporting など)を使用しているときに検出されることがあります。
別名テーブルの数	32

アーティファクト	制限
並列エクスポート・スレッドの数	8 Essbase でエクスポート・スレッドが使用される方法の詳細は、『Oracle Essbase テクニカル・リファレンス』のブロック・ストレージ・データベースに対する data export MaxL コマンドと、EXPORTTHREADS 構成設定を参照してください。

ブロック・ストレージ・データベースの制限

表 239 に、ブロック・ストレージ・データベースに関連する制限のリストを示します。

表 239　ブロック・ストレージ・データベースの制限

アーティファクト	制限
ブロック・ストレージ・アウトライン内のメンバー数	Essbase アウトラインに明示的に定義されている約 1,000,000 のメンバー。 ハイブリッド分析を使用している場合、および特定の方法でパーティションを使用している場合は、アウトラインで明示的にリストされているメンバーよりも多くのメンバーへのアクセスが可能です; データベースでアクセスできる実際のメンバー数はさらに多くなります。 マルチバイト文字を使用しているとメンバー名前が長くなる場合がありますが、許可されるメンバー数が少なくなります。
ブロック・ストレージ・データベースの密データ・ブロックにおけるゼロ以外の値の数	32 ビットおよび 64 ビット: 2^{28} -1 (268,435,455)
ブロック・ストレージ・データベース内の可能な疎ブロック数	疎次元の保管済メンバー数の積は、次の値以下にする必要があります: ● 32 ビット: 2^{104} ● 64 ビット: 2^{128} ブロック・ストレージ・データベースでの可能な疎ブロック数の算出方法は1161 ページの「潜在的なデータ・ブロック数」を参照してください。
ブロック・ストレージ・アウトライン内の 1 ブロック当たりの可能なセル数(アウトライン保存時のセルの上限)	32 ビットおよび 64 ビット: 2^{52}
ブロック・ストレージ・データベース内のリンク・レポート・オブジェクト数	32 ビットおよび 64 ビット: 2^{104} **注：**　セルに添付されたリンク・レポート・オブジェクトを表示または編集するときに、アウトラインに非保管済メンバーがある場合、オブジェクト数が 2^{104} にならないうちに制限に達する可能性があります。この場合、「エラー 1140013 無効なセル・アドレスが入力されました。」というエラーが表示されます。この制限を回避するには、アウトライン内の次元を並べ替えます。
別名テーブルの数	32

アーティファクト	制限
並列エクスポート・スレッドの数	1024 Essbase でエクスポート・スレッドが使用される方法の詳細は、『Oracle Essbase テクニカル・リファレンス』の集約ストレージ・データベースに対する data export MaxL コマンドと、EXPORTTHREADS 構成設定を参照してください。

Oracle アプリケーションへのドリルスルーの制限

表 240 に、ドリルスルー URL に関連する制限のリストを示します。

表 240　Oracle アプリケーションへのドリルスルーの制限

アーティファクト	制限
データベース当たりのドリルスルー URL の数	255
ドリルスルー URL のドリル可能領域の数	256
ドリル可能領域当たりの文字数	65536

その他のサイズ制限または数の制限

表 241 に、その他のサイズ制限と数の制限のリストを示します。

表 241　その他のサイズ制限または数の制限

アーティファクト	制限
式のサイズ	65534 バイト
セキュリティ・フィルタの数	• Essbase サーバー当たり: 65535 • Essbase データベース当たり: 32290
ユーザーおよびグループの数(結合)	30,000 この制限を超えるとエラーになります。
集約に含まれる集約ビューの数	1023
ルール・ファイルの最大レコード・サイズ	64 KB
MDX 最大クエリー制約条件およびセット・サイズ	『Oracle Essbase テクニカル・リファレンス』の MDX クエリー制約条件のトピックを参照してください。
計算スクリプトと式における変数配列の最大サイズ。配列では、MEMBERS、NUMBERS、STRINGS などの計算関数、またはそれらに関連付けられている値に一時的に引数が格納されます。	最大の整数値は 2,147,483,647
MaxL ステートメントの最大長	81,920 文字(バイト)

B サンプル・アプリケーションの設定

この付録の内容

はじめに

Essbase サーバーのインストールには、Essbase 機能に精通できるように設計されている、サンプル・アプリケーションが含まれています。各アプリケーションには 1 つ以上のデータベースが含まれ、各データベースには関連するデータ・ロード・ファイルが含まれています。表 242 を参照してください:

表 242　サンプル・データベースとデータ・ロード・ファイル

アプリケーション	データベース	データ・ロード・ファイル
Sample Interntl データベースと Xchgrate データベースでは、Essbase の通貨換算機能の例が示されています。	Basic Interntl Xchgrate	Calcdat.txt Currcalc.txt Rates.txt
Demo 『Oracle Essbase テクニカル・リファレンス』およびレポート・ライターのドキュメンテーションを参照してください。	Basic	Data.txt
Samppart Essbase のパーティション化機能の例を示します。第 15 章「パーティション・アプリケーションの設計」を参照してください。	会社	Calccomp.txt
Sampeast Essbase のパーティション化機能の例を示します。第 15 章「パーティション・アプリケーションの設計」を参照してください。	East	Calceast.txt
Sample_U Sample アプリケーションの Basic データベースの Unicode モードのバージョンが含まれています。これには、英語およびその他 4 つの文字セットの別名テーブルが含まれます。このアプリケーションの文字は UTF-8 でエンコードされています。	Basic	Calcdat.txt

アプリケーション	データベース	データ・ロード・ファイル
ASOsamp 集約ストレージ機能の例を示します。	Sample	dataload.txt[1]

[1]ASOsamp.Sample データベースは、dataload.rul ルール・ファイルを使用してロードする必要があります。その他のサンプルでは、ルール・ファイルは必要ありません。

サンプル・データベースへのデータのロード

サンプル・アプリケーションを使用する場合、その前にアプリケーションにデータをロードしておく必要があります。

注： Partitioning サンプル・データベースにデータをロードする場合は、始めにパーティション・ユーザーまたは他のユーザー名を作成する必要があります。1153 ページの「パーティション・ユーザーの作成」を参照してください。

➤ サンプル・データベースにデータをロードするには:

1. 管理サービス・コンソールを開きます。
2. エンタープライズ・ビューまたはカスタム・ビューで、対応する Essbase サーバー、アプリケーションおよびデータベースに移動します。
3. データベースを右クリックして、「データのロード」を選択します。
4. データのロード・オプションを設定します。
   - 「データ・ソース・タイプ」で、「データ・ファイル」を選択します。
   - 「モード」で、次のオプションのいずれかを選択します:
     - 「ロードのみ」: データ・ロードを実行します。
     - 「ビルドのみ」: 次元構築を実行します。
     - 「両方」: データ・ロードと次元構築を両方とも実行します。
   - 「データ・ソース」では、「データ・ファイルの検索」をクリックします。「Essbase サーバー」タブで、適切なデータ・ソースに移動します。「ファイルのタイプ」で「データ・ファイル(*.txt)」が選択されていることを確認します。
   - 「ルール・ファイル」では、データ・ソースにルール・ファイルが必要な場合は、「ルール・ファイルの検索」をクリックします。「Essbase サーバー」タブで、ルール・ファイルに移動します。「ファイルのタイプ」で「ルール・ファイル(*.rul)」が選択されていることを確認します。

     注： ASOsamp データベースは、dataload.rul ファイルを使用してロードする必要があります。その他のサンプルでは、ルール・ファイルは必要ありません。

データ・ロードが完了すると、ロードされたファイルに関する情報と、エラーが発生したかどうかを示すダイアログ・ボックスが表示されます。

サンプル・アプリケーションへのユーザー・アクセスの提供

Essbase には、複数ユーザーの環境をセキュリティで保護するための包括的なセキュリティ・システムが用意されています。デフォルトでは、サンプル・アプリケーションはセキュリティ・アクセス・レベル「なし」で作成されます。これは、ユーザーが管理者として定義されている場合を除き、どのユーザーもサンプル・データベースにはアクセスできないことを意味します。

システム管理者は、Essbase のインストール時に定義され、自動的に管理者権限を保持します。したがって、システム管理者は、サンプル・アプリケーションを他のユーザーが使用できるようにすることが可能です。

➤ すべてのユーザーにサンプル・アプリケーションへの書込みアクセス権を与える手順は、次のとおりです:

1. **システム管理者アカウントを使用して、管理サービス・コンソールにログオンします。**
2. **エンタープライズ・ビューで、目的の Essbase サーバーおよびアプリケーションを探します。**
3. **アプリケーションを選択して右クリックし、「プロパティの編集」を選択します。**
4. **「アプリケーション・プロパティ」で、「一般」タブを選択します。**
5. **「最小アクセス・レベル」で「書込み」を選択します。**

   たとえば、アプリケーションのすべてのデータベースに対して少なくとも書込みアクセス権をすべてのユーザーに与える(つまり、すべてのユーザーがデータ値を更新できるようにする)には、「書込み」を選択します。
6. **「最小データベース・アクセス」グループから「書込み」を選択します。**
7. **「適用」をクリックします。**
8. **データベースごとにこの手順を繰り返します。**

これで選択したアプリケーションを使用できるようなります。別のアプリケーションにアクセスする場合は、この手順を繰り返します。

集約ストレージのサンプル・アプリケーションの準備

➤ サンプルの集約ストレージ・アプリケーションを使用するように準備をするには:

1. **次のデータおよびルール・ファイルを使用して、データをロードします:**

```
ARBORPATH
/app/ASOsamp/Sample/dataload.txt


ARBORPATH
/app/ASOsamp/Sample/dataload.rul
```

集約ストレージ・データベースにデータをロードする方法は、Oracle Essbase Administration Services Online Help を参照してください。

2 **次のいずれかの方法を使用して、取得時間を改善するためにデータベース上の集約を事前計算します:**

- Administration Services の集約設計ウィザード。

  Oracle Essbase Administration Services Online Help を参照してください。

- MaxL の **execute aggregate process** ステートメント。

  『Oracle Essbase テクニカル・リファレンス』を参照してください。

3 **`essbase.jar` ファイルが `ESSBASEPATH/java` ディレクトリ内にあることを確認します。**

サンプルのパーティション・アプリケーションの準備

環境の設定

Essbase には、パーティション化の機能の例を示すサンプル・アプリケーションとデータベースが 2 つあります:

- Samppart - Company
- Sampeast - East

Partitioning アプリケーションとデータベースには、`.ddb` ファイルに保管されているパーティション定義が含まれています。`.ddb` ファイルは、ターゲット・データベース「Company」とソース・データベース「East」のメンバーの組合せ間のマップを定義します。

Partitioning アプリケーションを使用している環境で動作させるには、partitionuser という名前のユーザーを作成するか(1153 ページの「パーティション・ユーザーの作成」を参照)、`.ddb` ファイル内の埋込みユーザー名を変更する(1153 ページの「サンプル・パーティション定義に埋め込まれているユーザー名の変更」を参照)ことが必要な場合があります。

注: ユーザー名情報は、`.ddb` ファイル内で直接変更しないでください。

パーティション・ユーザーの作成

Samppart アプリケーションおよび Sampeast アプリケーションを使用する前に、partitionuser というユーザー名を作成する必要があります。このユーザーは、両方のアプリケーションのアプリケーション・マネージャ・アクセス権を持つ必要があります。

➤ partitionuser を作成するには:

1. システム管理者アカウントを使用して、管理サービス・コンソールにログオンします。
2. エンタープライズ・ビューで、管理サーバーのノードに移動し、適切な Essbase 管理サーバー名を選択します。
3. 「ユーザー」ノードを右クリックして、「ユーザーの作成」を選択します。
4. 「Essbase 管理サーバー」「でのユーザーの作成」では、「ユーザー名」に partitionuser と入力します。
5. 「パスワード」にパスワードを入力します。
6. 「パスワードの確認」にパスワードをもう一度入力します。
7. 「管理者権限」は、「TRUE」を選択します。
8. 「OK」をクリックします。

サンプル・パーティション定義に埋め込まれているユーザー名の変更

partitionuser を作成しないと選択した場合は、ユーザーに管理者権限があれば、パーティション定義ファイル(.ddb)に埋め込まれているユーザー名を選択したユーザー名に変更できます。Samppart と Sampeast に同梱されている .ddb ファイルは、サーバー名「localhost」をベースにしています。

➤ Samppart.Company および Sampeast.East の .ddb ファイルのユーザー名を変更するには:

1. 管理サービス・コンソールを開きます。
2. エンタープライズ・ビューで、「アプリケーション」ノードを展開し、「Samppart」アプリケーションを選択します。
3. 「データベース」ノードを展開し、「Company」データベースを選択します。
4. 「パーティション」ノードを展開し、「ソース・データベース」を選択してからダブルクリックします:

```
servername:Sampeast:East [transparent]
```

5. 「データ・ソース」および「データ・ターゲット」に、「ユーザー」リストからユーザー名を選択します。
6. パスワードを入力します。

7 「修正」をクリックして、変更内容を保存します。

注： 「パーティションの修復」ダイアログ・ボックスで、「データ・ソース」グループおよび「データ・ターゲット」グループのユーザー名を変更した場合は、Sampeast.East の .ddb ファイルでユーザー名を変更するためにこのプロセスを繰り返す必要はありません。

C

Essbaseエラーのトラブルシューティング

この付録の内容

致命的なエラーの処理の理解

Essbase サーバー・カーネルは、次の場合に致命的なエラーが発生したと判断します:

- 1 つ以上の制御フィールドに、予期しない値または不整合な値がある場合。
- カーネルで、データの破損が検出された場合。
- カーネルで、データの整合性の確認に必要な操作を実行できない場合(ディスク・スペースが不十分な場合など)。
- データが破損する可能性がある状態をカーネルが検出した場合。

カーネルで致命的なエラーが検出されると、シャット・ダウンしてから再起動して、それ自体の再初期化とデータベース・リカバリの続行が試みられます。復元が始まると、Essbase で次のようなエラー・メッセージが表示されます:

```
1080022 Reinitializing the Essbase Kernel for database
dbname
 due to a fatal error...
```

このメッセージに続いて、データベースのリカバリに関する次のような情報メッセージが表示されます:

```
1080028 Performing transaction recovery for database
dbname
 during fatal error processing.
```

このようなメッセージが表示された場合、カーネルはシャット・ダウンされ、再起動が試みられています。Essbase サーバー・ログを確認し、再初期化メッセージが生成される直前に Essbase で致命的なエラー・メッセージが発行されたかどうか確認してください。814 ページの「Essbase サーバー・ログおよびアプリケーション・ログの使用」を参照してください。

カーネルで致命的なエラーが検出された場合、通常はその致命的なエラーの発生時にアクティブだった処理を再開する必要があります。その処理が計算またはデータ・ロードであった場合は、処理が中断されたところから続行できる場合もあります。Essbase サーバー・ログを確認して、Essbase の処理がどこまで行われたかを確認してください。不確かな場合は、処理を再開してください。875 ページの「サーバーの中断が発生した場合に予測される結果」を参照してください。

カーネルで重大なエラーが検出されていなかった場合には、ソフトウェア・プロバイダの技術サポート部門に問い合せて、カーネルのシャット・ダウンと再起動の原因を特定してください。

Essbase サーバー・ログのコンテンツについての説明は、801 ページの「Essbase サーバー・ログのコンテンツ」を参照してください。エラーが発生したコンポーネントの特定の詳細は、812 ページの「Essbase サーバー・ログおよびアプリケーション・ログのメッセージのカテゴリ」を参照してください。

密な再構築の障害からの回復

Essbase による再構築中にエラーやシステム障害が発生する場合、このようなことは 936 ページの「密な再構築」の手順 2 で発生する可能性があります。

➤ 936 ページの「密な再構築」の手順 2 の間の障害から回復するには:

1. **一時ファイルを削除してディスク・スペースを解放し、次のデータベース再構築時の競合を回避します。**
2. **データベースを再起動します。**

➤ 936 ページの「密な再構築」の手順 1、手順 3 または手順 4 の間の障害から回復するには:

1. **ディスクのディレクトリを確認して、再構築がどこまで処理されたかを確認します。**
2. **手順 4 以外のすべての処理が完了している場合は、一時ファイルの名前を適切なファイル名に変更します。**

疎な再構築で障害が発生した場合の回復

疎な再構築の手順のいずれかでシステム障害が発生した場合は、データベースを再起動することによって回復できます。

レポート・スクリプトのメンバー名とデータベース・アウトラインのメンバー名の同期化

レポートを実行するときは、レポート・スクリプト内のメンバー名をデータベース・アウトライン内のメンバー名に必ず一致させてください。レポート・エクストラクタで一致するメンバー名を検出できない場合は常に、エラーが表示されます。レポートの処理を続行する前に、レポート・スクリプト内の名前を修正する必要があります。

複数のレポートを実行する場合における Essbase サーバーの問題の処理

割り当てられているサーバー・スレッド・リソースよりも多くのレポートを並行して実行しようとすると、サーバー・コンピュータがフリーズする場合があります。

複数のレポート・スクリプトの実行中にサーバーがフリーズした場合は、サーバー上の `essbase.cfg` ファイルの構成ファイル設定 SERVERTHREADS の値を確認してください。実行中の各レポートには、少なくとも 1 つのスレッドが存在している必要があります。たとえば、22 個のレポートを実行している場合、SERVERTHREADS の値は少なくとも 22 である必要があります。

参照

パーティション、通貨換算およびデータ・ロードに関する章には、基本的なトラブルシューティングの情報が記載されています。

バックアップからの復元の詳細は、『Oracle Hyperion Enterprise Performance Management System バックアップおよびリカバリ・ガイド』を参照してください。

特定のエラー・メッセージの詳細は、『Oracle Essbase エラー・メッセージ・リファレンス』を参照してください。

エラーと例外の詳細は、800 ページの「Essbase ログの使用」を参照してください。

D

ディスク要件およびメモリー要件の見積り

この付録の内容

はじめに

この付録では、計算対象の多数のコンポーネントの記録に役立つ、ワークシートによるアプローチを使用します。このドキュメントの印刷版をご使用の場合は、ワークシートをコピーできます。それ以外の場合は、別紙でワークシートをシミュレートできます。DA や MA などのラベルは、計算済の様々なディスクおよびメモリーのコンポーネント値の記録に役立ちます。

Essbase でのデータの保管方法の理解

データベースのサイズを決定するには、Essbase で使用されるストレージのユニットを理解する必要があります。ここでは、次の基本的な概念に精通していることを前提として説明します:

- Essbase でのデータの保管方法(第 3 章「多次元データベースの理解」を参照)
- Essbase Kernel コンポーネントによる Essbase データベースの管理方法(第 49 章「データベース設定の管理」を参照)

Essbase のデータベースは多数のコンポーネントで構成されています。Essbase では、アウトライン・ファイルとデータ・ファイルに加え、いくつかのタイプのファイルとメモリー構造を使用して、データ・ストレージ、計算および取得操作の管理を行います。

表 243 で、データベースのディスクとメモリーの要件を見積るときに考慮すべき主要コンポーネントについて説明します。「はい」は示されているタイプのストレージに関連があることを意味し、「いいえ」は関連がないことを意味します。

表 243 ディスクとメモリーの要件の計算で考慮すべきストレージ・ユニット

ストレージ・ユニット	説明	ディスク	メモリー
アウトライン	データベースのすべての要素を定義する構造。アウトライン内のメンバーの数により、アウトラインのサイズが決まります。	はい	はい
データ・ファイル	Essbase によってデータ・ファイル内のデータ・ブロックのデータ値が保管されるファイル。 essxxxxx.pag という名前で、xxxxx は番号。Essbase では、この番号は各ディスク・ボリュームで ess00001.pag から始まり、1 つずつ増加します。このファイルは Essbase によってメモリーにコピーされるため、メモリーも影響を受けます。	はい	はい
データ・ブロック	データ・ファイルのサブディビジョン。各ブロックは、疎次元の特定の交差に関連する、すべての密次元のすべてのセルを示す、多次元の配列です。	はい	はい
インデックス・ファイル	Essbase でデータ・ブロックをデータ・ファイルから取得するために使用されるファイル。essxxxxx.ind という名前で、xxxxx は番号。Essbase では、この番号は各ディスク・ボリュームで ess00001.ind から始まり、1 つずつ増加します。	はい	はい
インデックス・ページ	インデックス・ファイルのサブディビジョン。データ・ブロックをポイントするインデックス・エントリが格納されます。インデックス・ページのサイズは 8 KB に固定されています。	はい	はい
インデックス・キャッシュ	インデックス・ページを保持するメモリー内のバッファ。Essbase はデータベースの起動時に、メモリーをインデックス・キャッシュに割り当てます。	いいえ	はい
データ・ファイル・キャッシュ	データ・ファイルを保持するメモリー内のバッファ。直接 I/O が使用されると、必要に応じて、データのロード、計算および取得操作中に Essbase によってメモリーがデータ・ファイル・キャッシュに割り当てられます。バッファ I/O では使用されません。	いいえ	はい
データ・キャッシュ	データ・ブロックを保持するメモリー内のバッファ。データ・ロード、計算および取得操作を行うと、必要に応じて Essbase によってメモリーがデータ・キャッシュに割当てられます。	いいえ	はい
計算機のキャッシュ	計算操作の際に、Essbase でデータ・ブロックを作成および追跡するために使用されるメモリー内のバッファ。	いいえ	はい

ディスク・スペース要件の決定

Essbase では、そのサーバー・ソフトウェア用と各データベース用にディスク・スペースを使用します。データベースのディスク・ストレージ要件を見積る前に、データベースに含まれている次元の数、次元の疎と密の度合い、各次元内のメンバーの数、および保管済メンバーであるメンバーの数がわかっている必要があります。

➤ データベースに必要なディスク・スペースを計算するには:

1 1161 ページの「ディスク要件のサイズ設定に使用される要素の計算」に示されている要素を計算します。

2 1 つのデータベースに必要なディスク・スペースを計算します。

1166 ページの「1 つのデータベースのディスク・スペース要件の見積り」を参照してください。

3 Essbase サーバーで必要な合計ディスク・スペースを計算します。

1174 ページの「Essbase サーバーの合計ディスク・スペース要件の見積り」を参照してください。

注： この章で示すデータベースのサイズ設定の計算は、データベース設計が最適でディスク・スペースに制限のない、理想的なシナリオを想定しています。ほとんどの多次元アプリケーションは疎のため、必要なスペースを正確に特定するのは困難です。

ディスク要件のサイズ設定に使用される要素の計算

データベースのディスク・スペース要件を見積る前に、見積りの計算に使用される要素を計算する必要があります。この章の後のほうでは、これらの値を使用してデータベースのコンポーネントを計算します。データベースごとに、そのコンポーネントのサイズを合計します。

表 244 に、これらの要素の計算方法について説明している項のリストを示します。示されている項に進んで、計算を実行してから、「値」列に計算された値を記入します。

表 244 ワークシート: データベースのディスク・スペース要件に影響する要素のリスト

データベースのサイズ計算要素	ラベル	値
1161 ページの「潜在的なデータ・ブロック数」	DA	
1163 ページの「既存のデータ・ブロック数」	DB	
1164 ページの「展開済データ・ブロックのサイズ」	DC	
1165 ページの「圧縮データ・ブロックのサイズ」	D	

潜在的なデータ・ブロック数

潜在的なデータ・ブロック数とは、データベースに格納できるデータ・ブロックの最大数のことです。

データベースがすでにロードされている場合、Administration Services の「データベース・プロパティ」ダイアログ・ボックスの「統計」タブにブロックの潜在数が表示されます。

データベースがまだロードされていない場合は、その値を計算する必要があります。

➤ 潜在的なデータ・ブロック数を決定する場合には、保管済メンバーのすべての組合せに対してデータ値が存在するものと仮定します。

1 **表 245 をワークシートとして使用して、それぞれの疎次元と、その保管済メンバーの数をリストします。8 つ以上の疎次元が存在する場合は、次元を他の場所にリストして、計算にすべての疎次元を含めてください。**

次のタイプのメンバーは、保管済メンバーではありません:

- 属性次元のメンバー
- 共有メンバー
- ラベルのみメンバー
- 動的計算メンバー(動的計算および保管メンバーは、保管済メンバーです)

表 245　ワークシート: 疎次元と保管済メンバーの数のリスト

疎次元の名前	保管済メンバーの数を記入
	a.
	b.
	c.
	d.
	e.
	f.
	g.

2 **最初の疎次元の保管済メンバーの数(a 行)に、2 番目の疎次元の保管済メンバーの数(b 行)を乗算し、さらに 3 番目の疎次元の保管済メンバーの数(c 行)を乗算し、というようにして各行の値を乗算します:**

```
     a * b * c * d * e * f * g (and so on)
= potential number of blocks
```

その結果の値を、表 244 の DA とラベル付けされたセルに記入します。

例

Sample.Basic データベースには、次の疎次元が含まれます:

- Product (保管済メンバー数は 19)
- Market (保管済メンバー数は 25)

潜在的データ・ブロック数は、19 * 25 = 475 となります。

既存のデータ・ブロック数

潜在的なブロック数と比較すると、既存のブロックとは、それらのデータ・ブロックのうち、Essbase で実際に作成されるブロックのことです。Essbase でブロックを作成するには、疎次元からの保管済メンバーの組合せに対して少なくとも 1 つの値が存在する必要があります。多数の組合せが欠落している可能性があるため、通常、既存のデータ・ブロック数は潜在的なデータ・ブロック数よりもかなり少なくなります。

➤ すでにロードされているデータベースの既存のブロック数を確認するには、Administration Services の「データベース・プロパティ」ダイアログ・ボックスの「統計」タブで既存のブロック数を探します。その値を、表 244 の DB とラベル付けされたセルに記入します。

データベースがまだロードされていない場合は、値を見積る必要があります。

➤ 既存のデータ・ブロックの数を見積るには:

1. **疎次元の保管されるメンバーの組合せのうちで、値を持つものの割合(データベースの密度)を見積ります。**
2. **想定されるデータ・ブロック数に対してこの割合を乗算します:**

```
     number of existing blocks
= estimated density
* potential number of blocks
```

実際のブロック数を、表 244 の DB とラベル付けされたセルに記入します。

例

次に、疎のレベルが異なる 3 つの例を示します。潜在的なデータ・ブロック数は 100,000,000 と仮定しています:

- きわめて疎: 潜在的なデータ・セルのうち 5 パーセントが存在します。

```
     .05 (estimated density)
* 100,000,000 (potential blocks)
= 5,000,000 existing blocks
```

- 疎: 潜在的なデータ・セルのうち 15 パーセントが存在します。

```
     .15 (estimated density)
* 100,000,000 (potential blocks)
= 15,000,000 existing blocks
```

- 密: 潜在的なデータ・セルのうち 50 パーセントが存在します。

```
     .50 (estimated density)
* 100,000,000 (potential blocks)
```

```
= 50,000,000 existing blocks
```

展開済データ・ブロックのサイズ

各データ・ブロックの潜在的な拡張(非圧縮)サイズは、ブロック内のセルの数と、各セルに使用されるバイト数に基づきます。セルの数は、密次元内の保管済メンバーの数に基づきます。Essbase では、ブロック内の交差する値のそれぞれを保管するため、8 バイトを使用します。

➤ ロードされているデータベースの既存のブロック数を確認するには、Administration Services の「データベース・プロパティ」ダイアログ・ボックスの「統計」タブで展開済データ・ブロックのサイズを探します。

データベースがロードされていない場合は、その値を見積る必要があります。

➤ 展開済データ・ブロックのサイズを決定するには:

1 **表 246 をワークシートとして使用し、それぞれの密次元と、その保管済メンバーの数を入力します。8 つ以上の密次元が存在する場合は、次元を他の場所にリストして、計算にすべての密次元を含めてください。**

次のタイプのメンバーは、保管済メンバーではありません:

- 属性次元のメンバー
- 共有メンバー
- ラベルのみメンバー
- 動的計算メンバー(動的計算および保管メンバーは、保管済メンバーです)

表 246　ワークシート: 密次元と保管済メンバーの数のリスト

密次元の名前	保管済メンバーの数
	a.
	b.
	c.
	d.
	e.
	f.
	g.

2 **最初の密次元の保管済メンバーの数(a 行)に、2 番目の密次元の保管済メンバーの数(b 行)を乗算し、さらに 3 番目の密次元の保管済メンバーの数(c 行)を乗算し、これを繰り返して、ブロック内のセルの総数を決定します:**

```
     a * b * c * d * e * f * g (and so on)
= the total number of cells
```

3 **結果のセル数に 8 バイトを乗算して、拡張ブロック・サイズを決定します:**

```
      (Total number of cells) * 8 bytes per cell
= expanded block size
```

その結果の値を、表 244 の DC とラベル付けされたセルに記入します。

例

Sample.Basic データベースには、次の密次元が含まれます:

- Year (保管済メンバー数は 12)
- Measures (保管済メンバー数は 8)
- Scenario (保管済メンバー数は 2)

次の計算を行い、Sample.Basic データベース内のデータ・ブロックの潜在的なサイズを決定します:

```
   12 * 8 * 2 = 192 data cells

192 data cells
* 8 bytes
= 1,536 bytes (potential data block size)
```

圧縮データ・ブロックのサイズ

圧縮は、データ・ファイルで使用される実際のディスク・スペースに影響します。4 つのタイプの圧縮(ビットマップ、ランレングス符号化(RLE)、zlib およびインデックス値)は、ディスク・スペースに対してそれぞれ異なる影響を与えます。サイズ要件の見積りに関係のないデータ圧縮の詳細は、853 ページの「データ圧縮」を参照してください。

圧縮を使用しない場合や、RLE 圧縮を使用可能にしている場合は、この計算を飛ばして1167 ページの「保管済データ・ファイル」に進んでください。

注: ブロック内には疎も存在するため、実際の(圧縮された)ブロック密度はブロックによって大きな違いがあります。ここで説明する計算は、見積りのみを目的としています。

➤ ビットマップ圧縮が使用可能な場合の平均圧縮ブロック・サイズを計算するには:

1 **ブロック密度の平均値を決定します。**

- データベースがロードされている場合は、Administration Services の「データベース・プロパティ」ダイアログ・ボックスの「統計」タブで展開済データ・ブロックのサイズを確認できます。「ブロックの密度」に表示される値を使用します。

- データをロードする前にブロック密度を見積る場合は、潜在的なデータ値に対する既存のデータ値の比率を評価します。

2 圧縮ブロック・サイズを決定するには、次の計算を実行します:

```
     expanded block size * block density
= compressed block size
```

結果のブロック・サイズを、表 244 の DD とラベル付けされたセルに記入します。

例

拡張ブロック・サイズを 1,536 バイト、ブロック密度を 25% (.25)と仮定します:

```
   1,536 bytes
*  .25
= 384 bytes (compressed block size)
```

1 つのデータベースのディスク・スペース要件の見積り

データベースのディスクスペース要件を見積るには、表 247 のコピーを作成するか、1 つのデータベース用のワークシートとして別個の用紙を使用します。サーバーに複数のデータベースがある場合は、データベースごとにこのプロセスを繰り返します。ワークシートにデータベースの名前を記入します。

このワークシートの各行は、そのコンポーネントのサイズ設定方法が記載されている項を示します。それぞれの計算を実行し、その結果を「サイズ」列の適切なセルに記入します。この計算では、表 244 に記入された要素を使用します。

表 247 ワークシート: データベースのディスク要件の見積り

データベース名:	
データベース・コンポーネント	**サイズ**
1167 ページの「保管済データ・ファイル」	DE
1169 ページの「インデックス・ファイル」	DF
1170 ページの「断片化許容量」	DG
1170 ページの「アウトライン」	DH
1172 ページの「作業領域」: DE + DF + DG + DH	DI
1173 ページの「リンク・レポート・オブジェクトに関する考慮事項」(必要な場合)	DJ

データベース名:	
データベース・コンポーネント	**サイズ**
データベースに必要な合計ディスク・スペース: `DI + DJ` このデータベースの名前および必要な合計ディスク・スペースを、表 248 に入力します。このワークシートによって、サーバーのすべてのデータベースで必要なディスク・スペースが決定されます(データベース用に、a から g までの 7 つのプレースホルダが用意されています)。	

サーバー上のデータベースごとにこれを繰り返します。サーバー上のすべてのデータベースのディスク・スペースの見積りが完了した後で、1174 ページの「Essbase サーバーの合計ディスク・スペース要件の見積り」に進みます。

保管済データ・ファイル

保管済データベースのサイズは、データベースが圧縮されているかどうか、およびそのデータベースの圧縮方法によって異なります。Essbase には、ビットマップ、ランレングス符号化(RLE)、zlib、インデックス値および圧縮なしの 5 つの圧縮方法があります。

圧縮データベースのサイズの計算は、次のような数多くの理由により複雑です:

- 使用する圧縮方法がブロックによって異なる場合があります。
- 場合によっては、Essbase により、ブロック・レベルで最適な方法とみなされるものが選択されます。

サイズ要件の見積りに関係のないデータ圧縮の包括的な説明は、853 ページの「データ圧縮」を参照してください。ここで説明する計算は、見積りのみを目的としています。

圧縮データ・ファイル(`essxxxxx.pag`)の保管に必要なスペースの計算には、次の要素を使用します:

- 既存のブロック数(表 244 の値 DB)
- 固定サイズのオーバーヘッド(1 ブロック当たり 72 バイト)
- 展開済データ・ブロックのサイズ(表 244 の値 DC)
- データベースの密度: 値を持つ疎次元の保管済メンバーの組合せの割合。(計算には、1163 ページの「既存のデータ・ブロック数」を参照。)

圧縮なしの計算

➤ 圧縮オプションが圧縮なしの場合にデータベース・サイズを計算するには:

```
Number of blocks * (72 bytes + size of expanded data block)
```

この結果を、表 247 の DE とラベル付けされたセルに記入します。1169 ページの「インデックス・ファイル」に進んでください。

圧縮データベースの計算

使用される圧縮方法はブロックごとに異なる可能性があるため、次の計算式は一般的なデータベース・サイズの見積りです。実際の実装は、この計算よりも大きい数や小さい数になることがあります。

ビットマップ圧縮

➤ 圧縮オプションがビットマップの場合にデータベース・サイズを見積るには、次の式を使用します:

```
    Number of blocks
* (72 bytes
+ (average size in bytes of an expanded data block)
* (1/64 + proportion of cells that are not #MISSING))
```

たとえば、1000 ブロックあり、拡張ブロックの平均サイズが 2000 バイトで、セルの 30%が#MISSING の場合、結果は次のようになります:

```
   1000 * (72 + 2000 * (1/64 + 0.7) )
= 1000 * 1503
= 1,503,000 bytes (approximately)
```

この結果を、表 247 の DE とラベル付けされたセルに記入します。1169 ページの「インデックス・ファイル」に進んでください。

インデックス値圧縮

➤ 圧縮オプションがインデックス値の場合にデータベース・サイズを見積るには、次の式を使用します:

```
    Number of blocks * (72 bytes
+ (1.5 * database density * expanded data block size)
```

この結果を、表 247 の DE とラベル付けされたセルに記入します。1169 ページの「インデックス・ファイル」に進んでください。

RLE 圧縮

➤ 圧縮オプションが RLE の場合にデータベース・サイズを見積るには、ビットマップ圧縮の計算式を使用します。

圧縮方法が RLE の場合、Essbase でより適切に圧縮されると判断できる場合は、ブロックに対して自動的にビットマップまたはインデックス値の方法が使用されます。ビットマップ計算を使用した見積りでは、最大サイズが見積られます。

この結果を、表 247 の DE とラベル付けされたセルに記入します。1169 ページの「インデックス・ファイル」に進んでください。

ZLIB 圧縮

➤ 圧縮オプションが zlib の場合にデータベース・サイズを見積るには、ビットマップ圧縮の計算式を使用します。

zlib 圧縮を使用した場合、データ・ブロックのサイズの決定は困難です。個々のブロックは、他の圧縮タイプを使用した圧縮に比べ、大きくなることも小さくなることもあります。いずれにせよビットマップ圧縮を使用して計算すれば、この演習で使用するための近似値が得られます。

この結果を、表 247 の DE とラベル付けされたセルに記入します。1169 ページの「インデックス・ファイル」に進んでください。

インデックス・ファイル

インデックス・ファイル(essxxxxx.ind)の保管に必要な容量の計算には、次の要素を使用します:

- 既存のブロック数(表 244 の値 DB)
- 112 バイト- インデックス項目のサイズ

インデックスの最小サイズは 8,216,576 バイト(8 MB)です。すべてのインデックス・ファイルを含むデータベース・インデックスのサイズを計算するには、次の計算を実行します:

```
number of existing blocks * 112 bytes = the size of database index
```

表 247 の DF とラベル付けされたセルに、計算の結果か 8,216,576 のどちらか大きい方の数を記入します。

例

次の例では、15,000,000 ブロックのデータベースを想定しています。

```
   15,000,000 blocks
* 112
= 1,680,000,000 bytes
```

注： データベースがすでにロードされている場合は、「データベース・プロパティ」ウィンドウで「ストレージ」タブを選択してサイズを参照します。

断片化許容量

ビットマップ圧縮または RLE 圧縮を使用している場合は、ある程度の断片化が発生します。その量はデータベースとオペレーティング・システムの個々の構成に基づき、正確に予測できません。

おおまかな見積りとして、圧縮されたデータベース・サイズ(表 247 の値 DE)の 20%を計算し、その結果を同じ表の DG とラベル付けされたセルに記入します。

例

圧縮データベース・サイズを 5,769,000,000 バイトと仮定した、断片化許容量の計算:

```
   5,769,000,000 bytes
* .2
= 1,153,800,000 bytes
```

アウトライン

アウトラインに必要なスペースには、2 つのコンポーネントが格納されることがあります。

- アウトラインの主要領域。ディスク要件およびメモリー要件の両方に関係するコンポーネントであり、次の要素を使用して計算します:
  - アウトライン内のメンバー数
  - メンバー名および別名の長さ(シングル・バイト換算での文字数)
- アウトラインの属性の関連付けにかかわる領域。ディスク・スペースのみに関係するコンポーネントであり、次の要素を使用して計算します:
  - 各基本次元内のメンバー数
  - 各属性次元内のメンバー数

➤ アウトライン・ファイルのサイズを見積るには:

1 **アウトラインの主要領域を見積るには、メンバー数にメンバー名の長さ(350 - 450 バイト)を掛けます。**

データベースに含まれている別名が少ない場合や、非常に短い別名と短いメンバー名が含まれている場合は、この範囲内でより小さい数を使用します。メンバー名や別名が非常に長いことがわかっている場合は、この範囲内でより大きい数を使用します。

名前の長さは概算平均であるため、次の式で求められる値は、アウトラインの主要領域のおおまかな見積りにすぎません:

```
number of members  * name-length factor  = size of main area of outline
```

注： メンバー名および別名の最大サイズは、付録 A「制限」を参照してください。

この章で後述するメモリー容量要件の計算に、アウトラインの主要領域のサイズを使用します。

2 **ディスク・スペース要件を求めるには、アウトラインに属性次元が含まれている場合は、データベースの属性関連付けの領域のサイズを計算します。この領域のサイズは、基本次元ごとに計算します。基本次元のメンバー数に、基本次元に関連付けられているすべての属性次元のメンバー数の合計を掛け、8 で割ります:**

```
(number of base-dimension members * sum of count of attribute-dimension
members)/8 = size of attribute association area for a base dimension
```

注： メンバー数には、ラベルのみメンバーおよび共有メンバーは含めないでください。

3 **各次元の属性の関連付け領域を合計し、アウトラインの属性の関連付け領域の合計を決定します。**

4 **アウトラインに必要な合計ディスク容量を求めるには、アウトラインの主要領域と属性の関連付け領域を加算します:**

```
main area of outline + total attribute association area
```

この計算の結果を、表 247 の DH とラベル付けされたセルに記入します。

例

アウトラインに次の特性があると仮定します:

- メンバー数は 26,000
- メンバー名の長さは 400 バイト
- 基本次元「Product」(ラベルのみメンバーと共有メンバーを除いたメンバー数が 23,000)には、「Ounces」(メンバー数 20)と「Pkg Type」(メンバー数 50)の 2 つの属性次元が関連付けられています
- 基本次元「Market」(ラベルのみメンバーと共有メンバーを除いたメンバー数が 2,500)には、属性次元「Population」(メンバー数 12)が関連付けられています

次の計算を行います:

1. アウトラインの主要領域を計算します:

```
name-length factor of 400 bytes  * 26,000 members  = 10,400,000 bytes
```

2. 属性の関連付け領域を計算します:
   - 基本次元「Product」について:

```
(23,000 * (20 + 50)) bits  / 8 bits per byte  = 201,250 bytes
```

- 基本次元「Market」について:

```
(2,500 * 12) bits  / 8 bits per byte  = 3,750 bytes
```

3. これらの領域を合計し、データベースの属性の関連付け領域の合計を求めます:

```
201,250 bytes + 3,750 bytes = 205,000 bytes
```

4. アウトライン・ディスク・スペースの概算合計を求めるには、アウトラインの主要領域と属性の関連付け領域の合計を足します:

```
10,400,000 bytes  + 205,000 bytes  = 10,605,000 bytes (outline disk space requirement)
```

注： この手順は、アウトラインのメモリー・スペース要件の計算には使用しないでください。1178 ページの「メモリー内で使用するアウトライン・サイズ」のアウトラインで説明するプロセスを使用してください。

作業領域

ディスク上の一時作業領域は、次の 3 つのプロセスで作成されます:

- Essbase では、リカバリ目的のため、ディスク上にデータ・リカバリ領域が保持されます。この領域のサイズは、データベースが再構築されるまで増大していきます。
- 再構築中、Essbase ではディスク上の再構築作業領域が使用されます。
- 前のリリースの Essbase からのアップグレード中は、リカバリ目的のため、Essbase により移行作業領域にデータベースのコピーが作成されます。

これらの一時作業領域を作成するには、Essbase でデータベース全体のサイズに等しいディスク・スペースが必要になる場合があります。再構築と移行には、アウトラインのサイズに追加の作業スペースが必要です。これらのアクティビティは同時に発生することはないため、単一の割当てによって 3 つの要件すべてに対応できます。

再構築、移行およびリカバリに使用される作業領域のサイズを計算するには、表 247 から次のデータベース・コンポーネントのサイズの合計を計算します:

- 圧縮データ・ファイル(値 DE)
- インデックス・ファイル(値 DF)
- 断片化許容量(値 DG)
- アウトライン(値 DH)

次の式を使用して、作業領域のサイズを計算します:

```
   work area = size of compressed data files
+ size of index files
+ fragmentation allowance
+ outline size
```

この計算の結果を、表 247 の DI とラベル付けされたセルに記入します。

リンク・レポート・オブジェクトに関する考慮事項

リンク・レポート・オブジェクト(LRO)機能を使用して、オブジェクト(アーティファクトともいう)をデータ・セルと関連付けることができます。オブジェクトには、フラット・ファイル、HTML ファイル、グラフィック・ファイルおよびセル・ノートがあります。第 11 章「Essbase データへのオブジェクトのリンク」を参照してください。

LRO は、次の 2 つの側面でディスク・スペースに影響します:

- オブジェクトのサイズ。Essbase では LRO として使用されるファイルがサーバーにコピーされるため、LRO として使用しているすべてのファイルのサイズ合計がわかっている必要があります。

  注: リンク・オブジェクトがファイルである場合(セル・ノートではない場合)、リンク・オブジェクトのサイズに対して制限を設定できます。997 ページの「LRO ファイル・サイズの制限」を参照してください。

- LRO カタログには、Essbase Kernel で LRO に関する情報が保管されます。セル・ノートと URL もこのカタログ内に保管されます。カタログ・エントリはインデックス・ページとして保管されます。Essbase では、カタログ・エントリごとに 8KB を使用します。

➤ LRO のディスク・スペース要件を見積るには:

1. **オブジェクトのサイズを見積ります。制限が設定されている場合は、LRO の数にその制限を掛けます。設定されていない場合は、予想されるすべての LRO のサイズを合計します。**
2. **LRO カタログのサイズを求めます。LRO の合計数に 8,192 バイトを掛けます。**
3. **2 つの領域を合計し、この計算の結果を、表 247 の DJ とラベル付けされたセルに記入します。**

例

データベースで、次のような構成の 1,500 個の LRO を使用していると仮定します:

- 1,000 個の URL (各 URL の最大バイト数は 512 バイト)
- 500 個のセル・ノート

次の計算を行います:

1. 1,000 * 512 の乗算によって、URL の保管に必要な最大バイト数 512,000 を求めます。
2. LRO カタログのサイズを計算します。1,500 個の合計 LRO * 8,192 バイト= 12,288,000 バイト。
3. 2 つの領域を合計します。次に例を示します:

```
      512,000 bytes
+ 12,288,000 bytes
= 12,800,000 bytes total LRO disk space requirement
```

1146 ページの「ブロック・ストレージ・データベースの制限」で、リンク・レポート・オブジェクトに関する考慮事項も参照してください。

Essbase サーバーの合計ディスク・スペース要件の見積り

この章の前述の計算は、単一データベースのデータ・ストレージ要件を見積るものです。多くの場合、サーバーには複数のデータベースが存在します。

サーバー上の Essbase データ・ストレージ要件の合計には、各データベースに必要なデータ・ストレージ以外に、Essbase ソフトウェアも含まれます。Essbase ソフトウェアおよびサンプル・アプリケーションの基本インストール用に、約 200 MB (209,715,200 バイト)を確保します。この容量はプラットフォームおよびファイル・システムによって異なります。

➤ サーバーの合計ディスク・スペース要件を見積るには:

1 表 248 のワークシートに、データベース名と、各データベースについて計算したディスク・スペース要件をリストします。

2 データベース要件を合計し、DL とラベル付けされたセルにバイトで記入します。

3 DM とラベル付けされたセルに、サーバー上にインストールするソフトウェアのディスク・スペース要件(たとえば 209,715,200 バイト)を記入します。

4 バイト単位の合計サーバー・ディスク・スペース要件について、セル DL および DM の値を合計します。この値を DN とラベル付けされたセルに記入します。

5 この値を 1,048,576 バイトで割ることで、セル DN の合計を MB に変換します。この値を DO とラベル付けされたセルに記入します。

表 248 ワークシート: 合計サーバー・ディスク・スペース要件の見積り

データベースのリスト [1]	サイズ
a.	
b.	
c.	
d.	

データベースのリスト[1]	サイズ
e.	
f.	
g.	
データベース・ディスク・サイズの合計: a + b + c + d + e + f + g	DL:
Essbase サーバー・ソフトウェアのサイズ(バイト)	DM:
Essbase サーバーの合計ディスク要件(バイト): DL + DM	DN:
Essbase サーバーの合計ディスク要件(MB): DN / 1,048,576 bytes	DO:

[1] 表 247 を参照してください

メモリー要件の決定

Essbase を実行するための最小メモリー要件は、64 MB です。UNIX システムの場合、最小要件は 128 MB です。アプリケーションおよびデータベースの数と、サーバーでのデータベース操作に基づいて、必要なメモリーが多くなる場合があります。

➤ データベースに必要なメモリーを計算するには:

1 **1176 ページの「メモリー要件のサイズ計算に使用される要素」に示されている要素を計算します。**

2 **アプリケーションのメモリー要件を計算します。**

1177 ページの「アプリケーションの起動用メモリー要件の見積り」を参照してください。

3 **単一のデータベースに必要なメモリーを計算します。**

1177 ページの「データベースの起動用メモリー要件の見積り」を参照してください。

Essbase サーバーに複数のデータベースが含まれている場合、各データベースに対してこれらの計算を実行します。

4 **Essbase サーバーの合計メモリー要件を計算します。**

1174 ページの「Essbase サーバーの合計ディスク・スペース要件の見積り」を参照してください。

メモリー要件のサイズ計算に使用される要素

見積りを始める前に、見積りの計算に使用する要素を計算します。

表 249 に、サイズを決定する要素と、これらのサイズ決定についての情報が記載されている参照先(この章と他の章の項)をリストします。指定されたセクションに移動して計算を実行し、表 249 に戻ります。このワークシートの「値」列に、バイト単位でサイズを書き込みます。

表 249 の値は、この章で後述する様々な計算に使用できます。

表 249　ワークシート: データベースのメモリー要件の計算に使用する要素

データベースのサイズ計算要素	値
論理ブロック内のセル数 1180 ページの「論理ブロック内のセル数」を参照してください。	MI:
SERVERTHREADS ESSCMD によって割り当てられたスレッドの数 『Oracle Essbase テクニカル・リファレンス』を参照してください。	MJ:
潜在的な保管されるブロックのサイズ 1164 ページの「展開済データ・ブロックのサイズ」を参照してください。	MK:

この章の計算は、使用されるメモリー量に複雑に影響する次の要素には対応していません。これらはここでの説明には含まれていません:

- キャッシュ・メモリーのロック。キャッシュ・メモリーのロックが使用可能かどうかは、オペレーティング・システムと Essbase でメモリーがどのように管理されるかに影響します。916 ページの「キャッシュ・メモリーのロックを使用するかどうかの決定」を参照してください。
- 様々な操作のタイプとそれらに関連するキャッシュ割当て。データ・ロード操作、データ取得操作および計算操作では、データ・ファイル、データ、計算キャッシュのためのメモリー、そしてキャッシュに関連したオーバーヘッド用のメモリーが確保されます。
- 取得バッファおよび取得ソート・バッファのサイズ。取得バッファおよび取得ソート・バッファのサイズの重要性の説明は、993 ページの「バッファのサイズの変更」を参照してください。
- データベースのワークロード。たとえば、計算スクリプトの複雑さや、データ・クエリーの数および複雑さ。
- `essbase.cfg` ファイル内の CALCLOCKBLOCK 設定を SET LOCKBLOCK 設定(使用する CALCLOCKBLOCK 設定を指定)と組み合せて使用して定義されたデータ・ブロックの数。
- 属性次元内のメンバーを含む、アウトライン内の動的計算メンバーの数。
- 分離レベル設定
- 同期点
- トリガーの使用
- MDX への影響

アプリケーションの起動用メモリー要件の見積り

アプリケーションの起動時に使用される全メモリーに加え、各データベースのメモリー要件を計算する必要があります。

起動される各アプリケーションには、起動時に次のメモリー要件があります:

- Essbase コードおよび静的データ(10MB)。この数値は、オペレーティング・システムにより異なる場合があります。
- (**オプション**) JVM (2-4 MB)。これは、カスタム定義関数に使用されます。この値はオペレーティング・システムによって異なります。

サーバーで同時に実行されるアプリケーションの数に起動要件を掛けて、その結果の値を表 252 の ML とラベル付けされたセルに記入します。

データベースの起動用メモリー要件の見積り

Essbase サーバーの各データベースのメモリー要件を計算します。

データベースごとに、表 250 のコピーを作成するか、1 つのデータベース用のワークシートとして別個の用紙を使用します。Essbase サーバーに複数のデータベースがある場合は、データベースごとにこのプロセスを繰り返します。ワークシートにデータベース名を記入します。

各行は、そのコンポーネントのサイズ設定方法を説明する情報にリンクします。それぞれの計算を実行し、その結果を「サイズ(MB)」列の該当のセルに記入します。計算によっては、表 249 に記入した要素を使用します。「サイズ」列にすべてのサイズを記入した後、それらのサイズを合計して、そのデータベースのメモリー要件を決定します。

表 250　ワークシート: データベースのメモリー要件の見積り

データベース名:	**サイズ(MB)**
メモリー要件:	
1 つのデータベースの起動要件:	
データベース・アウトライン 1178 ページの「メモリー内で使用するアウトライン・サイズ」を参照してください。	MA:
インデックス・キャッシュ 917 ページの「キャッシュのサイズ設定」を参照してください。	MB:
キャッシュに関連するオーバーヘッド 1179 ページの「キャッシュに関連するオーバーヘッド」を参照してください。	MC:
データ構造用領域 1181 ページの「データ構造用メモリー領域」を参照してください。	MD:
操作要件:	

データベース名:	サイズ(MB)
データ取得に使用するメモリー 1181 ページの「データ取得用の追加メモリー要件の見積り」を参照してください。	ME:
計算に使用するメモリー 1186 ページの「計算用の追加メモリー要件の見積り」を参照してください。	MF:
データベースに必要な合計メモリー: MA + MB + MC +MD + ME + MF	MG:
MB 単位の合計データベース・メモリー要件(MB): MG / 1,048,576 bytes このデータベースの名前と必要な合計メモリーを表 252 に入力します。このワークシートで、サーバー上のすべてのデータベースのメモリー要件が決定されます(a から g までの 7 つのプレースホルダがデータベース向けに用意されています)。	MH:

サーバー上のデータベースごとにこれを繰り返します。サーバー上のすべてのデータベースで必要なメモリーの見積りが完了した後、1174 ページの「Essbase サーバーの合計ディスク・スペース要件の見積り」に進みます。

メモリー内で使用するアウトライン・サイズ

ディスク・スペースの計算に含まれる属性の関連付け領域は、メモリーのサイズ決定の要素ではありません。アウトラインの主要領域のみを計算します。

メモリー・サイズ要件を求めるには、次の要素を使用してアウトライン・サイズを計算します:

- アウトライン内のメンバー数
- メンバー名および別名の長さ(シングル・バイト換算での文字数)

➤ アウトラインのメモリー要件を計算するには、メンバーの数を 300 バイト-400 バイトの名前の長さで乗算し、その結果を表 250 の MA とラベル付けされたセルに記入します。

データベースに含まれている別名が少ない場合や、非常に短い別名と短いメンバー名が含まれている場合は、300 バイト-400 バイトの範囲内でより小さい数を使用します。名前や別名が非常に長いことがわかっている場合は、この範囲内でより大きい数を使用します。

名前の長さは概算平均であるため、次の式で求められる値は、アウトラインの主要領域のおおまかな見積りにすぎません:

```
   memory size of outline
= number of members
* name-length factor
```

注： メンバー名および別名の最大サイズは、付録 A「制限」を参照してください。

例

次の例では、アウトラインのメンバー数が 26,000、メンバー名の長さは、メジアンであると仮定しています。次の計算によって、メモリー内で使用するアウトライン・サイズを見積ります:

```
   26,000 members
* 350 bytes per member
= 9,100,000 bytes
```

インデックス・キャッシュ

Essbase では、起動時にインデックス・キャッシュ用のメモリーが確保されます。インデックス・キャッシュのサイズはユーザーが指定できます。インデックス・キャッシュのサイズを決定するには、917 ページの「キャッシュのサイズ設定」を参照し、サイズを表 250 の MB とラベル付けされたセルに記入します。

キャッシュに関連するオーバーヘッド

Essbase では、キャッシュを操作している間は追加のメモリーを使用します。

このキャッシュに関連するオーバーヘッドの計算には、次の要素を使用します:

- インデックス・キャッシュ(表 250 の値 MB)
- サーバー・スレッドの数(表 249 の値 MJ)

➤ 起動時にキャッシュ関連のオーバーヘッドを計算するには:

1 **インデックス・キャッシュ・サイズの半分をバイト単位で計算します。**

```
     index cache size
* .5
= index cache-related overhead
```

2 **次の式を使用して、追加キャッシュのオーバーヘッドをバイト単位で計算します:**

```
     ((# of server threads allocated to the Essbase Server process * 3)
* 256)
+ 5242880 bytes
= additional cache overhead
```

3 **インデックス・キャッシュのオーバーヘッドと追加キャッシュのオーバーヘッドを合計します。**

```
     cache-related overhead
```

```
= index cache-related overhead
+ additional cache overhead
```

結果を、表 250 の MC とラベル付けされたセルに記入します。

論理ブロック内のセル数

論理ブロックという用語は、メモリー内の拡張ブロックを指します。

➤ 論理ブロックのセル数を決定するには、各密次元の全メンバー数(動的計算メンバーと動的計算および保管メンバーを含む。ラベルのみメンバーと共有メンバーは除く)を乗算します:

1 **表 251 をワークシートとして使用して、各密次元と、ラベルのみメンバーと共有メンバーを除いたメンバー数を入力します。8 つ以上の密次元がある場合は、他の場所に次元をリストし、すべての密次元を計算に含めます。**

表 251　ワークシート: 論理ブロック内のセル数

密次元の名前	メンバー数
	a.
	b.
	c.
	d.
	e.
	f.
	g.

2 **最初の密次元のメンバー数(a 行)に、2 番目の密次元のメンバー数(b 行)を乗算し、さらに 3 番目の密次元のメンバー数(c 行)を乗算し、というようにして各行の値を乗算し、論理ブロック内のセルの総数を決定します:**

```
a * b * c * d * e * f * g = total number of cells
```

結果を、表 249 の MI とラベル付けされたセルに記入します。

例

ラベルのみメンバーと共有メンバーを除き、Sample.Basic の密次元には、17 個(年)、14 個(メジャー)および 4 個(シナリオ)のメンバーが含まれています。Sample.Basic の論理ブロックのセル数の計算は、次のとおりです:

```
17 * 14 * 4 = 952 cells
```

データ構造用メモリー領域

Essbase では、アプリケーションの起動時に、次の要素に基づいてメモリーの領域が確保されます:

- アウトライン内のメンバー数
- 論理ブロック内のセルの数(表 249 の値 MI)
- サーバー上のスレッドの数(表 249 の値 MJ)

➤ メモリー内のデータ構造領域を計算するには:

1 次の式を使用して、バイト単位でサイズを計算します:

```
     Number of threads
* ((Number of members in the outline * 26 bytes)
+ (Logical block cell count * 36 bytes))
```

2 結果を、表 250 の MD とラベル付けされたセルに記入します。

例

この例では、Sample.Basic データベース用に 20 のスレッドがあると仮定しています。データ構造に必要なメモリー内の起動領域は、次のように計算します:

```
   20 threads
* ((79 members * 26 bytes) + (952 cells * 36 bytes))
= 726,520 bytes 726,520 bytes
/ 1,048,576 bytes = .7 MB
```

データベース操作用の追加メモリー要件の見積り

起動時のメモリー要件に加え、クエリーや計算などの操作に追加のメモリーが必要です。多数の変数があるため、操作のメモリー要件を見積る唯一の方法は、サンプルの操作を実行し、その間に使用されるメモリーを監視することです。次の項でガイドラインを示します。

データ取得用の追加メモリー要件の見積り

Essbase では、様々なソースからのデータベース情報に対する要求(クエリー)が処理されます。たとえば、Essbase では Oracle Essbase Spreadsheet Add-in やレポート・ライターからのクエリーを処理します。Essbase ではこれらのクエリーでデータが取得される際(特に、データ取得のために Essbase で動的計算を実行する必要がある場合)は、追加メモリーを使用します。この項では、クエリー処理に対する Essbase のメモリー要件について説明します。

Essbase はマルチスレッド・アプリケーションであり、クエリーにはスレッドが割り当てられます。スレッドは、Essbase の起動時に自動的に作成されます。通常、スレッドは、Essbase サーバーをシャット・ダウンするまで存在します。758 ページの「マルチスレッド処理」を参照してください。

Essbase でクエリーが処理されるたびに、使用可能なスレッドが循環します。たとえば、起動時に 20 個のスレッドが使用可能であると仮定します。各クエリーが処理されると、Essbase により、それに続く各クエリーが次の順番のスレッドに割り当てられます。20 番目のスレッドが割り当てられた後、Essbase は循環して初めに戻り、最初のスレッドに 21 番目のクエリーを割り当てます。

クエリーの処理中、スレッドには一定のメモリーが割り当てられ、クエリーが完了すると、このメモリーのほとんどが解放されます。オペレーティング・システムに対して解放されるメモリーと、データベースが使用する動的計算キャッシュに対して解放されるメモリーがあります。ただし、以降のクエリーの処理に使用できるように、スレッドではこのメモリーの一部が保持されます。結果として、スレッドでその最初のクエリーが処理された後、そのスレッドによって保持されるメモリーは、Essbase の最初の起動時と比較して大きくなります。

Essbase では、次の両方の条件に当てはまる場合は、クエリー処理に最大のメモリーが使用されます:

- 最大数の同時クエリーが処理される場合。
- Essbase によって、すべてのスレッドが少なくとも 1 つのクエリーに割り当てられている場合。

起動時に少なくとも 20 個のスレッドが使用可能なこの例では、少なくとも 20 個のクエリーが処理されており、かつ最大数の同時クエリーが処理中である場合に、クエリーで使用されるメモリー量は最大になります。

追加メモリーの最大必要量の計算

➤ 実際のクエリーを監視することによって、クエリーのメモリー要件を見積るには:

1. **クエリー中のメモリーの使用状況を監視します。**
2. **同時実行のクエリーに使用されるメモリー量を合計して、クエリー処理に使用される可能性のある最大メモリー量を計算します。次に、クエリーに備えて待機中のスレッドで確保されている追加メモリーを加算します。**

クエリー処理の前後における最大メモリー使用率の見積り の式を計算する際は、次の変数を使用します:

- スレッドの合計数(`Total#Threads`)
- 任意のクエリーの処理中における最大メモリー使用率(`MAXAdditionalMemDuringP`)
- 同時クエリーの許容最大数(`Max#ConcQueries`)
- 任意のクエリーの処理後における最大メモリー使用率(`MAXAdditionalMemAfterP`)

スレッドの合計数の決定

使用可能なスレッドの潜在数は、購入したライセンス・ポートの数に基づきます。使用可能なスレッドの実際の数は、エージェントまたはサーバーで定義する設定に依存します。以降の計算では、システム上のスレッドの数を `Total#Threads` の値として使用します。

同時クエリーの最大数の見積り

同時クエリーの最大数を決定し、この値を以降の計算の Max#ConcQueries に使用します。この値は Total#Threads を上回ることはありません。

クエリー処理の前後における最大メモリー使用率の見積り

個々のクエリーのメモリー使用率は、各クエリーのサイズと、Essbase で各クエリーを処理するためにアクセスする必要のあるデータ・ブロックの数によって決まります。メモリー使用率を見積るには、Essbase で各クエリーの処理中と処理後に使用される追加メモリーを計算します。

最も多くのメモリーを使用すると予想されるクエリーをいくつか選びます。多数のメンバーを処理する必要があるクエリー(たとえば、範囲やランクの処理を実行するクエリー)を考えます。

➤ クエリーのメモリー使用率を見積るには:

1. essbase.cfg 設定の DYNCALCACHEMAXSIZE を 0 (ゼロ)に設定して、動的計算キャッシュをオフにします。動的計算キャッシュをオフにすると、クエリー完了後に、動的計算中にブロックに使用されるメモリーが ESSSVR プロセスによってオペレーティング・システムに確実に解放されることで、スレッドで保持されたままのメモリーの測定が可能になります。
2. Essbase アプリケーションを起動します。
3. オペレーティング・システムのメモリー監視ツールを使用して、クエリーの処理前に Essbase サーバーで使用されるメモリーを調べます。ESSSVR プロセスに関連付けられた値を使用します。

   この値を MemBeforeP として使用します。
4. クエリーを実行します。
5. オペレーティング・システムのメモリー監視ツールを使用して、クエリー処理中の Essbase サーバーのピーク・メモリー使用率を調べます。この値は ESSSVR プロセスに関連付けられます。

   この値を MemDuringP として使用します。
6. オペレーティング・システムのメモリー監視ツールを使用して、クエリー完了後の Essbase のメモリー使用率を調べます。この値は ESSSVR プロセスに関連付けられます。

   この値を MemAfterP として使用します。
7. 次の 2 つの値を計算します:

   - Essbase でクエリーの処理中に使用される追加メモリー (AdditionalMemDuringP):

     ```
     AdditionalMemDuringP = MemDuringP - MemBeforeP
     ```

   - Essbase でクエリー処理の終了後に使用される追加メモリー (AdditionalMemAfterP):

```
AdditionalMemAfterP = MemAfterP - MemBeforeP
```

8 該当するすべてのクエリーに対して前述の計算を完了したら、すべての計算結果を比較して、次の 2 つの値を決定します:

- クエリーで使用される最大の AdditionalMemDuringP: (MAXAdditionalMemDuringP)
- クエリーで使用される最大の AdditionalMemAfterP: (MAXAdditionalMemAfterP)

9 次のステートメントに手順 7 で得た 2 つの値を挿入します。

データ取得のために必要な追加メモリーは、次の量を上回ることはありません:

```
      Max#ConcQueries

*
      MAXAdditionalMemDuringP

+ (
      Total#Threads
       -
      Max#ConcQueries
      )
*
      MAXAdditionalMemAfterP
```

表 250 の ME とラベル付けされたセルに、この計算の結果をバイト単位で記入します。

この計算方法では、すべての同時クエリーが最大サイズのクエリーと仮定しているため、結果が実際の要件を上回る場合もあります。同時に実行される実際のクエリーのタイプを推定するのは困難です。

クエリー中に使用されるメモリーを調整するため、取得バッファおよび取得ソート・バッファの値を設定できます。993 ページの「取得バッファのサイズの設定」および 994 ページの「取得ソート・バッファのサイズの設定」を参照してください。

実際のクエリーを監視しない、追加メモリー要件の見積り

このテストを実際のクエリーで実行できない場合は、クエリーの操作要件のおおまかな見積りを計算できます。要件は取得ごとにかなりの違いがあります。通常、この見積りでは、次の固定要素が使用されます:

- 取得バッファのサイズ(10,240 バイト)
- クエリーでソートを行う場合は、取得ソート・バッファのサイズ(20,480 バイト)
- フォーマットに関する情報を保持するバッファのサイズ(140 KB、切り上げると 144,000 バイト)

- レポート・ライター要素:
  - 取得内の選択したメンバー当たり 40 バイト
  - 最大次元のメンバー当たり 8 バイト

この見積りでは、次の変数も使用します:

- 動的に計算される値に使用されるメモリー。これは次の数に基づきます:
  - SET LOCKBLOCK コマンドで指定されたデータ・ブロック数。
  - 論理ブロック内のセル数。これには、動的計算メンバーが含まれます。表 250 の値 MH を参照してください。
- データ・キャッシュのサイズ。919 ページの「データ・キャッシュ・サイズの設定」を参照してください。
- 直接 I/O が使用される場合は、データ・ファイル・キャッシュのサイズ。918 ページの「データ・ファイル・キャッシュ・サイズの設定」を参照してください。
- レポート・ライターでは、次の数が要素の 1 つとして含まれます:
  - 平均取得で選択されたメンバー。
  - 計算する最大の次元内のメンバー

次の 2 つの計算式を使用すると、取得に必要なメモリーを計算できます:

- 各取得で使用するバッファおよび作業領域:

```
     retrieval buffer (10,240 bytes)
+ retrieval sort buffer (20,480 bytes)
+ formatting buffer (144,000 bytes)
+ each selected member in the retrieval (40 bytes)
+ each member of the largest dimension (8 bytes)
+ dynamic calc area
+ data cache size
+ data file cache size
```

- 見積りした同時取得数に必要なメモリー:

```
     Member storage area for the largest dimension
+ (number of retrievals
* sum of buffer and work areas used in each retrieval)
```

計算を合計し、その結果をバイト単位で、表 250 の ME とラベル付けされたセルに記入します。

例

同時クエリーに必要な最大メモリー量を見積るために、この例の次の値を想定します:

- 最大次元に必要な領域:

```
23,000 members * 8 bytes/member = 184,000 bytes
```

- 各取得に適用されるバッファおよび作業領域の値:
  - 取得バッファ: 10,240 バイト
  - 取得ソート・バッファ: 20,480 バイト
  - フォーマット・バッファ: 144,000 バイト
  - 動的計算領域: 761,600 バイト

    SET LOCKBLOCK で 100 ブロックと設定した場合、計算は次のようになります:

    ```
    100 blocks * 7616-byte block size = 761,600 bytes
    ```

  - データ・キャッシュ・サイズ: 3072 KB (3,145,728 バイトに等しい)
  - データ・ファイル・キャッシュのサイズ: ゼロ。この例では直接 I/O は使用されません。
- 最大の次元には 23,000 のメンバーがあります。
- 同時クエリーの最大数は 20 個です。
- 選択されたメンバーの数は、すべてのクエリーで 10,000 メンバーになるように一般化されます。概算のメモリー要件は、次と等しくなります:

  ```
  10000 members * 40 bytes/member = 400,000 bytes
  ```

取得メモリーの見積り:

```
   184,000 bytes + (20 concurrent inquiries
* (10,240 bytes + 20,480 bytes + 144,000 bytes
+ 761,600 bytes + 3,145,728 bytes + 400,000 bytes))
= 75,824,960 bytes
```

計算用の追加メモリー要件の見積り

既存の計算スクリプトの場合は、サーバーのオペレーティング・システムで提供されるメモリー監視ツールを使用して、メモリー使用率を監視できます。最も複雑な計算を実行し、計算の実行前と実行中のメモリー使用率を確認します。この差異を計算し、その数字を計算スクリプトの追加メモリー要件として使用します。

計算パフォーマンスの包括的な説明は、第 58 章「計算の最適化」を参照してください。

計算スクリプトを使用したテストを実行できない場合は、次の値を合計することによって、計算操作の操作要件の、非常におおまかな見積りを計算できます:

- 計算機キャッシュのサイズ。

  920 ページの「計算機キャッシュのサイズ設定」を参照してください。

- データ・キャッシュのサイズ。

  919 ページの「データ・キャッシュ・サイズの設定」を参照してください。
- アウトラインのサイズ。計算には、データベース・アウトラインのメンバー当たり約 30 バイトの追加メモリーが Essbase で使用されます。同時計算の詳細は、978 ページの「キャッシュの管理によるパフォーマンスの改善」を参照してください。
- ブロックによって使用されるメモリーのサイズは、SET LOCKBLOCK コマンドによって確保されます。
  - メモリー要件をバイト単位で計算するには、指定した数のデータ・ブロックにデータ・ブロックの論理サイズを掛けます。
  - 論理ブロック・サイズを求めるには、セルの数(表 249 の値 MI)に、セル当たり 8 バイトを掛けます。

合計計算要件の場合は、同時に実行するすべての計算に必要なメモリーを合計し、その合計を表 250 の MF とラベル付けされたセルに記入します。

注： 計算スクリプトのサイズと複雑さは、必要なメモリーに影響します。この影響を見積るのは困難です。

Essbase 合計メモリー要件の見積り

サーバー上で必要な合計メモリーを見積るために、表 252 をワークシートとして使用できます。

表 252　ワークシート: 合計サーバー・メモリー要件

コンポーネント	必要なメモリー(MB)
アプリケーションの起動用メモリー要件の合計 1177 ページの「アプリケーションの起動用メモリー要件の見積り」を参照してください。	ML:
行 a から g に、同時データベース(表 250 のコピーから)をリストし、右の列に各メモリー要件(MH)を記入します。	
a.	MH:
b.	MH:
c.	MH:
d.	MH:
e.	MH:
f.	MH:
g.	MH:
オペレーティング・システムのメモリー要件	MN:

コンポーネント	必要なメモリー(MB)
サーバー用の合計メモリー要件の見積り	M0:

➤ サーバーの Essbase 合計メモリー要件を見積るには:

1 1177 ページの「アプリケーションの起動用メモリー要件の見積り」で説明しているように、ML とラベル付けされたセルに、アプリケーションの起動用合計メモリー要件を記録します。

2 サーバー上で同時に実行されるデータベースの最大のセットをリストします。各データベースの MH 値に対して、「必要なメモリー」列に、表 250 のデータベース要件ワークシートで見積ったメモリー要件を記入します。

3 オペレーティング・システムのメモリー要件を決定し、表 252 の MN とラベル付けされたセルに、MB 単位で値を記入します。

4 すべての値を合計し、計算結果を M0 とラベル付けされたセルに記入します。

5 MN の値を、サーバー上の合計使用可能ランダム・アクセス・メモリー(RAM)と比較します。

注: さらに、Essbase サーバーのコンピュータにインストールされている可能性のある、Administration Services などのクライアント・ソフトウェアのメモリー要件も考慮してください。Oracle Enterprise Performance Management System Installation and Configuration Guide を参照してください。

キャッシュ・メモリーのロックが使用可能な場合は、合計メモリー要件が使用可能な RAM の 3 分の 2 を上回らないようにしてください。3 分の 2 を上回ると、システム・パフォーマンスが著しく低下する可能性があります。キャッシュ・メモリーのロックが使用不可の場合は、合計メモリー要件が使用可能な RAM を上回らないようにしてください。

使用可能なメモリーが不十分な場合は、キャッシュ設定を再定義し、メモリー要件を再計算できます。このプロセスは繰り返し実行できます。929 ページの「キャッシュ設定の微調整」を参照してください。場合によっては、追加の RAM の購入が必要になることもあります。

E

ESSCMDの使用

この付録の内容

関連項目:

- 『Oracle Essbase テクニカル・リファレンス』(ESSCMD の構文と使用方法)
- Oracle Essbase Administration Services Online Help(MaxL スクリプト・エディタの詳細)

ESSCMD について

ESSCMD を使用することにより、Essbase サーバーでは、対話モードまたはバッチ・モードで、コマンドラインからサーバーの操作を実行できます。

対話モードとは、コマンドを ESSCMD コマンドラインで入力し、必要な場合にプロンプトが表示されることを意味します。対話モードは、コマンドを入力する必要がほとんどない簡単な操作や、情報の即時チェック、エラー・チェックを行いたい場合に便利です。

バッチ処理モードは、定期的な Essbase サーバーのメンテナンス・タスクや診断タスクを自動化するために使用されます。スクリプト・ファイルやバッチ・ファイルを記述し、それをコマンドラインから実行できます。特定の一連のファイルを頻繁に使用する場合や、タスクに多数のコマンドが必要な場合は、バッチ処理モードが便利です。

構文のガイドラインについて

通常、ESSCMD コマンドを入力する際は、他の計算コマンドの場合と同じ構文を使用します。ただし、ESSCMD の対話型モードとバッチ処理モードの間には、引用符とセミコロンのステートメントのターミネータに関する要件で違いがあります。スクリプト・ファイルまたはバッチ・ファイルを作成する際には、この項のガイドラインに従ってください。

引用符

引用符(" ")は、文字列のパラメータおよびコマンド応答を囲むために使用されます。

- 対話型の ESSCMD では、引用符の使用はオプションです。引用符は、パラメータに埋込みのスペースがあるときに使用します。たとえば、次のとおりです:

```
CALC "Calc All;"
```

- ESSCMD スクリプト・ファイルの場合は、次のように、文字列のパラメータおよびコマンド応答はすべて引用符で囲んでください:

```
LOGIN "Localhost" "user1" "Password";
```

- 数値のパラメータおよびコマンド応答は、引用符で囲む必要はありません。
- 引用符を引用符で囲むことはできません。

セミコロン・ステートメント終端文字

; (セミコロン)ステートメント終端文字は、コマンドの終端を示します。たとえば次のように使用します:

```
SELECT "SAMPLE" "BASIC";
```

- 対話型の ESSCMD では、[Enter]キーを押すことで、コマンドが完了したことが ESSCMD に通知されます。ステートメント終端文字はオプションです。
- ESSCMD スクリプト・ファイルでは、終端文字の使用はオプションですが、コマンドに多数のパラメータを指定する場合は終端文字を使用してください。終端文字の使用が特に重要であるのは、一部のパラメータがオプションの場合に、パラメータ・リストの終端を通知する場合です。

オプションのパラメータの一部を省略し、リストの終端でセミコロンを使用しなかった場合、ESSCMD はそのファイル内の次のコマンドの中で残りの値を検索します。これによって、予測不可能な結果が生じる可能性があります。

たとえば、『Oracle Essbase テクニカル・リファレンス』で定義されている SETAPPSTATE コマンドおよび SETDBSTATE コマンドでは、プロセスにおける混乱を防止するために終端でセミコロンを使用する必要があります。

注： この章で示す構文例では、引用符とセミコロン終端文字を必ず使用しています。

ESSCMD 操作の取消し

ESSCMD の実行時に、ESSCMD が応答するまで[Esc]キーを押し続けることで、計算、エクスポートまたは再構築操作などの非同期操作を取り消せます。

ファイルの参照

一部のコマンドでは、アーチファクトまたはファイル名の前に 1 から 4 の数値パラメータを指定する必要があります。この数値パラメータで、Essbase に対してアーチファクトまたはファイルの検索場所を指定します。パラメータを指定すると、ESSCMD では、他のアプリケーション、データベースまたはシステムでファイルが検索されます。

表 253 は、数値パラメータのそれぞれの値(「数字」)、数値パラメータが適用されるファイルの場所、および各パラメータの設定時に、ESSCMD から要求される情報のリストです。なお、appName はアプリケーション名、dbName はデータベース名です。

表 253　数値パラメータのリスト

数字	ファイル	Essbase プロンプトによりユーザーに求められる情報:
1	ローカルまたはクライアント・ベースのファイル	• Windows: ARBORPATH/client/appname/dbname ディレクトリ内のファイル • UNIX: ARBORPATH/client/appname/dbname ディレクトリ内のファイル
2	リモートまたはサーバー・ベースのファイル	• Windows: ARBORPATH/app/appname/dbname ディレクトリ内のファイル • UNIX: ARBORPATH/app/appname/dbname ディレクトリ内のファイル
3	ファイル	そのファイルに対する完全修飾パス(ファイルが現在の ESSCMD ディレクトリに存在している場合は除く)
4	SQL テーブル	その SQL テーブルに対する完全なネットワーク情報およびデータベース情報

たとえば、LOADDATA コマンドでは、クライアントまたは Essbase サーバー上にあるデータ・ファイルをロードできます。このコマンドでは、データ・ファイルを検索する場所を Essbase に指示するために、数値パラメータが必要です。次の例では、ESSCMD で、ロード対象ファイルの完全修飾パス名を求めるプロンプトが表示されます:

```
LOADDATA 3
```

通常、ファイル拡張子の指定は、対話モードでもバッチ処理モードでもオプションです。ただし、使用するコマンドにおいて、ファイルの場所を示す数値パラメータが必要な場合は、ファイル拡張子を指定します:

- ファイル・オプション 3 (ファイル)を使用する場合は、対話モードでもバッチ処理モードでもファイル拡張子を入力する必要があります。
- ESSCMD を起動したディレクトリ内にアーティファクトがある場合は、パスを入力する必要はありません。

複数のデータベースへのアクセス

ESSCMD では Essbase サーバー上の複数のログイン・インスタンスをサポートしているため、単一セッション内の複数のデータベースにアクセスできます。複数のデータベースにログオンする場合でも、サーバー・ライセンスで使用するポートは 1 つのみです。

大文字と小文字の区別に関するガイドライン

Essbase サーバーでは、アプリケーションとデータベースの名前は、ユーザーが指定したとおりに作成されます。どのプラットフォームでも、大文字と小文字が変更されることはありません。

Essbase サーバーでは、後方互換性のために、最初に大文字と小文字を正確に使用して、既存のアプリケーションとデータベースの名前が検索されます。ただし、ファイルが見つからない場合は、大文字と小文字の可能なすべての組合せで、既存のアプリケーションとデータベースの名前が検索されます。

Essbase では、大文字と小文字のみが異なるアプリケーション名とデータベース名は作成できません。たとえば、アプリケーション mYdATA が存在しているときにアプリケーション MyData を作成しようとすると、Essbase でエラー・メッセージが表示されます。

メンバー名の大文字と小文字を区別するかは、選択できます。

➤ メンバー名の大文字と小文字を区別する方法は、『Oracle Essbase Administration Services オンライン・ヘルプ』の「メンバー名の大文字と小文字を区別する方法」を参照してください。

ヘルプの表示

個々の ESSCMD コマンドの詳細と構文は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

ESSCMD の開始準備

ESSCMD を開始する前に、次のアイテムがインストールおよび実行されていることを確認します:

- Essbase
- Essbase 上の通信プロトコル(TCP/IP)

ESSCMD の開始および終了

ESSCMD は、Essbase サーバーおよび Essbase 管理サーバー上の `esscmd.exe` および `esscmd.hlp`(UNIX プラットフォームの場合は `esscmd` および `esscmd.hlp`)という 2 つのファイルで構成されます。

➤ ESSCMD を開始するには、オペレーティング・システムのコマンド・プロンプトから、ESSCMD を入力します。

ESSCMD を開始すると、次のようなコマンド・プロンプトが表示されます:

```
:::[
n
```

```
]->
```

n は、アクティブなログイン・インスタンスの値です。その後のログインが成功するたびに、この値は 1 つずつ増加します。ESSCMD 開始時のインスタンス番号はゼロ(0)です。

注： アクティブなログイン・インスタンスを切り替えるには、SETLOGIN コマンドを使用します。アクティブなログイン・インスタンスを表示するには、LISTLOGINS コマンドを使用します。

➤ ESSCMD を終了するには、プロンプトで EXIT と入力し、[Enter]キーを押します。

ESSCMD がアプリケーション・サーバーから切断され、セッションが終了します。

対話モードの使用

対話モードでは、コマンドを入力してプロンプトに応答します。対話モードは、少ないコマンドで済む簡単なタスクを実行するときに便利です。多数のコマンドを必要とする複雑なタスクを実行する場合は、スクリプト・ファイルやバッチ・ファイルの作成を検討してください。1195 ページの「バッチ・プロセスにおけるスクリプト・ファイルおよびバッチ・ファイルの使用」を参照してください。

対話モードで実行中の構文の表記規則については、1189 ページの「構文のガイドラインについて」を参照してください。

Essbase サーバーへのログオン

ESSCMD を開始した後、Essbase サーバーに接続する必要があります。これにより、コマンドを入力できるようになります。

➤ Essbase サーバーにログインするには:

1. **ESSCMD プロンプトで、LOGIN コマンドを使用してサーバーにログオンします。**

   『Oracle Essbase テクニカル・リファレンス』を参照してください。

2. **サーバー名を入力します。**

   Essbase サーバー・ウィンドウから接続する場合、サーバー名はネットワークの設定によって異なります。たとえば、ここでは名前は aspen であるとします。

3. **ユーザー名を入力します。**
4. **パスワードを入力します。**

次のような ESSCMD プロンプトが表示されます:

```
aspen:::
userName
```

```
[1]->
```

userName は自分のログイン名です。

任意の有効な ESSCMD コマンドを入力できます。コマンドの完全なリストは、『Oracle Essbase テクニカル・リファレンス』を参照してください。

注： アプリケーションをメモリーにロードし、Essbase サーバー上のデータベースを選択するには、SELECT コマンドを使用して、サーバー上にあるアプリケーションからデータベースを選択します。

次のような ESSCMD プロンプトが表示されます:

```
aspen:
appname
:
dbname
:
userName
[1]->
```

appname はアプリケーションの名前です。dbname は接続しているデータベースの名前です。

コマンドの入力

対話モードでコマンドを入力する場合は、次のいずれかの方法を選択します:

- コマンドを入力して[Enter]を押します。

  ESSCMD により各コマンド・パラメータの入力を求めるプロンプトが表示されます。たとえば、SELECT コマンドには、次のコマンド構文に示すように、2 つのパラメータがあります:

  ```
  SELECT "
  appname
   " "
  dbname
   ";
  ```

  SELECT のみを入力し、[Enter]キーを押すと、ESSCMD によって最初のパラメータであるアプリケーション名(appName)が要求されます。アプリケーション名を入力し、[Enter]キーを押すと、ESSCMD によってデータベース名(dbName)を入力するように要求されます。

- コマンドとすべてのパラメータを入力してから[Enter]を押します。

  たとえば SELECT コマンドの場合では、次のように入力します:

```
SELECT "Sample" "Basic";
```

どの方法を使用しても、対話型のプロンプトにアプリケーションとデータベースの名前が反映されます。たとえば、次のプロンプトは、Sample アプリケーションおよび Basic データベースが選択されていることを示します:

```
aspen:Sample:Basic:User[1]->
```

この場合、通常は要求されるアプリケーションまたはデータベース名パラメータを指定せずに、他のコマンドを入力できます。

操作の取消し

ESSCMD の実行中に、ESSCMD が応答するまで[Esc]キーを押し続けることで、計算、エクスポートまたは再構築操作などの非同期操作を取り消せます。

注意 Essbase でコマンドが処理されている間は、システムを一時停止または中断しないでください。システムを一時停止すると、Essbase でコマンドが正しく完了しない可能性があります。

バッチ・プロセスにおけるスクリプト・ファイルおよびバッチ・ファイルの使用

一連のコマンドを頻繁に使用する場合、またはタスクを完了するために多数のコマンドを入力する必要がある場合には、スクリプト・ファイルまたはバッチ・ファイルを使用してタスクを自動化することを検討してください。これらのファイルは、バッチ・データのロードや複雑な計算の場合に便利です。

1189 ページの「構文のガイドラインについて」を参照してください。

- スクリプト・ファイルには、ESSCMD コマンドが含まれます。スクリプト・ファイルは、オペレーティング・システムのコマンド・ラインから、またはオペレーティング・システムのバッチ・ファイルから実行できます。スクリプト・ファイルは ESSCMD によって処理されます。デフォルトでは、ESSCMD スクリプト・ファイルの拡張子は.scr です。別の拡張子も使用できます。
- バッチ・ファイルとは、複数の ESSCMD スクリプトを呼び出すオペレーティング・システム・ファイルのことで、オペレーティング・システムのコマンドを組み込むこともできます。バッチ・ファイルを使用して、複数の ESSCMD セッションを実行できます。Essbase サーバー上のバッチ・ファイルは、オペレーティング・システムのプロンプトから実行できます。このファイルはオペレーティング・システムによって処理されます。Windows の場合、バッチ・ファイルの拡張子は*.bat です。

注： UNIX の場合、バッチ・ファイルまたはスクリプト・ファイルはシェル・スクリプトとして記述されます。通常、シェル・スクリプトの拡張子は、.sh (Bourne シェルまたは Korn シェル)または.csh (C シェル)です。

スクリプト・ファイルまたはバッチ・ファイルを実行すると、ESSCMD によってファイル内のコマンドがファイルの最後まで順に実行されます。

一部のコマンドは、Essbase の新しいリリースで変更されています。この変更は既存のスクリプトに影響する可能性があります。スクリプトが現在のリリースで正しく動作することを確認するには、『Oracle Essbase 新機能』および『Essbase Readme』で変更または削除されたコマンドの情報を確認し、必要な場合はスクリプトを変更してください。

スクリプト・ファイルの作成

ESSCMD スクリプト・ファイルでは、一連の使用頻度の高いコマンドまたは長いコマンドを自動化します。各スクリプト・ファイルは、ログイン、アプリケーションとデータベースの選択、ログアウト、終了の各コマンドを備えた完全な ESSCMD セッションである必要があります。

➤ スクリプト・ファイルを定義するには:

1 **ASCII テキスト形式で保存する機能を持つ任意のエディタで、ESSCMD コマンドを入力します。**

2 **ESSCMD スクリプト・ファイル拡張子.scr で、ファイルを保存します。**

たとえば、次のスクリプト・ファイル、test.scr はメモ帳で作成したものです:

```
     LOGIN "localhost" "User1" "password";
SELECT "Sample" "Basic";
GETDBSTATE
EXIT;
```

このスクリプトをオペレーティング・システムのコマンドラインから実行すると、Essbase の localhost サーバーへの User1 のログイン、Sample アプリケーションと Basic データベースの選択、データベース統計の取得および ESSCMD セッションの終了操作が実行されます。

スクリプト・ファイルの実行

➤ ESSCMD でスクリプト・ファイルを実行するには:

1 **オペレーティング・システムのプロンプトで次のコマンドを入力します:**

```
ESSCMD
scriptFileName
```

```
.scr
```

2 `scriptFileName` をスクリプト・ファイルの名前と置き換えます。

たとえば、スクリプト・ファイルが現在のディレクトリ内にある場合は、次のように入力します:

```
ESSCMD TEST.SCR
```

3 スクリプト・ファイルが現在のディレクトリにない場合は、パスも指定します。

例:

```
ESSCMD C:\WORK\SCRIPTS\TEST.SCR (an absolute path on Windows)
```

または

```
ESSCMD..\SCRIPTS\TEST.SCR (a relative path on Windows)
```

スクリプト・ファイル内でのコマンド・エラーの処理

ESSCMD のエラー処理機能には、スクリプト・ファイルのエラーの検査と処理があります。スクリプト・ファイルにエラー処理コマンドを記述するとエラーを検査したり、必要に応じて、適切なエラー処理応答に分岐させたりすることができます。

各 ESSCMD コマンドが実行されると、内部バッファに数値が保管されます。コマンドが正常に実行されると、内部バッファに 0 が戻されます。コマンドが失敗すると、バッファにエラー番号が保管されます。これは非ゼロ・ステータスと呼ばれます。

ESSCMD スクリプト・ファイル内のエラー・チェックでは、次のエラー処理コマンドを使用できます:

- IFERROR では、以前に実行されたコマンドが非ゼロの戻りステータス(実行に失敗)かどうかが検査されます。ステータスがゼロでない場合、後続のすべてのコマンドがスキップされ、ファイル内のユーザー指定のポイントにジャンプします。処理はこのポイントで再開されます。スクリプト・ファイルでは、エラー処理ルーチンまたはファイルの最後に分岐できます。
- RESETSTATUS を使用すると、保存されたすべてのステータス値が 0(ゼロ)に戻ります。これは、後続の処理の中でステータス検査を実行できるようにするためです。
- GOTO を使用すると、エラーが発生したかどうかにかかわらず、ファイル中のユーザー定義ポイントへ強制的に無条件分岐させることができます。

次の `load.scr` というサンプル・ファイルでは、LOADDATA コマンドの実行が成功しなかった場合、ファイルの終端に分岐されます。これにより、空のデータベースに対する計算およびレポート・スクリプト処理を回避できます:

```
   LOGIN "localhost" "User1" "password" "Sample" "Basic";
LOADDATA 2 "calcdat";
IFERROR "Error";
CALC "Calc All;";
IFERROR "Error";
RUNREPT 2 "Myreport";
IFERROR "Error";
[possible other commands]
EXIT;

:Error
```

注： OUTPUT コマンドを使用すると、エラーをテキスト・ファイルに記録できます。

ESSCMD エラー・コマンドの構文と使用方法は、『Oracle Essbase テクニカル・リファレンス』を参照してください。

サンプル・スクリプト・ファイルの確認

Essbase には、Sample.Basic データベースに基づいた、一般的なバッチ操作を示すサンプルのスクリプト・ファイルが付属しています。サンプルのスクリプト・ファイルは次のディレクトリにあります:

```
   ARBORPATH
   /app/Sample/Basic
```

サンプル・スクリプト: データのインポートと計算

`sample1.scr` ファイルでは、次のアクションを実行します:

- Essbase サーバーにログオンする
- アプリケーションおよびデータベースを選択する
- 他のユーザーによるログオンとデータベースの変更を防止する
- テキスト・ファイルからデータをインポートする
- データベースを計算する
- ログインを再度可能にする
- ESSCMD を終了する

```
   LOGIN "Poplar" "TomT" "Password";
SELECT "Sample" "Basic";
DISABLELOGIN;
IMPORT 2 "ACTUALS" 4 "Y" 2 "ACTUAL" "N";
CALCDEFAULT;
ENABLELOGIN;
```

```
EXIT;
```

サンプル・スクリプト: 次元構築、SQL ソースからのデータのインポートおよび計算

sample2.scr ファイルでは、次のアクションを実行します:

- Essbase サーバーにログオンする
- アプリケーションおよびデータベースを選択する
- 他のユーザーによるログオンとデータベースの変更を防止する
- SQL データ・ソースからアウトラインを更新する
- SQL からデータをインポートする
- データベースを計算する
- ログインを再度可能にする
- ESSCMD を終了する

```
   LOGIN "Poplar" "TomT" "Password";
SELECT "Sample" "Basic";
DISABLELOGIN;
BUILDDIM 2 "PRODRUL" 4 "PRODTBL" 4 "PROD.ERR";
IMPORT 4 "TOMT" "PASSWORD" 2 "ACTUAL" "N";
CALCDEFAULT;
EXIT;
```

サンプル・スクリプト: レポート印刷のスケジュール

sample3.scr ファイルでは、次のアクションを実行します:

- Essbase サーバーにログオンする
- アプリケーションおよびデータベースを選択する
- 後続の印刷処理のため、出力するレポートをファイルに割り当てる
- ESSCMD を終了する

```
   LOGIN "Poplar" "TomT" "Password";
SELECT "Sample" "Basic";
RUNREPT 2 "REP1" "REP1.OUT";
RUNREPT 2 "REP2" "REP2.OUT";
RUNREPT 2 "REP3" "REP3.OUT";
EXIT;
```

バッチ・ファイルの作成

1 つ以上の ESSCMD スクリプトを実行し、オペレーティング・システムのコマンドを含むバッチ・ファイルを記述できます。バッチ・ファイルの記述に関する構

文について学習するには、使用するオペレーティング・システムの説明書を参照してください。

バッチ・ファイルにおけるコマンド・エラーの処理

オペレーティング・システムのバッチ・ファイルでは、ESSCMD コマンドの戻り値を使用して、そのバッチ・ファイルによって実行されるスクリプトのフローを制御できます。

ESSCMD プログラムでは、終了時に整数値が戻されます。この値は、最後に実行されたコマンドのステータス(通常はそのコマンドが成功したか失敗したか)を表します。この値をテストし、テストに失敗した場合はエラー処理応答に分岐するように、バッチ・ファイルを設定できます。このプロセスは、スクリプト・ファイルの作成と同様です。1197 ページの「スクリプト・ファイル内でのコマンド・エラーの処理」を参照してください。

たとえば、バッチ・ファイルに、データをロードする ESSCMD バッチ・ファイル、データに対して計算を実行する計算スクリプト、および計算結果に関してレポートするレポート・スクリプトの 3 つのスクリプトが含まれているとします。ロードのバッチ・ファイルが失敗すると、計算とレポートも失敗します。この場合は、ファイルのロードが失敗した後でバッチ・ファイルを停止して、失敗の原因となったエラーを修正してから続行するのが最善の方法です。バッチ・ファイルでロード・プロセスの戻り値がテストされ、この戻り値が失敗を示す場合には、バッチ・ファイルは、ロードされなかったデータの計算を試みるのではなく、ファイルの末尾にジャンプして停止するか、または他のエラー処理プロシージャを実行できます。

次の例は、Windows オペレーティング・システムのバッチ・ファイルと、そのファイルで実行される ESSCMD スクリプトの 1 つ(`load.scr`)のコンテンツを示しています。エラーチェック要件は多様なため、この例の構文は実際のオペレーティング・システムの構文とは一致しない可能性があります。バッチ・ファイルのエラー・チェックについては、ご使用のオペレーティング・システムのドキュメンテーションを参照してください。

オペレーティング・システムのバッチ・ファイルには次のようなコマンドが含まれている可能性があります:

```
   ESSCMD LOAD.SCR
If not %errorlevel%==0 goto Error
ESSCMD CALC.SCR
If not %errorlevel%==0 goto Error
ESSCMD REPORT.SCR
If not %errorlevel%==0 goto Error
Echo All operations completed successfully
EXIT

:Error
Echo There was a problem running the script
```

F

Essbaseアプリケーション、データベース、次元、メンバーおよび別名の命名規則

この付録の内容

アプリケーションとデータベースの命名規則

アプリケーションとデータベースの命名を行うときは、次のルールに従います:

- 非 Unicode モードのアプリケーションとデータベースの命名では 8 バイト以下の文字列を使用します。
- Unicode モードのアプリケーションとデータベースの命名では 30 文字以下の文字列を使用します。
- 名前ではスペースを使用しないでください。
- 名前の中で表 254 にリストされている文字を使用しないでください:

表 254　アプリケーション名とデータベース名で制限される文字のリスト

文字	説明
*	アスタリスク
[]	大カッコ
:	コロン
;	セミコロン
,	カンマ
=	等号
>	大なり記号
<	小なり記号
.	ピリオド

文字	説明
+	プラス符号
?	疑問符
"	二重引用符
'	一重引用符
/	スラッシュ
\	円記号(バックスラッシュ)
|	垂直棒
	タブ

- データベース名には、次のものは使用しないでください:
  - 文字列 drxxxxxx (大文字と小文字を区別しません)
  - 予約語 Replay
- 集約ストレージ・データベースの場合は、アプリケーション名またはデータベース名に次の語を使用しないでください:

```
      DEFAULT
LOG
METADATA
REPLAY
TEMP
```

アプリケーション名とデータベース名では、大文字と小文字は区別されません。ただし、大文字と小文字を区別するファイル・システムでは、アプリケーション名やデータベース名は、入力したとおりに作成されます。したがって、大文字と小文字を区別するファイル・システムでアプリケーションとデータベースを作成、名前変更またはコピーしたとき、Essbase では、同じ名前で大文字と小文字の使用が異なるアプリケーション名やデータベース名が使用できないようになっています。たとえば、すべて大文字のアプリケーション名(NEWAPP)を作成した場合、同じ名前を持つアプリケーションは作成できませんが、名前に大文字と小文字が混在するアプリケーション(Newapp)は作成できます。また、アプリケーションおよびデータベース・ファイルをあるコンピュータから別のコンピュータに手動でコピーするときは、そのアプリケーションとデータベースのディレクトリ名の大文字と小文字を、両方のコンピュータで同じように使用する必要があります。

次元、メンバーおよび別名の命名規則

データベース・アウトライン内の次元、メンバー、別名の命名では、次の規則に従います:

- 非 Unicode モードの次元、メンバーまたは別名に名前を付ける場合は、80 バイトまでの名前を使用する。

- Unicode モードの次元、メンバーまたは別名に名前を付ける場合は、80 文字までの名前を使用する。
- 名前では、大文字と小文字の区別を有効にしていない場合には、大文字と小文字は区別されません。

  Oracle Essbase Administration Services Online Help の「アウトライン・プロパティの設定」を参照してください。
- 大文字と小文字の区別が有効な場合でも、重複メンバー名が有効な集約ストレージ・アウトラインで、大文字と小文字のみに違いのある同じ名前を次元に使用しないでください。たとえば、2 つの次元に「Product」と「product」という名前を付けることはできません。
- 次元、メンバーまたは別名の中に、二重引用符(" ")、大カッコ([ ])またはタブを使用しないでください。
- 次元名またはメンバー名の先頭に、表 255 にリストする文字を使用しないでください:

表 255　次元、メンバーおよび別名で制限される文字

文字	説明
@	アット・マーク
\	円記号(バックスラッシュ)
{ }	中カッコ
,	カンマ
-	ダッシュ、ハイフンまたはマイナス
=	等号
<	小なり記号
()	丸カッコ
.	ピリオド
+	プラス符号
'	一重引用符
_	アンダースコア
|	垂直棒

- 名前の先頭または末尾にスペースを入れないでください。Essbase ではこのようなスペースは無視されます。
- 次の語は、次元名、メンバー名または別名には使用しないでください:
  - 計算スクリプトのコマンド、演算子およびキーワード
  - レポート・ライター・コマンド
  - 関数名と関数引数。

- 他の次元の名前、他のメンバーの名前(メンバーが共有でない場合)、世代名、レベル名およびデータベースでの別名
- 次の語:

```
        ALL
AND
ASSIGN
AVERAGE
CALC
CALCMBR
COPYFORWARD
CROSSDIM
CURMBRNAME
DIM
DIMNAME
DIV
DYNAMIC
EMPTYPARM
EQ
EQOP
EXCEPT
EXP
EXPERROR
FLOAT
FUNCTION
GE
GEN
GENRANGE
GROUP
GT
ID
IDERROR
INTEGER
LE
LEVELRANGE
LOOPBLOCK
LOOPPARMS
LT
MBR
MBRNAME
MBRONLY
MINUS
MISSING
MUL
MULOP
NE
NON
NONINPUT
NOT
OR
PAREN
PARENPARM
PERCENT
PLUS
RELOP
```

```
SET
SKIPBOTH
SKIPMISSING
SKIPNONE
SKIPZERO
TO
TOLOCALRATE
TRAILMISSING
TRAILSUM
UMINUS
UPPER
VARORXMBR
XMBRONLY
$$$UNIVERSE$$$
#MISSING
#MI
```

動的時系列メンバーの命名規則

動的時系列メンバーを使用可能にしている場合は、次の関連する世代名は使用しないでください:

- 履歴
- 年
- 季節
- 期間
- 四半期
- 月
- 週
- 日

属性計算次元のメンバー名の命名規則

属性次元(および属性計算次元)が含まれる一意のメンバー・アウトラインでは、属性計算次元のデフォルト名を変更する場合を除き、次の名前を使用しないでください:

- 合計
- カウント
- 最小
- 最大
- 平均

アウトラインが重複メンバー・アウトラインとしてタグ付けされている場合、デフォルト名を使用して他の基本メンバーまたは属性メンバーに名前を付けます。

176 ページの「属性計算次元のメンバー名の変更」を参照してください。

計算スクリプト、レポート・スクリプト、式、フィルタ、代替変数値および環境変数値での命名規則

代替変数値、環境変数値、計算スクリプト、レポート・スクリプト、フィルタ定義、パーティション定義または式では、次のような場合に、MDX ステートメントで使用する場合はメンバー名を大カッコ([ ])で囲み、それ以外で使用する場合は引用符(" ")で囲む必要があります:

- 名前が 1 つ以上の数字で始まる場合(たとえば、100)。
- 名前に、表 256 にリストするスペースまたは文字が含まれる場合:

表 256　メンバー名を囲む必要のある文字

文字	説明
&	アンパサンド
*	アスタリスク
@	アット・マーク
\	円記号(バックスラッシュ)
{ }	中カッコ
:	コロン
,	カンマ
-	ダッシュ、ハイフンまたはマイナス
!	感嘆符
=	等号
>	大なり記号
<	小なり記号
()	丸カッコ
%	パーセント記号
.	ピリオド
+	プラス符号
;	セミコロン
/	スラッシュ
~	チルド

計算スクリプトと式では、Essbase キーワードでもあるメンバー名を、ブロック・ストレージ・データベースでは引用符(" ")、集約ストレージ・データベースでは大カッコ([ ])で囲む必要があります:

```
   BEGIN
DOUBLE
ELSE
END
FUNCTION
GLOBAL
IF
MACRO
MEMBER
RANGE
RETURN
STRING
THEN
```

Essbase システム定義の次元名およびメンバー名のリスト

属性次元を使用する際、Essbase では次の次元名およびメンバー名が作成されます:

属性次元のメンバー名:

- TRUE
- FALSE

次元名: 属性計算

属性計算次元のメンバー名:

- 合計
- カウント
- 最小
- 最大
- 平均

1205 ページの「属性計算次元のメンバー名の命名規則」を参照してください。

MaxL DDL 予約語のリスト

『Oracle Essbase テクニカル・リファレンス』の、予約語リストに関する項を参照してください。

G 異なるセキュリティ・モデルにおけるセキュリティ関連操作の相違

Essbase には、複数のセキュリティ・モデルがあります:

- Essbase ネイティブ・セキュリティ・モード: ユーザー、グループ、役割、アプリケーションなど、すべてのセキュリティの局面が Essbase によって制御されます。すべてのセキュリティ情報は、ARBORPATH の下の Essbase セキュリティ・ファイル(essbase.sec)に格納されます。
- EPM System セキュリティ・モード: セキュリティ管理の大部分が Essbase から Shared Services に移ります。Shared Services ネイティブ・ディレクトリに格納されているセキュリティ情報には、次のものが含まれます: ユーザーおよびグループの情報、ユーザーおよびグループへの役割の割当て、アプリケーション情報など。外部プロバイダ(LDAP、MSAD および OID など)のユーザーに対するユーザー認証は、各プロバイダを介して行われます。essbase.sec ファイルのセキュリティ情報は最小限に維持されます。
- Fusion セキュリティ・モード: Fusion JPS セキュリティでは、ユーザーとグループの役割管理に加え、ユーザーとグループの管理も行われます。外部プロバイダ(LDAP、MSAD および OID など)のユーザーに対するユーザー認証は、各プロバイダを介して行われます。essbase.sec にセキュリティ情報は格納されません。アプリケーションおよびデータベースの情報は essbase.sec に格納されます。

表 257 では、セキュリティ関連操作が各セキュリティ・モードでどのように処理されるかを説明しています。

表 257　異なるセキュリティ・モデルにおけるセキュリティ関連操作の相違

要求	Essbase ネイティブ・セキュリティ	Oracle Enterprise Performance Management System セキュリティ	Fusion セキュリティ
ユーザーの作成	essbase.sec ファイルにユーザーを作成	Shared Services ネイティブ・ディレクトリに Shared Services ネイティブ・ユーザーを作成	無効(エラー・メッセージが戻されます)
グループの作成	essbase.sec ファイルにグループを作成	Shared Services ネイティブ・ディレクトリに Shared Services ネイティブ・グループを作成	無効(エラー・メッセージが戻されます)
ユーザーの削除	essbase.sec ファイルでユーザーを削除	Shared Services ネイティブ・ディレクトリで Shared Services ネイティブ・ユーザーを削除	無効(エラー・メッセージが戻されます)
グループの削除	essbase.sec ファイルでグループを削除	Shared Services ネイティブ・ディレクトリで Shared Services ネイティブ・グループを削除	無効(エラー・メッセージが戻されます)

要求	Essbase ネイティブ・セキュリティ	Oracle Enterprise Performance Management System セキュリティ	Fusion セキュリティ
ユーザー名の変更	essbase.sec ファイルでユーザー名を変更	Shared Services ネイティブ・ディレクトリで Shared Services ネイティブ・ユーザー名を変更	無効(エラー・メッセージが戻されます)
グループ名の変更	essbase.sec ファイルでグループ名を変更	Shared Services ネイティブ・ディレクトリで Shared Services ネイティブ・グループの名前を変更	無効(エラー・メッセージが戻されます)
グループへのユーザーの追加	essbase.sec ファイルでグループにユーザーを追加	Shared Services ネイティブ・ディレクトリでユーザーをグループに追加	無効(エラー・メッセージが戻されます)
グループへのグループの追加	グループ内のグループはサポートされていません	Shared Services ネイティブ・ディレクトリでグループをグループに追加	無効(エラー・メッセージが戻されます)
パスワードの設定	ユーザー・パスワードを設定	Shared Services ネイティブ・ユーザー・パスワードを設定	無効(エラー・メッセージが戻されます)
Shared Services モードへの移行	Essbase から Shared Services モードに移行	無効(Essbase はすでに移行されています)	無効(エラー・メッセージが戻されます)
ユーザーを Shared Services モードに移行	無効(Shared Services モードにのみ適用されます)	失敗したユーザーを Shared Services モードに移行	無効(エラー・メッセージが戻されます)
グループを Shared Services モードに移行	無効(Shared Services モードにのみ適用されます)	失敗したグループを Shared Services モードに移行	無効(エラー・メッセージが戻されます)
移行に失敗したユーザーをリスト	無効(Shared Services モードにのみ適用されます)	移行に失敗したユーザーをリスト	無効(エラー・メッセージが戻されます)
移行に失敗したグループをリスト	無効(Shared Services モードにのみ適用されます)	移行に失敗したグループをリスト	無効(エラー・メッセージが戻されます)
アプリケーションの作成	essbase.sec ファイルにアプリケーションを作成	essbase.sec ファイルにアプリケーションを作成し、そのアプリケーションを Shared Services ネイティブ・ディレクトリに登録	essbase.sec ファイルにアプリケーションを作成
データベースの作成	essbase.sec ファイルにデータベースを作成	essbase.sec ファイルにデータベースを作成(Essbase データベースは Shared Services ネイティブ・ディレクトリに登録されません)	essbase.sec ファイルにデータベースを作成
アプリケーションの再登録	無効(Shared Services モードにのみ適用されます)	Shared Services ネイティブ・ディレクトリにアプリケーションを再登録	無効(Shared Services モードにのみ適用されます)
アプリケーションの削除	essbase.sec ファイルでアプリケーションを削除	essbase.sec ファイルでアプリケーションを削除し、Shared Services ネイティブ・ディレクトリでそのアプリケーションの登録を解除	essbase.sec ファイルでアプリケーションを削除

要求	Essbase ネイティブ・セキュリティ	Oracle Enterprise Performance Management System セキュリティ	Fusion セキュリティ
データベースの削除	essbase.sec ファイルでデータベースを削除	essbase.sec ファイルでデータベースを削除(Essbase データベースは Shared Services ネイティブ・ディレクトリに登録されません)	essbase.sec ファイルでデータベースを削除
ユーザーの設定	ユーザーのセキュリティ情報を設定	ユーザーのセキュリティ情報を設定	無効(エラー・メッセージが戻されます)
外部ユーザーの設定	外部プロバイダ・ユーザーのセキュリティ情報を設定	外部プロバイダ・ユーザーのセキュリティ情報を設定	無効(エラー・メッセージが戻されます)
ユーザーへのアクセス権の付与	ユーザーに system/application/database へのアクセス権を付与(essbase.sec ファイルに格納されます)	ユーザーに system/application/database へのアクセス権を付与(Shared Services ネイティブ・ディレクトリに格納されます)	無効(Shared Services CSS API からエラー・メッセージが戻されます)
グループへのアクセス権の付与	グループに system/application/database へのアクセス権を付与(essbase.sec ファイルに格納されます)	グループに system/application/database へのアクセス権を付与(Shared Services ネイティブ・ディレクトリに格納されます)	無効(Shared Services CSS API からエラー・メッセージが戻されます)
ユーザー/グループの表示	essbase.sec ファイルに格納されているユーザー/グループを表示	Oracle Hyperion Shared Services から Essbase にプロビジョニングされたユーザー/グループを表示	essbase.sec ファイルに格納されているユーザー/グループを表示(フィルタが割り当てられます)
ユーザーへのフィルタの付与	ユーザーにフィルタを付与(essbase.sec ファイルに格納されます)	ユーザーにフィルタを付与(essbase.sec ファイルに格納されます)	ユーザーにフィルタを付与(essbase.sec ファイルに格納されます)
グループへのフィルタの付与	グループにフィルタを付与(essbase.sec ファイルに格納されます)	グループにフィルタを付与(essbase.sec ファイルに格納されます)	グループにフィルタを付与(essbase.sec ファイルに格納されます)
フィルタの表示	essbase.sec ファイルに格納されているフィルタを表示	essbase.sec ファイルに格納されているフィルタを表示	essbase.sec ファイルに格納されているフィルタを表示

H

Essbase 32ビットおよび64ビットの互換性

この付録の内容

Essbase と 32 ビットおよび 64 ビットのクライアントおよびサーバーとの互換性

次の表に、32 ビットと 64 ビットのクライアントおよびサーバーの Essbase サーバーとの互換性を示します:

クライアント	サーバー	Essbase サーバー: クライアントが接続できるプラットフォーム
32 ビット Administration Services コンソール	32 ビット Administration Services サーバー	32 ビット、64 ビット
32 ビット Administration Services コンソール	64 ビット Administration Services サーバー	32 ビット、64 ビット
64 ビット Administration Services コンソール	64 ビット Administration Services サーバー	64 ビット
32 ビット Essbase Studio コンソール	32 ビット Essbase Studio サーバー	32 ビット、64 ビット
32 ビット Essbase Studio コンソール	64 ビット Essbase Studio サーバー	32 ビット、64 ビット
64 ビット Essbase Studio コンソール	64 ビット Oracle Essbase Studio サーバー	64 ビット
32 ビット Integration Services コンソール	32 ビット Integration Services サーバー	32 ビット、64 ビット
32 ビット Integration Services コンソール	64 ビット Oracle Essbase Integration Services サーバー	32 ビット、64 ビット
32 ビット Smart View	32 ビット Provider Services	32 ビット、64 ビット
32 ビット Oracle Hyperion Smart View for Office	64 ビット Provider Services	64 ビット

クライアント	サーバー	Essbase サーバー: クライアントが接続できるプラットフォーム
32 ビット Administration Services コンソール	32 ビット Provider Services	32 ビット、64 ビット
64 ビット Oracle Essbase Administration Services コンソール	64 ビット Provider Services	64 ビット
32 ビット Java API または XMLA クライアント・アプリケーション	32 ビット Provider Services	32 ビット、64 ビット
64 ビット Java API または XMLA クライアント・アプリケーション	64 ビット Provider Services	64 ビット

32 ビットおよび 64 ビットのプラットフォームでの Essbase API の互換性

Essbase では、32 ビット・プラットフォーム用と 64 ビット・プラットフォーム用に API が提供されるので、いずれのプラットフォームでも、Essbase サーバーに接続するインタフェースを持つクライアント・プログラムを作成したり、コンパイルしたりできます。

- Essbase の C API または Visual Basic API を使用して、32 ビット・プラットフォームで開発されたクライアント・プログラムは、32 ビット・プラットフォームで実行でき、32 ビットまたは 64 ビットのいずれの Essbase サーバーにも接続できます。
- 32 ビットの Essbase Visual Basic API を使用して開発され、プリコンパイルされたクライアント・プログラムは、64 ビット Essbase サーバーに接続する 64 ビット Windows プラットフォームで動作します。ただし、ドキュメントに記載された方法に従って 32 ビットの実行時環境が設定されている必要があります。
- Essbase C API を使用して 64 ビット・プラットフォーム用に開発されたクライアント・プログラム:
  - 64 ビットで実行し、32 ビットまたは 64 ビットの Essbase サーバーに接続できます
  - 32 ビット・プラットフォームでは実行できません。

  注意 64 ビット・プラットフォーム用に開発されたクライアント・プログラムでは、バイト整列に#pragma ディレクティブは必要ありません。
- Essbase Visual Basic API を使用して、64 ビット Windows 用のクライアント・プログラムは開発できません。

次の表に、Essbase の一連の API で開発されたクライアント・プログラムの互換性をまとめます:

クライアント開発: API のバージョンとプラットフォーム	クライアントが実行可能なプラットフォーム	Oracle Essbase サーバー: クライアントが接続できるプラットフォーム
32 ビット C API/ランタイム・クライアント	32 ビット	32 ビット、64 ビット
32 ビット VB API/ランタイム・クライアント	32 ビット Windows	32 ビット、64 ビット
	64 ビット Windows	64 ビット
32 ビット Java API または XMLA クライアント・アプリケーション	32 ビット Provider Services サーバー	32 ビット、64 ビット
32-bit Embedded Java (API クライアント・アプリケーション)		32 ビット、64 ビット
64 ビット C API/ランタイム・クライアント	64 ビット	32 ビット、64 ビット
64 ビット Java (API または XMLA クライアント・アプリケーション)	64 ビット Oracle Hyperion Provider Services サーバー	64 ビット
64-bit Embedded Java (API クライアント・アプリケーション)		64 ビット

用語集

!「感嘆符」を参照してください。

#MISSING「欠落データ」を参照してください。

2 パス 他のメンバーの計算済の値に依存するメンバーを再計算するために使用される、Essbase のプロパティです。2 パスのメンバーは、2 番目のパスの間にアウトラインから計算されます。

Calculation Manager Planning および Financial Management のユーザーがグラフィカルな環境でビジネス・ルールを設計、検証、管理するために使用できる、Enterprise Performance Management Architecture(EPMA)のモジュールの 1 つです。

CDF「カスタム定義関数」を参照してください。

CDM「カスタム定義マクロ」を参照してください。

Cookie Web サイトによってコンピュータ上に配置されたデータのセグメントです。

EPM Oracle インスタンス EPM System 製品のアクティブで動的なコンポーネント(実行時に変更できるコンポーネント)が含まれているディレクトリです。EPM Oracle インスタンス・ディレクトリの場所は構成時に EPM System コンフィグレータを使用して定義します。

EPM Oracle ホーム EPM System 製品で必要なファイルを含むミドルウェア・ホームのサブディレクトリです。EPM Oracle ホームの場所は、EPM System インストーラでのインストール中に指定されます。

essbase.cfg Essbase のオプションの構成ファイルです。管理者は、このファイルを編集して Essbase サーバー機能をカスタマイズできます。一部の構成は、Essbase クライアントで使用することにより、Essbase サーバーの設定を上書きできます。

EssCell 特定の Essbase データベース・メンバーの交差を示す値を取得するために、Essbase Spreadsheet Add-in のセルに入力する関数です。

ESSCMD Essbase の操作を対話的に実行したり、バッチ・スクリプト・ファイルから実行したりするための、コマンドライン・インタフェースです。

ESSLANG テキスト文字を解釈するために使用されるエンコード方式を定義する Essbase 環境変数です。「エンコード方式」も参照してください。

ESSMSH「MaxL シェル」を参照してください。

Extensible Markup Language (XML) データに属性を割り当てるタグのセットで構成される言語です。スキーマに基づいて、複数のアプリケーションの間で解釈可能です。

Extract-Transform-Load (ETL) データを抽出してアプリケーションに移行するための、データ・ソース固有のプログラムです。

GUI グラフィカル・ユーザー・インタフェース

ID 外部認証におけるユーザーまたはグループの固有の ID です。

Interactive Reporting 接続ファイル(.oce) データベース API(ODBC、SQL*Net、など)、データベース・ソフトウェア、データベース・サーバーのネットワーク・アドレス、データベース・ユーザー名などのデータベース接続情報を含むファイルです。管理者は、Interactive Reporting 接続(.oce)ファイルを作成して発行します。

Java アプリケーション・サーバー・クラスタ Java 仮想マシン(JVM)のアクティブ-アクティブ・アプリケーション・サーバー・クラスタです。

Java データベース接続(JDBC) Java ベース・クライアントとリレーショナル・データベースにより使用されるクライアントとサーバー間の通信プロトコルです。JDBC インタフェースにより、SQL ベースのデータベース・アクセスのための呼出しレベル API が提供されます。

JSP Java Server Page です。

KeyContacts ガジェット Smart Space ユーザーのグループを含み、Smart Space Collaborator へのアクセスを提供するガジェットです。たとえば、マーケティング・チームおよび開発チーム向けに個別の KeyContacts ガジェットを使用できます。「ガジェット」も参照してください。

LRO「リンク・レポート・オブジェクト」を参照してください。

MaxL Essbase で使用される多次元データベース向けアクセス言語です。データ定義の言語(MaxL DDL)とデータ操作の言語(MaxL DML)から構成されます。「MaxL DDL」、「MaxL DML」および「MaxL シェル」も参照してください。

MaxL DDL Essbase で、バッチまたは対話的なシステム管理のタスクに使用されるデータ定義の言語です。

MaxL DML Essbase で、データのクエリーと抽出に使用されるデータ操作の言語です。

MaxL DML の計算済メンバー 分析を目的として設計されたメンバーです。MaxL DML クエリーのオプションの WITH セクションで定義されます。

MaxL Perl モジュール Essbase MaxL DDL の一部である Perl モジュール(essbase.pm)です。このモジュールを Perl パッケージに追加すると、Perl プログラムから Essbase データベースにアクセスできます。

MaxL シェル MaxL ステートメントを Essbase サーバーに渡すためのインタフェースです。MaxL シェルの実行可能ファイル(UNIX は essmsh、Windows は essmsh.exe)は、Essbase の bin ディレクトリに格納されています。

MaxL スクリプト・エディタ 管理サービス・コンソールのスクリプト開発環境です。MaxL スクリプトで Essbase を管理するとき、テキスト・エディタと MaxL シェルの代替として MaxL スクリプト・エディタを使用できます。

MDX (多次元式) 多次元準拠データベースでクエリーと計算に使用される言語です。

MIME タイプ アイテムのデータ・フォーマットを示す属性です。システムは、これによってオブジェクトを開くアプリケーションを判断します。ファイルの Multipurpose Internet Mail Extension(MIME)タイプはファイル拡張子または HTTP ヘッダーにより決定されます。プラグインはブラウザに対して、サポートされる MIME タイプ、および各 MIME タイプに対応するファイル拡張子を通知します。

NULL 値 データのない値です。NULL 値はゼロに等しくありません。

ODBC Open Database Connectivity の略です。データベース管理システム(DBMS)の情報処理方法に関係なく、あらゆるアプリケーションにより使用されるデータベース・アクセスの方法です。

OLAP メタデータ・カタログ Essbase 統合サービスにおける、リレーショナル・データ・ソースから引き出されたデータの性質、ソース、場所および種別について説明するメタデータを含むリレーショナル・データベースです。

OLAP モデル Essbase Integration Services における、リレーショナル・データベースのテーブルおよび列から作成された論理モデル(スター・スキーマ)です。OLAP モデルは多次元データベースの構造を生成するために使用されます。「オンライン分析プロセス(OLAP)」も参照してください。

Open Database Connectivity (ODBC) 標準のアプリケーション・プログラミング・インタフェース(API)です。これにより、アプリケーションから複数のサードパーティ・データベースにアクセスできます。

Oracle ホーム 特定の製品に必要なインストール済ファイルを含むディレクトリで、ミドルウェア・ホームのディレクトリ構造内にあります。「ミドルウェア・ホーム」も参照してください。

PL 勘定(P&L) 損益勘定です。P&L は通常、会社の損益計算書を構成する費用勘定と収益勘定の集合を指します。

POV (視点) 行、列、またはページ軸に割り当てられていないメンバーを選択することにより、データ・フォーカスを設定する機能です。たとえば、FDM での POV の選択項目には、場所、期間、カテゴリ、およびターゲット・カテゴリが含まれる可能性があります。また、Smart View で POV をフィルタとして使用すると、Currency 次元を POV に割り当て、Euro メンバーを選択できます。データ・フォームで POV を選択すると、ユーロ建でデータが表示されます。

Production Reporting「SQR Production Reporting」を参照してください。

PVA「期別価額メソッド」を参照してください。

root メンバー 次元分岐における最上位のメンバーです。

Shared Services レジストリ Shared Services リポジトリの一部です。ほとんどの EPM System 製品(インストール・ディレクトリ、データベース設定、コンピュータ名、ポート、サーバー、URL、依存サービス・データなど)の EPM System 配置情報を管理します。

SPF ファイル SQR Production Reporting Server で作成される、プリンタに依存しないファイルです。フォント、間隔、ヘッダー、フッターなど、フォーマットされた実際のレポート出力を表します。

SQL スプレッドシート SQL クエリーの結果セットを表示するデータ・オブジェクトです。

SQR Production Reporting データ・アクセス、データ操作、および SQR Production Reporting ドキュメント作成のための専用プログラミング言語です。

Structured Query Language (SQL) リレーショナル・データベースに対する指示を処理するために使用される言語です。

TCP/IP「Transmission Control Protocol/Internet Protocol」を参照してください。

Transmission Control Protocol/Internet Protocol(TCP/IP) 異なるオペレーティング・システムおよび内部アーキテクチャを持つコンピュータをリンクする標準的な通信プロトコルのセットです。TCP/IP を使用すると、LAN および WAN に接続する多様なコンピュータとの間でのファイルの交換、メールの送信、およびデータの格納が可能です。

Unicode モードのアプリケーション 文字テキストが UTF-8 でエンコードされている Essbase アプリケーションです。様々な言語に設定されているコンピュータを使用するユーザーが、アプリケーション・データを共有できます。

WebLogic Server ホーム WebLogic Server インスタンスで必要なインストール済ファイルを含むミドルウェア・ホームのサブディレクトリです。WebLogic Server ホームは、Oracle ホームのピアです。

WITH セクション MaxL DML で、オプションで使用できるクエリーのセクションです。セットまたはメンバーを定義する再利用可能な論理を作成するために使用されます。WITH セクションでセットまたはカスタムのメンバーを一度定義すると、クエリー中に複数回参照できます。

Workspace ページ 複数のソース(ドキュメント、URL、その他のコンテンツ・タイプ)からのコンテンツを使用して作成されるページです。Oracle およびその他のソースからのコンテンツを集約するために使用できます。

ws.conf Windows プラットフォーム用の構成ファイルです。

wsconf_platform UNIX プラットフォーム用の構成ファイルです。

XML「Extensible Markup Language」を参照してください。

XOLAP アウトラインのメンバーのみを格納し、クエリー時にリレーショナル・データベースからすべてデータを取得する、Essbase の多次元データベースです。XOLAP は、集約ストレージ・データベースおよび重複メンバー名を含むアプリケーションをサポートします。

Y 軸スケール「調査」セクションに表示される、チャートの Y 軸上の値の範囲です。たとえば、各チャートについて一意の Y 軸スケールを使用したり、すべての詳細チャートに同一の Y 軸スケールを使用したり、または列内のすべてのチャートに同一の Y 軸スケールを使用することが可能です。多くの場合、共通の Y 軸スケールを使用すると、一目でチャートを比較できるようになります。

Zero Administration サーバー上のプラグインの最新バージョン番号を識別するソフトウェア・ツールです。

「OK」ステータス 集計ステータスの 1 つです。エンティティが集計済であり、階層の下にあるデータが変更されていないことを示します。

「データなし」ステータス 集計ステータスの 1 つです。特定の期間と勘定科目のデータがこのエンティティに含まれていないことを示します。

「ロック済」ステータス 集計ステータスの 1 つです。変更できないデータがエンティティに含まれていることを示します。

「変更済」ステータス エンティティのデータが変更されたことを示す集計ステータスです。

「影響」ステータス 親エンティティに連結する子エンティティの変更を示すステータスです。

「非アクティブ」ステータス エンティティの当期の連結が非アクティブ化されていることを示すステータスです。

お気に入りガジェット Reporting and Analysis ドキュメントと URL へのリンクを含むガジェットです。「ガジェット」も参照してください。

アウトライン 多次元データベースのデータベース構造です。すべての次元、メンバー、タグ、タイプ、集計、および算術的関係を含みます。データは、アウトラインに定義された構造に応じてデータベースに保管されます。

アウトライン同期 パーティション・データベースで、データベースのアウトラインの変更を他のデータベースに伝播するプロセスです。

アクセス権 リソースに対してユーザーが実行できる一連の操作です。

アクティビティ・レベルの承認 操作対象のデータに依存せず、アプリケーションへのユーザー・アクセス、およびアプリケーションで実行できるアクティビティのタイプを定義します。

アクティブ-アクティブ高可用性システム すべての使用可能メンバーが要求に対応でき、アイドル状態のメンバーがないシステムです。通常、アクティブ-アクティブ・システムは、アクティブ-パッシブ・システムより豊富な拡張性オプションを備えています。「アクティブ-パッシブ高可用性システム」と対比してください。

アクティブ-パッシブ高可用性システム 常に要求に対応するアクティブ・メンバーと、アクティブ・メンバーに障害が発生した場合にのみアクティブ化されるパッシブ・メンバーが存在するシステムです。「アクティブ-アクティブ高可用性システム」と対比してください。

アクティブ・サービス 実行タイプが保留ではなく開始に設定されているサービスです。

アセンブリ EPM System 製品またはコンポーネントのインストール・ファイルです。

アダプタ ターゲットおよびソース・システムのデータやメタデータにプログラムを統合するためのソフトウェアです。

アップグレード 新しいソフトウェアのリリースを配置、アプリケーションやデータを移動、前の配置から新しい配置へ情報をプロビジョニングするプロセスです。

アド・ホック・レポート エンド・ユーザーが動的に作成するオンライン分析クエリーです。

アプリケーション 1)特定のタスクまたはタスクのグループを実行するために設計されたソフトウェア・プログラムです(スプレッドシート・プログラム、データベース管理システムなど)。2)必要とされる特定の分析のセットまたはレポートのセット、あるいはその両方に対応するために使用される、次元および次元メンバーの関連するセットです。

アプリケーションの通貨 アプリケーションのデフォルトのレポート用通貨です。

アプリケーション・サーバー・クラスタ 同時に稼働し、信頼性と拡張性を実現するために連携している複数のアプリケーション・サーバーが緩やかに結合したグループです。ユーザーには 1 つのアプリケーション・サーバー・インスタンスのように見えます。「垂直アプリケーション・クラスタ」および「水平アプリケーション・クラスタ」も参照してください。

アプリケーション移行ユーティリティ アプリケーションとアーチファクトの移行に使用されるコマンド・ライン・ユーティリティです。

アプリケーション管理者 アプリケーションの設定、構成、維持、および制御の担当者です。アプリケーションのすべての権限およびデータ・アクセス権を持ちます。

アペンダ 宛先を意味する Log4j の用語です。

アーチファクト 個別のアプリケーションまたはリポジトリ・アイテムです(スクリプト、フォーム、ルール・ファイル、Interactive Reporting ドキュメント、財務レポートなど)。オブジェクトとも呼ばれます。

イメージ・ブックマーク Web ページやリポジトリ・アイテムへのグラフィック・リンクです。

インストール・アセンブリ EPM System インストーラにプラグインする製品インストール・ファイルです。

インデックス 1)疎データの組合せによりブロック・ストレージ・データベースでデータを取得する方法です。2)インデックス・ファイルを指します。

インデックス・キャッシュ インデックス・ページを含むバッファです。

インデックス・ファイル ブロック・ストレージのデータ取得情報を格納する Essbase ファイルです。ディスクに常駐し、インデックス・ページを含みます。

インデックス・ページ インデックス・ファイルの下位区分です。インデックス・ページには、データ・ブロックを指すポインタが含まれます。

インデックス項目 疎次元の交差へのポインタです。インデックス・エントリはディスク上のデータ・ブロックをポイントし、オフセットを使用してセルを検索します。

イントロスペクション データベース固有の関係に基づいて階層を判断するために、データ・ソースを詳細に検査することを指します。「スクレーピング」と対比してください。

インポート・フォーマット FDM における、ソース・ファイルの構造の定義です。これにより、ソース・データ・ファイルを FDM データのロード位置にロードできます。

エンコード方式 テキストの作成、保管、表示のためにビット組合せを文字にマッピングする方法の 1 つです。各エンコード方式には UTF-8 などの名前が付けられています。各エンコード方式では、それぞれの値は特定のビット組合せにマッピングされています。たとえば、UTF-8 では大文字の A は HEX41 にマッピングされています。「コード・ページ」、「ロケール」も参照してください。

エンタープライズ・ビュー Administration Services の機能の 1 つです。グラフィカルなツリー・ビューを使用して Essbase 環境を管理できます。エンタープライズ・ビューを使用すると、Essbase アーチファクトを直接操作できます。

エンティティ 部門、子会社、工場、地域、製品など、組織での財務報告単位例: 部門、子会社、工場、地域、製品や財務報告単位。

エージェント アプリケーションやデータベースの開始と停止、ユーザー接続の管理、ユーザー・アクセスのセキュリティ対策を行う Essbase サーバー・プロセスです。このエージェントは、ESSBASE.EXE と呼ばれます。

オンライン分析プロセス(OLAP) 複数ユーザーによりクライアントとサーバー間の計算を行える多次元の環境です。集計された企業データをリアル・タイムで分析するユーザーが使用します。OLAP システムの機能には、ドリルダウン、データのピボット、複素数計算、トレンド分析、およびモデリングが含まれます。

カスケード メンバー値のサブセットに対して複数のレポートを作成するプロセスです。

カスタム・カレンダ システム管理者が作成したカレンダです。

カスタム・プロパティ ユーザーが作成した次元または次元メンバーのプロパティです。

カスタム・レポート 設計レポート・モジュールで作成する複雑なレポートです。コンポーネントの様々な組合せによって構成されます。

カスタム定義マクロ(CDM) Essbase のマクロです。Essbase 計算機の関数および専用マクロ関数を使用して記述されます。カスタム定義マクロが使用する Essbase の内部マクロ言語により、計算関数を組み合せることが可能となり、複数の入力パラメータが処理されます。「カスタム定義関数」も参照してください。

カスタム定義関数(CDF) Java で開発され、MaxL により Essbase の標準計算スクリプト言語に追加された、Essbase 計算関数です。「カスタム定義マクロ」も参照してください。

カスタム次元 ユーザーが作成および定義する次元です。チャネル、製品、部署、プロジェクト、または地域がカスタム次元になることがあります。

カタログ・ペイン アクティブ・セクションで利用可能な要素のリストを表示する領域です。クエリーがアクティブ・セクションの場合はデータベース・テーブルのリストが表示されます。ピボットがアクティブ・セクションの場合は結果列のリストが表示されます。ダッシュボードの場合は埋込み可能セクション、グラフィック・ツール、およびコントロール・ツールが表示されます。

カテゴリ データ編成に使用するグループです(月など)。

カレンダ ユーザー定義の期間、およびその関係です。暦年または会計年度は、Q1、Q2、Q3、およびQ4 から構成されます。

ガジェット EPM のコンテンツを容易に表示し、Reporting and Analysis のコア機能にアクセスできる、軽量で単純な専用アプリケーションです。

キャッシュ データを一時的に保持する、メモリー内のバッファです。

キューブ 3 つ以上の次元を含むデータのブロックです。Essbase データベースはキューブです。

キューブ・スキーマ Essbase Studio におけるメジャーおよび階層などのメタデータ要素です。キューブの論理モデルを指します。

キューブ配置 Essbase Studio で、アウトラインを構築してデータを Essbase アプリケーションおよびデータベースにロードするために、モデルのロード・オプションを設定するプロセスです。

クエリー・ガバナー Essbase Integration Server のパラメータまたは Essbase サーバーの構成設定です。データ・ソースに対して実行されるクエリーの時間とサイズを制御します。

クラスタ 単一リソースとして動作して、タスクの負荷を共有し、フェイルオーバーのサポートを提供する一連のサーバーまたはデータベースです。システムにおける単一障害点となるサーバーやデータベースを排除します。

クラスタ・サービス システムとしてクラスタ・メンバーの操作を管理するソフトウェアです。クラスタ・サービスを使用すると、一連のリソースやサービスを定義して、クラスタ・メンバー間でのハートビート・メカニズムを監視し、これらのリソースやサービスをできるだけ効率よくかつ透過的に別のクラスタ・メンバーに移動できます。

クラスタ内部接続 ノード障害を検出するためにハートビート情報についてハードウェア・クラスタで使用されるプライベート・リンクです。

クラスタ棒グラフ カテゴリを横に並べたグラフです。垂直棒グラフでのみ使用されます。

クリーン・ブロック 計算スクリプトによってすべての次元が一度に計算された場合、または計算スクリプトで SET CLEARUPDATESTATUS コマンドが使用された場合の、データベース全体の計算が完了しているデータ・ブロックを指します。

クロス集計レポート テーブル・フォーマットでデータの分類および集計を行うレポートです。テーブルのセルには、交差する分類に合致するデータの集計結果が保管されています。たとえば、製品販売情報のクロス集計レポートに、列見出しとして Small や Large などのサイズ属性、行見出しとして Blue や Yellow などの色属性を表示できます。テーブルの中で Large と Blue が交差するセルには、サイズが Large のすべての Blue 製品の総売上げが表示されます。

グリッドの POV 行、列、またはページの交差に次元を配置せずに、グリッド上で次元メンバーを指定する手段です。レポート設計者はグリッド・レベルで POV の値を設定し、ユーザーの POV がグリッドに影響を与えないように防ぐことができます。次元に含まれるグリッドの値が 1 つのとき、その次元は行、列、またはページではなくグリッドの POV に配置します。

グループ 複数のユーザーに同様のアクセス権を割り当てるためのコンテナです。

グローバル・レポート・コマンド 別のグローバル・コマンドに置き換えられるか、またはファイルが終了するまで実行し続けるレポート・スクリプトのコマンドです。

コミット・アクセス Essbase の取引の処理方法に影響する Essbase カーネルの分離レベルです。コミット・アクセスでは、同時取引は書込みロックを長期間保持し、予測可能な結果を生成します。

コンテキスト変数 タスクフロー・インスタンスのコンテキストを特定するために、特定のタスクフローに定義される変数です。

コンテンツ リポジトリに格納されたあらゆるタイプのファイルの情報です。

コンテンツ・ブラウザ コンテンツを参照して選択し、Workspace ページに配置するために使用できるコンポーネントです。

コントリビューション 子エンティティから親に追加される値です。それぞれの子は親に対するコントリビューションを持ちます。

コード・ページ 一連のテキスト文字へのビット組合せのマッピングです。コード・ページは、それぞれ異なる文字セットをサポートします。各コンピュータには、ユーザーが必要とする言語の文字セットについてのコード・ページ設定が含まれます。このドキュメントでは、コード・ページは非 Unicode のエンコードのビット組合せに文字をマッピングします。「エンコード方式」も参照してください。

サイクル・スルー データベースを計算しながら、そのデータベース内で複数パスを実行します。

サイレント応答ファイル インストール管理者が提供する必要があるデータをかわりに提供するファイルです。応答ファイルによって、ユーザーが操作または入力しなくても EPM System インストーラや EPM System コンフィグレータは実行されます。

サブスクライブ 項目またはフォルダが更新されるときに自動的に通知を受け取るように、項目またはフォルダにフラグを付けることを指します。

サブ勘定科目のナンバリング 不連続の整数を使用してサブ勘定科目のナンバリングを行うためのシステムです。

サポート詳細 セルの値を算出した計算および仮定です。

サンプリング エンティティの特性を判別するためにエンティティの代表的な部分を選択するプロセスです。「メタデータのサンプリング」も参照してください。

サービス ビジネス項目を取得、変更、追加、および削除するためのリソースです(権限付与、認証など)。

サーブレット Web サーバーが実行可能なコンパイルされたコードです。

システム抽出 アプリケーションのメタデータから ASCII ファイルにデータを変換する機能です。

シナリオ データを分類するための次元です(Actuals、Budget、Forecast1、Forecast2 など)。

シリアル計算 デフォルトの計算設定です。1 つの計算を複数のタスクに分割して、一度に 1 つのタスクを計算します。

シングル・サインオン(SSO) 一度ログオンすれば、認証を再度求められることなく複数のアプリケーションにアクセスできる機能です。

シングル・ポイント障害 障害が発生した場合にユーザーが通常の機能にアクセスできなくなる、システムのコンポーネントです。

ジョブ 出力生成のために起動できる特殊なプロパティを持つドキュメントです。ジョブには Interactive Reporting、SQR Production Reporting、または汎用ドキュメントを含めることができます。

ジョブの出力 ジョブの実行によって生成されたファイルやレポートです。

スキーマ リレーショナル・データベースにおける、データおよびデータ間の関連を表す論理モデルです。

スクレーピング データ・ソースを検査して、最も基本的なメタデータ要素を得ることを指します。「イントロスペクション」と対比してください。

スケール スケールによって値の表示方法を決定します(整数、十単位、百単位、千単位、百万単位など)。

スコア ターゲットを達成するレベルです。通常はターゲットのパーセンテージとして表されます。

スコアカード 目標を達成する上での、従業員、戦略要素、または責任要素の進行状況を示すビジネス・オブジェクトです。スコアカードに追加された各メジャーおよび子スコアカードについて収集されるデータに基づいて、進行状況が確認されます。

スコープ Essbase の操作または設定により包含されるデータ領域です(セキュリティ設定の影響を受けるデータ領域など)。通常、スコープには 3 レベルの粒度があり、上位レベルが下位レベルを包含します。これらのレベルは上位から下位の順で、システム全体(Essbase サーバー)、Essbase サーバー上のアプリケーション、Essbase サーバー・アプリケーション内のデータベースとなります。「持続性」も参照してください。

ステージ 1)通常は個別のユーザーにより実行される、タスクフロー内の 1 つの論理ステップを形成するタスクの説明です。ステージには手動と自動の 2 つのタイプがあります。2)Profitability では、組織内での割当てプロセスのステップを表す、モデル内の論理区分です。

ステージング・テーブル 特定のアプリケーションの必要性に対応するために作成するデータベースです。ステージング領域は、1 つ以上の RDBMS のスナップショット(再構築されたバージョン)です。

ステージング領域 特定のアプリケーションの必要性に対応するために作成するデータベースです。ステージング領域は、1 つ以上の RDBMS のスナップショット(再構築されたバージョン)です。

ステージ・アクション 自動ステージで、ステージを実行するために呼び出されたアクションです。

ステージ内割当て 同じステージ内のオブジェクトに対する財務フローの割当てです。

ステージ後割当て 割当てモデルにおける割当てです。後に続くモデル・ステージの場所に割り当てられます。

ステータス・バー コマンド、勘定科目、およびデータ・ファイルの現在のステータスに関する有用な情報が表示される画面下部のバーです。

スナップショット 特定の時点の読取り専用データです。

スポットライタ 選択された条件に基づくカラー・コーディングを可能にするツールです。

スマート・カット URL フォームのリポジトリ項目へのリンクです。

スマート・タグ Microsoft Office アプリケーションでのキーワードです。スマート・タグのメニューから使用可能な定義済アクションに関連付けられています。Oracle EPM System 製品でも、スマート・タグを使用して Reporting and Analysis のコンテンツのインポートや Financial Management および Essbase の機能へのアクセスが可能です。

スーパーバイザ サーバーのすべてのアプリケーション、データベース、関連ファイル、セキュリティ機構にフル・アクセスできるユーザーです。

ズームチャート チャートを拡大することにより詳細情報を表示するためツールです。ズームチャートは、チャートに表示されるメトリックについて数値情報を詳細に表示できます。

セカンダリ・メジャー プライマリ・メジャーよりも優先度の低いメジャーです。セカンダリ・メジャーには業績レポートがありませんが、スコアカードで使用したり、次元メジャーのテンプレートを作成するために使用できます。

セキュリティ・エージェント Web アクセス管理プロバイダ(Oracle Access Manager、Oracle Single Sign-On または CA SiteMinder など)です。企業の Web リソースを保護します。

セキュリティ・プラットフォーム Oracle EPM System 製品で外部認証とシングル・サインオン機能を使用するためのフレームワークです。

セル 1)多次元データベースの次元の交差を表すデータ値です。ワークシート上の行および列の交差を指します。2)管理ドメインに属するノードの論理グループです。

セル・ノート Essbase データベースでセルに付けられるテキスト注釈です。セル・ノートは LRO の一種です。

タイトル・バー Strategic Finance 名、ファイル名、およびシナリオ名のバージョン・ボックスを表示するバーです。

タイムライン・ビューア 特定の場所について、完了したプロセス・フロー・ステップの日時を表示するための FDM の機能です。

タイム・イベント ジョブの実行をトリガーします。

タイム・スケール 指定された期間別(毎月、四半期ごとなど)にメトリックを表示するスケールです。

タスクフロー ビジネス・プロセスの自動化を指します。手続きのルールに従って、あるタスクフロー参加者から別の参加者にタスクが渡されます。

タスクフロー・インスタンス タスクフローの状態と関連データが含まれる、タスクフローの単一のインスタンスです。

タスクフロー参加者 手動ステージおよび自動ステージの両方について、タスクフローのステージのインスタンスに関連付けられているタスクを実行するリソースです。

タスクフロー定義 ステージとステージ間の関係のネットワーク、タスクフローの開始と終了を示す基準、および個別のステージに関する情報(参加者、関連アプリケーション、関連アクティビティなど)から構成される、タスクフロー管理システムのビジネス・プロセスです。

タスクフロー管理システム タスクフローを定義および作成し、その実行を管理するシステムです。定義付け、ユーザーまたはアプリケーションのやりとり、およびアプリケーションの実行可能ファイルが含まれます。

タスク・リスト 特定のユーザーについて、タスクの詳細ステータスを示すリストです。

タブ Strategic Finance で、勘定科目とレポートのナビゲーションを行うことができるビューです。

タプル MDX 構文の要素です。セルは、各次元からのメンバーの交差として参照されます。次元が削除されている場合、最上位のメンバーが示されます。次に例を示します: (Jan); (Jan, Sales); ([Jan], [Sales], [Cola], [Texas], [Actual])。

ダッシュボード 業務の要約を対話的に示すメトリックと指標の集まりです。ダッシュボードにより分析アプリケーションを構築して配置できます。

チャート・テンプレート ワークスペース・チャートに表示するメトリックを定義するテンプレートです。

テキスト・メジャー Essbase では、メジャーが示されている次元で Text としてタグ付けされたメンバーを指します。セルの値は定義済テキストとして表示されます。たとえば、Satisfaction Index というテキスト・メジャーについては、Low、Medium、および High という値を含めることがあります。「型付きメジャー」、「テキスト・リスト」および「導出テキスト・メジャー」も参照してください。

デフォルト通貨単位 データの単位スケールです。たとえば、千単位で分析を定義するように選択して 10 を入力すると、10,000 と解釈されます。

データベース接続 データ・ソースへの接続に使用する定義とプロパティを保管し、データベース参照を移動可能にして広く使用できるようにするファイルです。

データ・キャッシュ 非圧縮データ・ブロックを保持するメモリー内のバッファです。

データ・セル「セル」を参照してください。

データ・ファイル・キャッシュ 圧縮データ(PAG)ファイルを保持する、メモリー内のバッファです。

データ・フォーム Web ブラウザなどのインタフェースからデータベースにデータを入力でき、データまたは関連テキストを表示して分析できるグリッド表示です。一部の次元メンバー値は固定され、データが特定の視点から表示されます。

データ・モデル データベース・テーブルのサブセットを示します。

データ・ロック 指定された基準(期間、シナリオなど)に従ってデータの変更を防ぐ機能です。

データ・ロードのルール テキストベース・ファイル、スプレッドシートまたはリレーショナル・データ・セットからのデータをデータベースにロードする方法を決定する一連の基準です。

データ・ロード位置 FDM で、ソース・データをターゲット・システムに送信する報告単位です。通常、ターゲット・システムにロードされる各ソース・ファイルに対して FDM のデータ・ロード位置が 1 つあります。

データ値「セル」を参照してください。

データ関数 データを集約する関数です。データの平均、最大値、カウントを求めたり、他の統計値によりデータのグループを集約します。

トップ・ラベルとサイド・ラベル 列と行の見出しです。ピボット・レポートの上部とサイドにそれぞれ表示されます。

トラフィック・ライト 2 つの次元メンバーの比較または一定の制限値に基づいて、レポートのセルまたはピンを色分けする機能です。

トリガー ユーザーが指定した基準に従ってデータを監視するための、Essbase の機能の 1 つです。基準に一致すると、Essbase はユーザーまたはシステム管理者にアラートを送信します。

トレーサビリティ メタデータ要素を物理ソースまで追跡する機能です。たとえば Essbase Studio の場合、キューブ・スキーマをそれ自体の階層およびメジャー階層から次元要素、日付/時間要素、メジャー、そして最終的には物理ソース要素まで追跡できます。「系列」も参照してください。

トレース・レベル ログ・ファイルに取り込まれた詳細のレベルです。

トレース割当て 財務データのフローに対する視覚的な追跡を可能にする Profitability の機能です。この追跡は単一の交差から、モデル内の前方または後方に実行できます。

トークン 外部認証システム上の 1 つの有効なユーザーまたはグループの暗号化された ID です。

トースト・メッセージ 画面右下の隅で消えていくメッセージです。

ドライバ Profitability and Cost Management で、ドライバを使用する複数ソース間の数学的関係、およびこれらのソースがコストや収益を割り当てる宛先を示す、割当て方法の一種です。Business Modeling については、「コスト・ドライバ」および「アクティビティ・ドライバ」も参照してください。

ドリルスルー あるデータ・ソースの値から別のソースの対応するデータに移動することです。

ドリルダウン 次元の階層を使用してクエリー結果セット内をナビゲートすることです。ドリルダウンにより、ユーザーのパースペクティブが集約データから詳細に移ります。たとえば、ドリルダウンにより年と四半期の階層関係、または四半期と月の階層関係が明らかになります。

ネイティブ認証 サーバーまたはアプリケーション内で、ユーザー名とパスワードを認証するプロセスです。

ネスト列見出し 複数の次元からのデータを表示するレポート列の列見出しのフォーマットです。たとえば、Year と Scenario のメンバーが含まれる列見出しはネスト列です。ネスト列見出しでは、見出しの一番上の行の Q1(Year 次元)が、見出しの一番下の行の Actual および Budget(Scenario 次元)で修飾されます。

ハイパーテキスト・マークアップ言語(HTML) Web ブラウザでのデータ表示を指定するプログラミング言語です。

ハイパーリンク ファイル、Web ページまたはイントラネット HTML ページへのリンクです。

ハイブリッド分析 リレーショナル・データベースに格納された下位のデータを、Essbase に格納された要約レベルのデータにマッピングする分析です。リレーショナル・システムの大規模スケーラビリティと多次元データを組み合せます。

ハードウェア・クラスタ ネットワーク・サービス(たとえば IP アドレス)やアプリケーション・サービス(データベースや Web サーバーなど)のシングル・ビューを、これらのサービスのクライアントに提供するコンピュータの集合です。ハードウェア・クラスタの各ノードは、独自のプロセスを実行するスタンドアロン・サーバーです。これらのプロセスは互いに通信して、連携してアプリケーション、システム・リソースおよびデータをユーザーに提供する 1 つのシステムのようなものを形成します。

バックアップ アプリケーション・インスタンスの複製コピーです。

バッチ POV ユーザーの POV において、バッチに含まれる各レポートおよびブックのすべての次元の集合です。バッチのスケジュールを立てる際は、バッチ POV で選択されたメンバーを設定できます。

バッチ・ファイル 複数の ESSCMD スクリプトを呼び出して複数の ESSCMD セッションを実行できるオペレーティング・システム・ファイルです。Windows システムの場合、バッチ・ファイルには BAT というファイル拡張子が付けられます。UNIX の場合、バッチ・ファイルはシェル・スクリプトとして記述されます。

バッチ・ローダー 複数ファイルの処理を可能にする FDM コンポーネントです。

バッチ処理モード サーバー管理や診断のルーチン・タスクを自動化するために使用できるバッチやスクリプト・ファイルを記述するために、ESSCMD を使用する方法です。ESSCMD スクリプト・ファイルは複数のコマンドを実行でき、オペレーティング・システムのコマンドラインから実行したり、オペレーティング・システムのバッチ・ファイルから実行したりすることが可能です。バッチ・ファイルを使用すると、複数の ESSCMD スクリプトを呼び出したり、ESSCMD の複数インスタンスを実行したりできます。

バッチ計算 データベースにおいてバッチで実行される計算です(計算スクリプト、すべてのデータベース計算など)。動的計算はバッチ計算とはみなされません。

バージョン データのシナリオのコンテキスト内で使用される、起こりうる結果です。たとえば、Budget - Best Case と Budget - Worst Case では、Budget がシナリオであり、Best Case と Worst Case がバージョンです。

パターン照合 条件として入力されるアイテムの一部またはすべての文字と値を照合する機能です。欠落文字は、疑問符(?)またはアスタリスク(*)などのワイルド・カード値で表せます。たとえば、「Find all instances of apple」では apple が戻されるのに対して、「Find all instances of apple*」では apple、applesauce、applecranberry などが戻されます。

パフォーマンス・インディケータ ユーザーが指定した範囲に基づくメジャーおよびスコアカード・パフォーマンスを示すために使用されるイメージ・ファイルです。ステータス記号とも呼ばれます。デフォルトのパフォーマンス・インディケータを使用することもできますが、無制限に独自のパフォーマンス・インディケータを作成することも可能です。

パブリック・ジョブ・パラメータ 管理者が作成する再利用可能な名前付きジョブ・パラメータです。必要なアクセス権を持つユーザーがアクセスできます。

パブリック反復タイム・イベント 管理者が作成する再利用可能なタイム・イベントです。アクセス・コントロール・システムからアクセスできます。

パレット JASC に準拠し、.PAL 拡張子を持つファイルです。各パレットには相互に補完し合う 16 色が含まれ、ダッシュボードの色要素の設定に使用することが可能です。

パースペクティブ スコアボードのメジャーやアプリケーションにおける戦略目標をグループ化するために使用されるカテゴリです。パースペクティブにより、主要な利害関係者(顧客、従業員、株主、金融関係者など)またはキー・コンピテンシ領域(時間、コスト、品質など)を示すことができます。

パーティション化 データ・モデルの間で共有またはリンクされるデータの領域を定義するプロセスです。パーティション化は Essbase アプリケーションのパフォーマンスとスケーラビリティに影響することがあります。

パーティション領域 データベース内のサブ・キューブです。パーティションは、データベースの一部からの 1 つ以上のセル領域から構成されます。複製パーティションおよび透過パーティションの場合、2 つのパーティションが同じ形状となるために、領域内のセルの数がデータ・ソースとターゲットで同一となる必要があります。データ・ソース領域に 18 個のセルが含まれる場合、データ・ターゲット領域にも対応する 18 個のセルが含まれている必要があります。

ビジネス・プロセス 集合的にビジネス上の目標を達成するための一連のアクティビティです。

ビジネス・ルール 期待される一連の結果値を生成するためにアプリケーション内に作成される論理式または式です。

ビジュアル・キュー 特定のタイプのデータ値をハイライトする、フォントや色などのフォーマットが設定されたスタイルです。データ値は、次元メンバー、親メンバー、子メンバー、共有メンバー、動的計算、式を含むメンバー、読取り専用データ・セル、読取りおよび書込みデータ・セル、またはリンク・オブジェクトのいずれかになります。

ビュー 年次累計または期別のデータ表示です。

ピボット 取得したデータのパースペクティブを変更します。Essbase では、まず次元が取得され、データが行に展開されます。その後、データのピボット(並替え)を行うことにより、異なる視点を得ることができます。

ピン ピンボードと呼ばれるグラフィック・レポート上に配置される対話型アイコンです。ピンは動的です。ピンは、基盤となるデータ値や分析ツールの基準に基づいて、イメージやトラフィック・ライトの色を変更できます。

ピンボード 3 種類のデータ・オブジェクトの表示タイプの 1 つです。ピンボードは、背景およびピンと呼ばれる対話型アイコンから構成されるグラフィックです。ピンボードを使用するにはトラフィック・ライトを定義する必要があります。

ファイルの区切り文字 データ・ソース内のフィールドを区切る文字です(カンマ、タブなど)。

ファクト・テーブル スター結合スキーマの中心のテーブルです。外部キー、および次元テーブルから取得した要素により特徴付けられます。通常、このテーブルにはスキーマの他のすべてのテーブルに関連する数値データが含まれます。

フィルタ データ・セットで、特定の基準に従って値を制限する制約です。たとえば、特定のテーブル、メタデータ、または値を除外したり、アクセスを制御したりする場合に使用されます。

フェイルオーバー プライマリ・データベース、サーバーまたはネットワークに障害が発生したり、これらがシャットダウンしたりする場合に、冗長性のあるスタンバイ・データベース、サーバーまたはネットワークに自動的に切り替える機能です。フェイルオーバー用にクラスタリングされているシステムは、高可用性、サーバーの冗長性を利用したフォルト・トレランス、および共有ディスクなどのフォルト・トレラント・ハードウェアを提供します。

フォーマット文字列 1) Essbase では、セル値の表示方法を変換する方法です。2) Data Relationship Management では、Format または Formatted Date 派生プロパティのパラメータで、返すプロパティ値のフォーマットを指定します。

フッター レポート・ページ下部に表示されるテキストまたはイメージです。ページ番号、日付、ロゴ、タイトル、ファイル名、作成者名など、動的な関数や静的なテキストが含まれます。

フリーフォーム・グリッド 動的計算のために、複数のソースからのデータを提示、入力、および統合するためのオブジェクトです。

フリーフォーム・レポート作成 ワークシートに次元メンバーまたはレポート・スクリプト・コマンドを入力することにより、レポートを作成することを指します。

フレーム デスクトップ上の領域です。ナビゲーション・フレームとワークスペース・フレームが 2 つの主要な領域となります。

フロー勘定科目 期別と年次累計の符号なしの値を格納する勘定科目です。

ブック 1) Financial Reporting では、類似したドキュメントのグループを保持するコンテナです。ブックは、次元セクションまたは次元の変更を指定する場合があります。2) Data Relationship Management では、グループとして同時に実行できるエクスポートの集合です。エクスポート結果は、結合することも、個別に出力することもできます。

ブックの POV ブックが実行される次元のメンバーです。

ブックマーク ユーザー個人のページに表示されるレポート・ドキュメントまたは Web サイトへのリンクです。ブックマークのタイプには、マイ・ブックマークとイメージ・ブックマークがあります。

ブロック プライマリ・ストレージ・ユニットです。多次元配列であり、すべての密次元のセルを表します。

ブロックされた勘定科目 手動で入力する必要があるために連結ファイルの計算に含めない勘定科目です。

ブロック・ストレージ・データベース 疎次元に定義されたデータ値の密度に基づいてデータを分類および格納する、Essbase のデータベース・ストレージ・モデルです。データ値はブロック単位で格納され、ブロックは値を含む疎次元メンバーについてのみ存在します。

ブロードキャスト・メッセージ Planning アプリケーションにログオンしているユーザーに対して管理者が送信する単純なテキスト・メッセージです。メッセージには、システムの可用性、アプリケーション・リフレッシュの通知、アプリケーションのバックアップなどの情報が詳細に表示されます。

プライマリ・メジャー 企業および事業のニーズにとって重要な、優先度の高いメジャーです。コンテンツ・フレームに表示されます。

プランナ データの入力と送信、他のプランナが作成したレポートの使用、ビジネス・ルールの実行、タスク・リストの使用、電子メール通知の使用、および Smart View の使用が可能なユーザーです。プランナは、大多数のユーザーから構成されます。

プランニング・ユニット シナリオ、バージョン、およびエンティティの交差におけるデータ・スライスです。プラン・データの準備、確認、注釈付け、および承認のための基本単位です。

プレゼンテーション Web Analysis ドキュメントのプレイリストです。レポートの分類、整理、並べ替え、配布、および確認を行うことができます。リポジトリ内のレポートを参照するポインタを含みます。

プロキシ・サーバー セキュリティを保証するために、ワークステーション・ユーザーとインターネットの間で仲介を行うサーバーです。

プロジェクト 実装でグループ化された Oracle Hyperion 製品のインスタンスです。たとえば、Planning プロジェクトには Planning アプリケーション、Essbase キューブ、Financial Reporting サーバー・インスタンスが含まれることがあります。

プロセス監視レポート FDM データ変換プロセスにおける、場所とその位置のリストです。プロセス監視レポートを使用して、決算手続のステータスを監視できます。レポートにはタイム・スタンプが付けられるので、時間データがロードされた場所を判断するために使用できます。

プロット領域 X 軸、Y 軸、および Z 軸で囲まれている領域です。円グラフの場合は、その周りに表示される長方形の領域です。

プロビジョニング ユーザーおよびグループに対して、リソースへのアクセス権限を付与するプロセスです。

ページ グリッドまたはテーブルでの情報表示の一種です。しばしば Z 軸により示されます。ページには、1 つのフィールドからのデータ、計算により得られるデータ、またはテキストを含めることができます。

ページ・ファイル Essbase のデータ・ファイルです。

ページ・メンバー ページ軸を決定するメンバーです。

ページ見出し レポートの現在のページで表示されているメンバーをリストした、レポート見出しの一種です。ページ上のすべてのデータ値には、ページ見出し内のメンバーが共通属性として適用されています。

ホスト アプリケーションとサービスがインストールされているサーバーです。

ホスト・プロパティ ホストに関係するプロパティです。ホストに複数の Oracle EPM ホームが含まれる場合は、いずれかの Oracle EPM ホームに関係するプロパティとなります。

マイ Workspace ページ ユーザーが作成するカスタマイズ可能な Workspace ページです。リポジトリを参照することなく 1 つの場所から簡単にアクセスできるように特別なマークが付いています。

マスター・データ・モデル 複数のクエリーによりソースとして参照される独立するデータ・モデルです。このモデルが使用される場合は、「クエリー」セクションのコンテンツ・ペインに「ロック済データ・モデル」と表示されます。データ・モデルは「データ・モデル」セクションに表示されるマスター・データ・モデルにリンクされています(管理者によって非表示になっていることがあります)。

マップ・ナビゲータ 戦略、責任、および因果関係の各マップに現在の位置を示す機能です。赤色のアウトラインで示されます。

マップ・ファイル 外部データベースとの間でのデータの送信や取得に使用される定義を格納するファイルです。マップ・ファイルの拡張子は、データ送信用が.mps、データ取得用が.mpr です。

マルチロード 複数の期間、カテゴリ、および場所を同時にロードすることを可能にする FDM の機能です。

マージ データ・ロード・オプションの 1 つです。データ・ロード・ファイルで指定された勘定科目の値のみを消去し、データ・ロード・ファイルの値で置換します。

ミドルウェア・ホーム Oracle WebLogic Server ホームを含み、EPM Oracle ホームおよびその他の Oracle ホームも含むことができるディレクトリです。ミドルウェア・ホームは、ローカル・ファイル・システム、または NFS を介してアクセス可能なリモート共有ディスク上に配置できます。

ミニスキーマ データ・ソースからのテーブルのサブセットをグラフィカルに示したものです。データ・モデリングのコンテキストを表します。

ミニレポート レポートのコンポーネントの 1 つです。レイアウト、コンテンツ、ハイパーリンク、およびレポートのロード用の一つまたは複数のクエリーを含みます。各レポートには、1 つ以上のミニレポートを含めることができます。

メジャー OLAP データベースのキューブに含まれる数値で、分析に使用されます。メジャーには、利益幅、売上原価、売上数量、予算などがあります。「ファクト・テーブル」も参照してください。

メタアウトライン Essbase Integration Services における、OLAP モデルから Essbase アウトラインを作成するための構造とルールを含んでいるテンプレートです。

メタデータ データベースに格納された、またはアプリケーションにより使用されるデータのプロパティと属性を定義および説明するデータ・セットです。メタデータには、次元名、メンバー名、プロパティ、期間、およびセキュリティなどが含まれます。

メタデータのサンプリング ドリルダウン操作で次元に含まれるメンバーのサンプルを取得するプロセスです。

メタデータ・セキュリティ ユーザーにより特定のアウトライン・メンバーへのアクセスを制限するための、メンバー・レベルのセキュリティ・セットです。

メタデータ要素 データ・ソースから算出されるメタデータ、および Essbase Studio で使用するために格納され、カタログが作成されるおよびその他のメタデータです。

メンバー 次元内の個別のコンポーネントです。メンバーにより、類似する単位の集まりが個別に特定および区別されます。たとえば、時間次元には Jan、Feb、および Qtr1 などのメンバーが含まれることがあります。

メンバー・リスト 次元のメンバー、関数、他のメンバー・リストを示す名前付きのグループです。システムまたはユーザーにより定義されます。

メンバー・ロード Essbase 統合サービスにおける、次元およびメンバーを(データなしに)Essbase アウトラインに追加するプロセスです。

メンバー専用レポート・コマンド レポート・ライターのフォーマット・コマンドの 1 つです。レポート・スクリプトで現れると実行されます。このコマンドは関連するメンバーにのみ影響し、メンバーを処理する前にフォーマット・コマンドを実行します。

メンバー選択レポート・コマンド 兄弟、世代、レベルなどのアウトラインの関係に基づいて、メンバーの範囲を選択するレポート・ライター・コマンドの一種です。

モデル 1)アプリケーション固有のデータ表現を含むファイルまたはコンテンツの文字列です。モデルは Shared Services により管理される基本データであり、次に示す 2 つの主要なタイプがあります: 次元および非次元のアプリケーション・オブジェクト; 2)ビジネス・モデリングで、検査対象の領域からの業務および財務上のフローを示し、計算するために接続されたマシン・ネットワークです。

ユーザー・ディレクトリ ユーザーおよびグループの情報を集中管理する場所です。リポジトリまたはプロバイダとも呼ばれます。一般的に使用されているユーザー・ディレクトリには、Oracle Internet Directory (OID)、Microsoft Active Directory (MSAD) および Sun Java System Directory Server があります。

ユーザー変数 ユーザーのメンバー選択に基づいてデータ・フォームを動的に配置し、指定されたエンティティのみを表示する変数です。たとえば、Department というユーザー変数を使用すると、特定の部署および従業員を表示できます。

ユーザー定義メンバー・リスト ユーザー定義による、特定の次元に含まれるメンバーの静的なセットです。

ユーザー定義属性(UDA) アウトラインのメンバーに関連付けられ、メンバーの特性を説明する属性です。これを使用すると、指定された UDA が関連付けられているメンバーのリストが戻されます。

ライトバック 取得を行うスプレッドシートなどのクライアントが、データベースの値を更新する機能です。

ライフサイクル管理 製品環境間でアプリケーション、リポジトリまたは個別のアーチファクトを移行するプロセスです。

ライン・アイテムの詳細 勘定科目で最も下位の詳細レベルです。

リソース システムにより管理されるオブジェクトまたはサービスです(役割、ユーザー、グループ、ファイル、ジョブなど)。

リポジトリ ビューおよびクエリーに使用するためのメタデータ、フォーマットおよび注釈の情報を格納します。

リレーショナル・データベース 関連する 2 次元テーブルにデータを保管するデータベースです。「多次元データベース」と対比してください。

リンク 1)リポジトリ・オブジェクトへの参照です。リンクは、フォルダ、ファイル、ショートカットおよび他のリンクを参照できます。2)タスクフローで、あるステージのアクティビティが終了して次のアクティビティが開始するポイントです。

リンク・データ・モデル リポジトリのマスター・コピーにリンクされたドキュメントです。

リンク・パーティション データ・セルを使用して 2 つのデータベースをリンクするための共有パーティションです。ワークシートのリンク・セルをクリックすると、リンク・データベースの次元を示す新しいシートが開きます。これにより、表示される次元をドリルダウンできます。

リンク・レポート・オブジェクト(LRO) セル・ノート、URL、またはテキスト、オーディオ、映像、画像を含むファイルなどの外部ファイルへのセルベースのリンクです。Financial Reporting では、Essbase LRO 向けにサポートされるのはセル・ノートのみです。「ローカル・レポート・オブジェクト」と対比してください。

リンク条件 タスクフローのステージを順序付けるためにタスクフロー・エンジンにより評価される論理式です。

レイアウト領域 コンテンツを配置できる Workspace ページの領域です。

レイヤー 1)階層構造内で横並びにメンバーを含む場所です。世代(上から下へ)またはレベル(下から上へ)により指定されます。2)他のオブジェクトに対して相対的なオブジェクトの場所です。たとえば、Sample Basic データベースでは Qtr1 と Qtr4 は同じ年に含まれるので、世代が同一であることになります。しかし、不均衡階層を含むデータベースの場合、Qtr1 と Qtr4 は同一世代であっても同じレイヤーに位置しないことがあります。

レコード データベースで、1 つの完全な入力項目を形成するフィールドのグループです。たとえば、顧客レコードには、名前、住所、電話番号、および販売データのフィールドが含まれることがあります。

レベル 階層ツリー構造において、データベース・メンバーの関係を定義するレイヤーです。レベルは一番下の次元メンバー(レベル 0)から上位の親メンバーへと並べられます。

レベル 0 のブロック 疎のレベル 0 メンバーの組合せに使用されるデータ・ブロックです。

レベル 0 のメンバー 子の存在しないメンバーです。

レポートの通貨 財務諸表を準備するために使用される通貨です。現地通貨からレポートの通貨に変換されます。

レポート・エクストラクタ スクリプトの実行時に、Essbase データベースからのレポート・データを取得する Essbase コンポーネントです。

レポート・オブジェクト レポートの設計において、テキスト・ボックス、グリッド、イメージ、チャートなどの動作や外観を定義するプロパティを持つ基本要素です。

レポート・スクリプト 1 つまたは複数の運用レポートを生成する、Essbase レポート・ライター・コマンドを格納したテキスト・ファイルです。

レポート・ビューア レポート・スクリプトの実行後に完全なレポートを表示する Essbase コンポーネントです。

ログ・アナライザ Essbase ログのフィルタ、検索、および分析を行うための Administration Services の機能です。

ロケーション別名 データ・ソースを特定する記述子です。ロケーション別名により、サーバー、アプリケーション、データベース、ユーザー名、およびパスワードが指定されます。ロケーション別名は、DBA のデータベース・レベルで管理サービス・コンソール、ESSCMD、または API を使用して設定されます。

ロケール コンピュータで使用される言語、通貨および日付のフォーマット、データのソート順、および文字セットのエンコード方式を指定するコンピュータ設定です。Essbase ではエンコード方式のみが使用されます。「エンコード方式」、「ESSLANG」も参照してください。

ロケール・ヘッダー・レコード スクリプトなど、一部の非 Unicode でエンコードされたテキスト・ファイルの先頭で、エンコード・ロケールを特定するテキスト・レコードです。

ロック済 ユーザーやプロセスがデータを変更するのを防ぐために、ユーザーが呼び出すプロセスです。

ロック済データ・モデル ユーザーが変更できないデータ・モデルです。

ローカル・レポート・オブジェクト Explorer で Financial Reporting レポート・オブジェクトにリンクされていないレポート・オブジェクトです。「リンク・レポート・オブジェクト」と対比してください。

ローカル結果 データ・モデルのクエリー結果です。ローカルの結合で結果を使用する場合は、結果をデータ・モデルにドラッグして挿入できます。ローカルの結果を要求すると、カタログに表示されます。

ロード・バランサ 要求をクラスタの個々のアプリケーション・サーバーに分散するハードウェアおよびソフトウェアであり、システムへの唯一のエントリ・ポイントです。

ロード・バランシング サーバー・グループ全体で要求を分散することです。これによって、エンド・ユーザーのパフォーマンスが最適化されます。

ロールアップ「連結」を参照してください。

ワイルド・カード 検索文字列で単一の文字(?)または文字グループ(*)を示す文字です。

ワークフロー FDM でデータを最初から最後まで処理するために必要なステップです。ワークフローは、インポート(GL ファイルからのデータ・ロード)、検証(すべてのメンバーが有効なアカウントにマッピングされていることの確認)、エクスポート(マッピングされたメンバーのターゲット・アプリケーションへのロード)、およびチェック(ユーザー定義の検証ルールを使用してデータを処理することにより、データの精度を確認)から構成されます。

ワークブック 多数のワークシートを含むスプレッドシート・ファイル全体です。

一意でないメンバー名「重複メンバー名」を参照してください。

一意のメンバー名 データベース・アウトライン内に一度のみ存在する、共有されないメンバーの名前です。

一意メンバーのアウトライン 重複メンバー名を使用できないデータベース・アウトラインです。

三角換算法 残高をある通貨から別の通貨へ第三の共通通貨を介して変換する方法です。たとえば、デンマーク・クローネから英国ポンドへ残高を変換するには、残高をクローネからユーロへ、ユーロからポンドへ変換できます。

上位レベル・ブロック 少なくとも 1 つの疎メンバーが親レベルのメンバーになっているデータ・ブロックです。

世代 データベースでのメンバー関係を定義する階層ツリー内のレイヤーです。世代は、次元の最上位のメンバー(世代 1)から子メンバーへと 1 世代ずつ下に配置されていきます。一意の世代名を使用すると、階層ツリー内のレイヤーを特定できます。

並列エクスポート Essbase データを複数のファイルにエクスポートする機能です。並列エクスポートは、1 つのファイルにエクスポートした場合に比べて時間を短縮できます。また、1 つのデータ・ファイルでサイズが大きくなりすぎた場合の、オペレーティング・システムでの操作上の問題を解決できます。

並列データ・ロード Essbase で、複数プロセスのスレッドによりデータ・ロードのステージを同時に実行することを指します。

並列計算 計算オプションの 1 つです。Essbase では計算がタスクに分割され、一部のタスクは同時に計算されます。

事前計算 ユーザーが取得する前にデータベースの計算を実行することです。

交差 多次元データベース内の次元の交差を表すデータの単位。ワークシートのセル。

仕訳(JE) シナリオと期間の勘定科目残高の借方または貸方に計上する調整の集合です。

代替階層 共有メンバーの階層です。代替階層はデータベース・アウトラインの既存の階層に基づきますが、次元に代替レベルを持ちます。代替階層により、異なる視点(POV)から同一データを表示できます。

会社間消去「消去」を参照してください。

会社間照合 アプリケーション内の会社間勘定科目の対の残高を比較するプロセスです。会社間の受取勘定科目は、対応する会社間の支払い勘定科目と比較されます。一致する勘定科目は、組織の連結合計から会社間のトランザクションを消去するために使用されます。

会社間照合レポート 会社間勘定科目の残高を比較して、勘定の収支が合っているかどうかを示すレポートです。

使用済ブロック 最後に計算された後に変更されたセルを含むデータ・ブロックです。子ブロックが使用済である場合(つまり更新されている場合)、上位のブロックにも使用済のマークが付けられます。

例外 事前定義済の条件を満たす値です。フォーマット・インディケータを定義したり、例外が生成されたときに登録ユーザーに通知したりできます。

依存エンティティ 組織内の他のエンティティに属するエンティティです。

保存された仮定 ビジネス上の主要な計算を推進するための、Planning でのユーザー定義の仮定です(事業所の床面積 1 平方フィート当たりのコストなど)。

保管階層 集約ストレージ・データベースのアウトラインのみで、アウトラインの構造に従ってメンバーが集約される階層を指します。保管階層のメンバーには、式を含められないなどの一定の制限があります。

信頼できるユーザー 認証されたユーザーです。

修飾名 定まったフォーマットのメンバー名です。重複メンバーのアウトラインにおいて、重複メンバー名を区別します([Market].[East].[State].[New York]、[Market].[East].[City].[New York]など)。

個人の反復タイム・イベント 再利用可能なタイム・イベントです。作成したユーザーのみがアクセスできます。

個人用ページ リポジトリ情報を参照するための個人用ウィンドウです。表示する情報、およびレイアウトと色を選択します。

個人用変数 複雑なメンバー選択の特定の選択ステートメントです。

値次元 入力値、換算値および連結の詳細を定義するために使用される次元です。

兄弟 他の子メンバーと同じ世代で、すぐ上に同じ親を持つ子メンバーです。たとえば、メンバー Florida とメンバー New York はメンバー East の子であり、互いの兄弟です。

入力データ 計算されるのではなくソースからロードされるデータです。

共有 Workspace ページ 専用のシステム・フォルダに格納され、組織全体で共有する Workspace ページです。権限を持つユーザーが共有 Workspace ページの「ナビゲート」メニューからアクセスできます。

共有ストレージ フェイルオーバー・クラスタのすべてのノードに対して使用できる必要のあるデータを含むディスク・セットです。共有ディスクとも呼ばれます。

共有ディスク「共有ストレージ」を参照してください。

共有メンバー ストレージ・スペースを別の同名メンバーとの間で共有するメンバーです。Essbase アウトラインに複数回現れるメンバーが重複して計算されることを防ぎます。

再構成 URL ユーザーが Workspace にログオンしているときに、サーブレット構成設定を動的に再ロードするための URL です。

再構築 データベース・インデックス、また場合によってはデータ・ファイルの再生成もしくは再構築を行う操作です。

冗長データ 重複データ・ブロックです。Essbase で更新されたブロックがコミットされるまで、取引の間保持されます。

凡例ボックス 次元のデータ・カテゴリを特定するためのラベルを含むボックスです。

出資比率 エンティティが親によって所有される程度です。

分離レベル データベース操作のロックとコミットの動作を決定する Essbase カーネルの設定です。選択肢は: コミット・アクセスまたはアンコミット・アクセス。

列 Data Relationship Management における、インポート・ソース、またはクエリー、比較、検証またはエクスポートの結果に関連付けられたデータのフィールドです。

別名テーブル メンバーの代替名を含むテーブルです。

割当て 割当てモデルでのソースと宛先の関連付けです。割り当てられたコストや収益のフローの方向を制御します。

加重 スコアカードのアイテムに割り当てられた値です。スコアカード全体のスコアの計算において、そのアイテムの相対的な重要性を示します。スコアカードのすべてのアイテムの加重を総計すると 100% になります。たとえば、ある製品について新機能を開発する重要性を認識する場合、開発者のスコアカード上の New Features Coded のメジャーに Number of Minor Defect Fixes のメジャーよりも大きな加重が割り当てられます。

動的ビュー勘定科目 勘定科目の 1 つです。勘定科目の値は、表示されているデータから動的に計算されます。

動的メンバー・リスト システムにより作成される名前付きメンバー・セットです。ユーザーが定義した基準が使用されます。アプリケーションでリストが参照されるとき、リストは自動的にリフレッシュされます。次元メンバーの増減に応じて基準が自動的にリストに適用され、変更内容が反映されます。

動的レポート レポートを実行するときに更新されるデータを含むレポートです。

動的参照 データ・ソース内のヘッダー・レコードを指すルール・ファイル内のポインタです。

動的時系列 ブロック・ストレージ・データベースで、期間累計のレポート作成を実行するプロセスです。

動的計算 Essbase で、動的計算メンバー、または動的計算および保管メンバーとしてタグ付けされているメンバーについてデータを取得する場合にのみ実行される計算です。メンバーの値は、バッチ計算で事前に計算されるのではなく、取得時に計算されます。

動的計算および保管メンバー ブロック・ストレージ・アウトラインで、値を最初に取得したときにのみ Essbase によって計算されるメンバーです。計算された値はデータベースに保管され、2 回目以降の取得では計算を実行する必要がありません。

動的計算メンバー ブロック・ストレージ・アウトラインで、Essbase により取得時にのみ計算が行われるメンバーです。取得要求の処理が完了すると、計算された値は破棄されます。

動的階層 集約ストレージ・データベースのアウトラインに限定して、取得時に値が計算されるメンバーの階層です。

勘定科目のブロック 連結ファイルで勘定科目が入力データを受け入れるプロセスです。ブロックされた勘定科目は加算連結プロセスで値を受け取りません。

勘定科目の消去 連結時に連結ファイル内で値がゼロに設定された勘定科目です。

勘定科目タイプ 時間の経過に伴う勘定科目の値のフロー、およびその符号の振舞いを決定するプロパティです。勘定科目の種別のオプションには、支出、収益、資産、負債、および資本が含まれます。

単項演算子 アウトラインのメンバーに関連付けられている算術インディケータ(+、-、*、/、%)です。単項演算子では、データベースのロールアップ中のメンバーの計算方法が指定されます。

原点 2 つの軸の交差です。

反復タイム・イベント ジョブの実行開始点と実行頻度を指定するイベントです。

反復テンプレート 各期間に対して同一の調整を行うための仕訳テンプレートです。

収益勘定科目 期別の値と年次累計値を格納する勘定科目です。値が正の場合は純利益が増えます。

同期 Shared Services とアプリケーション・モデルの同期です。

同期済 モデルの最新バージョンがアプリケーションと Shared Services の両方に存在する状態を指します。「モデル」も参照してください。

名前付きセット MaxL DML で、MaxL DML クエリーのオプションの WITH セクションに定義された論理を使用するセットです。名前付きセットはクエリー内で複数回参照することが可能です。

因果関係マップ 企業戦略を形成する要素の関連、およびこれらの要素が組織の戦略目標を達成するためにどのように連動するかを示すマップ。因果関係マップのタブは、各戦略マップについて自動的に作成されます。

垂直アプリケーション・クラスタ 複数のアプリケーション・サーバー・インスタンスが同じマシン上にあるクラスタです。

型付きメジャー Essbase で、メジャーが示されている次元で Text または Date というタグが付けられたメンバーです。セルの値は事前定義されたテキストまたは日付として表示されます。

基本エンティティ 組織の構造の一番下に位置し、他のエンティティを持たないエンティティです。

基本次元 1 つまたは複数の属性次元に関連付けられている標準次元です。たとえば、製品に香りがあるとすると、Product 次元が Flavors 属性次元の基本次元となります。

基本通貨 日常の業務取引が行われる通貨です。

変換 アプリケーションの移行後、宛先環境で適切に動作するようアーティファクトを変換するプロセス。

変換元通貨 為替レートを使用して変換先通貨に変換される前の、値の元の通貨です。

変換先通貨 残高の変換後の通貨です。為替レートを入力して、変換元通貨から変換先通貨に変換します。たとえばユーロから米ドルに変換する場合、変換先の通貨は米ドルです。

外部でトリガーされるイベント ジョブの実行をスケジュールするための、時間ベースでないイベントです。

外部認証 アプリケーションの外部に格納されたユーザー情報を使用して、Oracle EPM System 製品にログオンすることを指します。ユーザー・アカウントは EPM System で維持されますが、パスワード管理およびユーザー認証は Oracle Internet Directory (OID) や Microsoft Active Directory (MSAD)などの企業ディレクトリを使用して外部サービスで実行されます。

多次元データベース 3 つ以上の次元でデータを整理、格納、および参照する方法です。次元のセットが交差するポイントが個別の値となります。「リレーショナル・データベース」と対比してください。

子 データベース・アウトライン内で親を持つメンバーです。

子孫 データベース・アウトラインで親の下に位置するメンバーです。たとえば、年、四半期および月を含む次元では、メンバー Qtr2 およびメンバー April がメンバー Year の子孫となります。

安全率 より安全な投資から期待される利回りです (米国の長期国債など)。

定義済ドリル・パス データ・モデルでの定義に従って次の詳細レベルにドリルするために使用されるパスです。

宛先 1) Business Rules では、計算済の値が格納されるデータベースのブロックです。2) Profitability and Cost Management では、割当てモデルでのソースと宛先の関連付けです。割り当てられたコストや収益のフローの方向を制御します。

実績の頻度 日付のセットを作成して結果を収集および表示するために使用されるアルゴリズムです。

実行時プロンプト ビジネス・ルールが実行される前にユーザーが入力または選択する変数です。

密次元 ブロック・ストレージ・データベースでは、次元メンバーのすべての組合せについてデータを含んでいる可能性があります。たとえば、時間次元はしばしば密ですが、これは時間次元がすべてのメンバーのあらゆる組合せを含んでいる可能性があるからです。「疎次元」と対比してください。

対称トポロジ 運用サイトとスタンバイ・サイトの層全体で同一である Oracle Fusion Middleware 障害回復構成です。対称トポロジでは、運用サイトとスタンバイ・サイトにあるホスト、ロード・バランサ、インスタンス、アプリケーションの数は同じです。両方のサイトで同じポートが使用されます。システムは同一に構成され、アプリケーションは同じデータにアクセスします。

対称型マルチプロセッシング(SMP) マルチプロセッシングとマルチスレッディングを使用可能にするサーバーのアーキテクチャです。多数のユーザーが単一のインスタンスに同時に接続した場合でも、パフォーマンスが大きく低下することはありません。

導出テキスト・メジャー Essbase Studio で、範囲として表現された定義済ルールによって管理される値を持つテキスト・メジャーです。たとえば、販売高メジャーに基づく導出テキスト・メジャー「販売実績インデックス」は、「高」、「中」、および「低」の値で構成できます。対応する売上が該当する範囲に応じて High、Medium、および Low が表示されるように定義されます。「テキスト・メジャー」も参照してください。

属性 次元メンバーの特性です。たとえば、従業員次元メンバーには、名前、年齢、または住所の各属性がある場合があります。Product 次元のメンバーはサイズ、味などの複数の属性を持つ可能性があります。

属性の関連付け データベース・アウトラインでの関係です。これにより、属性次元のメンバーが基本次元のメンバーの特性を表します。たとえば、製品 100-10 がグレープ味である場合、製品 100-10 は Flavor 属性の関連付けがグレープになります。したがって、Product 次元のメンバー 100-10 は Flavor 属性次元のメンバー Grape に関連付けられることになります。

属性タイプ 様々な関数(データのグループ化、選択、または計算)を使用可能にするためのテキスト、数値、ブール値、日付、またはリンク属性タイプです。たとえば、Ounces 属性次元は数値タイプを持つので、各製品の属性として指定されるオンス数を使用して当該製品のオンス当たりの収益を計算できます。

属性レポート 基本次元メンバーの属性に基づくレポート作成プロセスです。「基本次元」も参照してください。

属性次元 次元の一種です。次元のメンバーの属性や特質に基づいて分析できます。

属性計算次元 メンバーのグループに対して、合計、カウント、平均、最小、および最大を計算するシステム定義の次元です。この次元は動的に計算され、データベース・アウトラインでは表示されません。たとえば、メンバー Avg を使用すると、製品 Red についてニューヨークでの 1 月の平均売上の値を計算できます。

履歴平均 多数の履歴期間にわたる勘定科目の平均です。

差異 プラン値と実績値などの 2 つの値の差です。

市場リスク割増額 国債よりもリスクの高い投資を投資家に呼びかけるための、安全率に追加して支払われる利回りです。予測される市場利回りから安全率を差し引いて計算されます。この数字が示すモデルは将来の市況に近いものとなる必要があります。

式 Data Relationship Management で、プロパティ値を動的に計算するために派生プロパティによって使用されるビジネス・ロジックです。

式の保存 データ取得中にワークシート内に保持される、ユーザーが作成した式です。

役割 リソースへのアクセス権をユーザーおよびグループに付与する際に使用される手段です。

従業員 特定のビジネス・オブジェクトに対して責任を負う(または関与する)ユーザーです。従業員は組織に勤めている必要はありません(コンサルタントなど)。従業員は、認可のためにユーザー・アカウントに関連付けられている必要があります。

復元 データベースが破損または破壊された場合にデータおよび構造の情報を再ロードする操作です。通常、データベースをシャット・ダウンおよび再起動した後で実行されます。

感嘆符(!) 一連のレポート・コマンドを終了して、データベースからの情報を要求する文字です。レポート・スクリプトは感嘆符を使用して終了する必要があります。レポート・スクリプト内では複数の感嘆符を使用できます。

戦略マップ 上位レベルのミッションおよびビジョンのステートメントを、構成要素である下位レベルの戦略的達成目標に組入れる方法を示します。

戦略目標(SO) 測定可能な結果によって定義された長期目標です。各戦略目標は、アプリケーション内の 1 つのパースペクティブに関連付けられ、1 つの親(エンティティ)を持ち、重要成功要因または他の戦略目的の親になります。

手動ステージ ユーザーの操作が必要なステージです。

抽出コマンド Essbase レポート作成コマンドの 1 つです。データベースから抽出される RAW データの選択、向き、グループ分け、および配列を処理します。小なり記号(<)から始まるコマンドです。

拡張リレーショナル・アクセス リレーショナル・データベースと Essbase 多次元データベースの統合を指します。これにより、すべてのデータがリレーショナル・データベースに保持され、Essbase データベース内の要約レベルのデータにマッピングされます。

持株会社 法的エンティティ・グループの一部であるエンティティです。グループ内のすべてのエンティティに対して直接的または間接的に投資しています。

持続性 Essbase の操作や設定に対する継続的または長期的な影響です。たとえば、ユーザー名やパスワードの有効性について、Essbase 管理者がその持続性を制限することがあります。

接続ファイル「Interactive Reporting 接続ファイル (.oce)」を参照してください。

換算「通貨換算」を参照してください。

換算レート「為替レート」を参照してください。

支出勘定科目 期別の値と年次累計値を格納する勘定科目です。値が正の場合は、純利益が減ります。

支配比率 所属するグループのコンテキスト内でエンティティが受ける支配の程度です。

数値属性範囲 基本次元メンバーを関連付けるために使用される機能です。メンバーは個別の数値を含み、値の範囲を示す属性を持ちます。たとえば、顧客を年齢別に分類する場合、Age Group 属性次元に 0-20、21-40、41-60、および 61-80 という年齢範囲に該当するメンバーを含めることができます。各 Customer 次元メンバーは Age Group 範囲に関連付けられます。データを取得する際は、個別の年齢の値ではなく年齢範囲に基づいて処理されます。

日付メジャー Essbase で、メジャーが示されている次元で「日付」のタグが付けられているメンバーです。セルの値はフォーマット済の日付として表示されます。メジャーとしての日付は時間次元を使用して示すことが困難なタイプの分析に役立つことがあります。たとえば、一連の固定資産の取得日をアプリケーションで追跡する必要がある場合、取得日の範囲が実現可能な時間次元モデリングの範囲を超えて長期にわたってしまうことがあります。「型付きメジャー」も参照してください。

時系列レポート カレンダの日付(年、四半期、月、週など)に基づくデータのレポート作成プロセスです。

時間次元 データが示す期間です(会計期間、暦時間など)。

暗黙の共有 メンバーが 1 つ以上の子を持ち、連結されている子は 1 つのみである場合、親と子が値を共有します。

最上位メンバー 次元のアウトラインで、階層ツリーの一番上に位置する次元メンバーです。次元メンバー間に階層の関係がない場合は、ソート順で最初のメンバーを指します。階層の関係がある場合、最上位メンバーの名前が次元名と同一となるのが一般的です。

最新 最新の期間として定義されたメンバーからデータ値を抽出するために使用される、スプレッドシートのキー・ワードです。

期別価額メソッド(PVA) 通貨換算プロセスの 1 つです。一定期間における期別の為替レート値を適用して通貨を算出します。

期末 チャートの日付範囲を調整できる期間です。たとえば、月の期末の場合、当月末までの情報がチャートに表示されます。

株式ベータ 株のリスクを指します。その株の収益と市場利益率の差異により測定され、ベータと呼ばれるインデックスで示されます。たとえば、市場利益率が 1%変動するのに伴って株の収益が通常 1.2%変動するのであれば、その株のベータ値は 1.2 です。

検証 アウトラインに対してビジネス・ルール、レポート・スクリプトまたはパーティション定義をチェックして、チェック対象のオブジェクトが有効であることを確認するプロセスです。

検証ルール データの整合性を強化するために FDM で使用されるルールです。たとえば、FDM では、検証ルールによって、FDM からターゲット・アプリケーションにデータがロードされた後に、特定の条件が満たされていることが保証されます。

構築方法 データベース・アウトラインを変更するために使用するメソッドの一種です。データ・ソース・ファイルのデータ・フォーマットに基づいて構築メソッドを選択します。

構造ビュー トピックをコンポーネントのデータ項目の単純なリストとして表示します。

標準仕訳テンプレート 各期間に共通する調整を転記するために使用する仕訳の機能です。たとえば、共通する勘定科目 ID、エンティティ ID、または金額を含む標準テンプレートを作成すると、これを多数の通常仕訳の基準として使用できます。

標準次元 属性次元以外の次元です。

権限 データまたは他のユーザーとグループを管理するために、ユーザーおよびグループに付与されるアクセス・レベルです。

欠落データ(#MISSING) ラベル付けされた場所のデータが存在しないか、値が含まれていないか、データが入力されていないかまたはロードされていないことを示すマーカーです。たとえば、勘定科目に当期ではなく過去または将来の期間のデータが含まれている場合は、欠落データが存在します。

次元 ビジネス・データを整理して値の抽出や保持のために使用されるデータ・カテゴリです。通常、次元には関連するメンバーをグループ化した階層が含まれます。たとえば、Year 次元は多くの場合四半期、月などの期間の各単位のメンバーが含まれます。

次元タイプ 定義済の機能を使用可能にする次元のプロパティです。時間のタグが付けられた次元は、定義済のカレンダ機能を持ちます。

次元タブ「ピボット」セクションで、行と列の間でデータのピボットを実行するためのタブです。

次元テーブル 1)特定のビジネス・プロセスに関する多数の属性を含むテーブルです。2)Essbase Integration Services では、Essbase の潜在的な次元を定義する 1 つ以上のリレーショナル・テーブルのための、OLAP モデルのコンテナを指します。

次元性 MaxL DML において、セットで示された次元およびその順序です。たとえば、{ (West, Feb), (East, Mar) }というセットの場合は、含まれている 2 つのタプルはいずれも次元(Region, Year)を反映しているため、同一の次元性であることになります。

次元構築 Essbase アウトラインに次元およびメンバーを追加するプロセスです。

次元構築のルール データ・ロードのルールに似た仕様です。Essbase でアウトラインを変更するために使用されます。変更は外部データ・ソース・ファイルのデータに基づきます。

次元間の無関係性 次元が他の次元と交差しない状況を指します。次元に含まれるデータは、交差しない次元からはアクセスできないため、交差しない次元はその次元とは無関係となります。

残高勘定科目 特定の時点に関連する符号なしの値を格納する勘定科目の種別です。

水平アプリケーション・サーバー・クラスタ 複数のアプリケーション・サーバー・インスタンスがそれぞれ異なるマシン上にあるクラスタです。

汎用ジョブ SQR Production Reporting または Interactive Reporting 以外のジョブを指します。

消去 組織内のエンティティ間での取引をゼロに設定(消去)するプロセスです。

消去済勘定科目 連結ファイルに表示されない勘定科目です。

為替レート・タイプ 為替レートの識別子です。異なるレートのタイプが使用されるのは、一定期間および年間について複数のレートが存在することがあるためです。従来より、期末時点でのレートを当期の平均レートおよび期末レートとして定義します。その他、履歴レート、予算レート、予測レートなどのレート・タイプがあります。レート・タイプは特定の時間に適用されます。

現地通貨 入力通貨タイプです。入力通貨タイプが指定されていない場合は、現地通貨がエンティティの基本通貨に一致します。

疎次元 ブロック・ストレージ・データベースで、他の次元と比較した場合に、すべてのメンバーの組合せのデータを含んでいる可能性の低い次元です。「密次元」と対比してください。たとえば、すべての顧客がすべての製品のデータを持っているわけではありません。

目標 指定された期間(日、四半期など)についてメジャーに期待される結果です。

直接レート 為替レート・テーブルに入力する通貨レートです。直接レートは通貨換算に使用されます。たとえば、残高を日本円から米ドルに変換する場合、変換元通貨を日本円、変換先通貨を米ドルとして、為替レート・テーブルに期間またはシナリオのレートを入力します。

相互割当て 財務フローの割当ての一種です。宛先の 1 つとしてソースが含まれます。

相関サブクエリー 親クエリーの各行で一度評価されるサブクエリーです。サブクエリーのトピック・アイテムを親クエリーのトピックに結合することにより作成されます。

確認レベル プロセス管理の確認ステータス・インディケータの 1 つです。「開始していません」、「第 1 パス」、「送信済」、「承認済」、「発行済」など、プロセス単位のレベルを示します。

祖先 その下にメンバーを含む分岐メンバーです。たとえば、メンバー Qtr2 とメンバー 2006 はメンバー April の祖先です。

移行スナップショット アプリケーションの移行のスナップショットです。移行ログに取込まれます。

移行ログ アプリケーションの移行のすべてのアクションとメッセージを取込むログ・ファイルです。

移行定義ファイル(.mdf) アプリケーションの移行に使用される移行パラメータを含むファイルです。これによりバッチ・スクリプトを処理できます。

移行監査レポート 移行ログから生成されるレポートです。アプリケーションの移行に関する追跡情報を提供します。

税金の初期残高 Strategic Finance では、損失の初期残高、収益の初期残高、および納税の初期残高のエントリは、Strategic Finance の最初の期間に先立つ期間に発生していることを前提とします。

算出ステータス 一部の値または式の計算が変更されたことを示す集計ステータスです。影響を受けるエンティティについて正しい値を取得するには、再集計する必要があります。

算術データ・ロード データベース内の値に対して演算(たとえば各値に 10 を加算するなど)を実行するデータ・ロードです。

算術演算子 式およびアウトラインでのデータの計算方法を定義する記号です。標準的な算術演算子またはブール演算子が使用されます(+、-、*、/、%など)。

管理対象サーバー 内蔵された Java 仮想マシン(Java Virtual Machine: JVM)で実行されるアプリケーション・サーバー・プロセスです。

精度 数値に表示される小数点以下の桁数です。

系列 異なるメタデータ要素間の関係です。メタデータ要素が他のメタデータ要素からどのように導き出されるかを示し、メタデータ要素を物理ソースまでトレースします。Essbase Studio では、この関係を系列ビューアでグラフィカルに表示できます。「トレーサビリティ」も参照してください。

系統データ 割当ての計算後にオプションで生成される追加データです。このデータにより、すべての割当てステップにわたるコストまたは収益のフローについてレポートを作成できます。

組織 各エンティティ、およびその関係を定義するエンティティの階層です。

結合 特定の列または行の共通のコンテンツに基づく 2 つのリレーショナル・データベース・テーブルまたはトピックの間のリンクです。通常、異なるテーブルまたはトピック内の同一または類似するアイテムの間で結合が起きます。たとえば、Customer テーブルと Orders テーブルで Customer ID の値が同一である場合、Customer テーブル内のレコードが Orders 内のレコードに結合します。

統制グループ 証明書および評価の情報を維持および整理するために FDM で使用されるグループです。サーベンス・オクスリ(Sarbanes-Oxley)法の規定に準拠する上で特に役立ちます。

統合 Shared Services を使用して Oracle Hyperion アプリケーションでデータを移動するために実行されるプロセスです。データ統合の定義によりソース・アプリケーションと宛先アプリケーションの間でのデータの移動が指定され、データの動きのグループ化、順序付けおよびスケジュールが決定されます。

繰返し 同じバージョンのデータを修正して移行する予算またはプランニング・サイクルのパスです。

置換 データ・ロードのオプションの 1 つです。データ・ロード・ファイルに指定された期間のすべての勘定科目からの既存の値を消去し、データ・ロード・ファイルからの値をロードします。ロード・ファイルに勘定科目が指定されていない場合、指定された期間に該当する値が消去されます。

耳折れ 折り曲げられたページの角です。チャートのヘッダー領域の右上の隅に表示されます。

自動ステージ ユーザーの操作を必要としないステージです(データ・ロードなど)。

自動逆仕訳 次期に逆仕訳する調整を入力するための仕訳です。

行の抑制 欠落値を含む行を除外し、スプレッドシート・レポートからの文字にアンダースコアを付ける設定です。

表示タイプ リポジトリに保存された 3 種類の Web Analysis フォーマット(スプレッドシート、チャート、ピンボード)のいずれかを指します。

製品 Shared Services における、Planning や Performance Scorecard などのアプリケーション・タイプです。

複製パーティション パーティション・マネージャにより定義されるデータベースの一部。あるサイトで管理されるデータの更新を別のサイトに伝播するために使用されます。ユーザーは、ローカルのデータベースと同じようにデータにアクセスできます。

要約チャート「調査」セクションで、同じ列内で下に表示される詳細チャートをロール・アップするチャートです。各チャート列最上位の要約レベルにメトリックを描画します。

親 直接レポートする依存エンティティを 1 つ以上含むエンティティです。親は少なくとも 1 つのノードに関連付けられたエンティティであるため、エンティティ、ノード、および親の情報が関連付けられています。

親の調整 親に関連して子に転記される仕訳エントリです。

計算スクリプト データベースの集計方法や集約方法を定義する一連のコマンドです。集計プロセスとは別に、割当てや他の計算ルールを指定するコマンドが計算スクリプトに含まれることもあります。

計算済勘定科目 計算式を変更することができない勘定科目です。計算式は、構築するモデルの勘定科目の整合性を保つために固定されています。たとえば、当期純利益、計算済勘定科目の計算式は戦略的財務に組み込まれており、履歴期間または予測期間で変更はできません。

計算結果アイテム データベースやキューブに物理的に格納される列に対して、仮想の列を指します。クエリー実行時にデータベースにより、または Interactive Reporting Studio の「結果」セクションで計算されます。計算結果アイテムは、関数、データ項目、およびダイアログ・ボックスで提供される演算子に基づくデータ計算であり、レポートに含まれたり他のデータの計算に再利用されることがあります。

設計レポート コンポーネント・ライブラリを使用してカスタム・レポートを作成するための Web Analysis Studio のインタフェースです。

詳細チャート 要約チャートで、詳細な情報を提供するチャートです。詳細チャートは要約チャートの下にある「調査」セクションに列で表示されます。要約チャートに円グラフが表示される場合、その下の詳細チャートには円の各区分が示されます。

認証 安全対策としての ID の確認です。一般に、認証はユーザー名およびパスワードに基づきます。パスワードおよびデジタル・シグネチャは認証のフォームです。

認証サービス 単一の認証システムを管理するコア・サービスです。

調整「仕訳」を参照してください。

調整勘定科目 会社間勘定科目の消去プロセスで均衡しない差額が格納される勘定科目です。

調査「ドリルスルー」を参照してください。

論理 Web アプリケーション Web アプリケーションの内部ホスト名、ポートおよびコンテキストの識別に使用される、別名が付けられた参照です。クラスタ環境または高可用性環境では、分散コンポーネントに対する単一の内部参照を確立する別名です。EPM System では、クラスタ化されていない論理 Web アプリケーションは Web アプリケーションを実行する物理ホストにデフォルトで設定されています。

論理グループ FDM で、ソース・ファイルが FDM にロードされた後に生成される 1 つ以上の論理勘定です。論理勘定はソース・データから導き出される計算済勘定です。

負債勘定科目 一定時点における会社の負債残高を格納する勘定科目タイプです(未払費用、買掛金勘定、長期借入金など)。

責任マップ 組織内の責任チーム(重要事業領域とも呼ばれます)の責任、報告、および依存関係の構造を視覚的、階層的に示します。

貸借一致の仕訳 借方の合計と貸方の合計が等しい仕訳です。

資産勘定科目 勘定科目タイプの 1 つです。会社の資産の値を保管します。

軸 1)測定と分類に使用されるグラフィックを貫通する直線です。2)多次元のデータを整理および関連付けるために使用されるレポートのアスペクトです(フィルタ、ページ、行、列など)。たとえば、Simple Basic でデータ・クエリーを実行する場合、軸では Qtr1、Qtr2、Qtr3、および Qtr4 の値の列を定義できます。Market と Product の階層による合計が行データとして取得されます。

透過パーティション ローカルのデータベースの一部であるかのように、リモート・データベースのデータにアクセスして変更できるようにする共有パーティションです。

透過ログイン ログイン画面を起動せずに認証されたユーザーをログインさせるプロセスです。

通常仕訳 特定の期間に一度かぎりの調整を入力するための機能です。通常仕訳は、貸借一致、エンティティごとに貸借一致、貸借不一致のいずれかになります。

通貨の上書き 任意の入力期間について選択した入力メソッドを上書きして、デフォルトの通貨/アイテムとして該当期間の値の入力を使用可能にする機能です。入力メソッドを上書きするには、数値の前または後にシャープ(#)を入力します。

通貨パーティション アプリケーションでの定義に従って、基本通貨から現地通貨メンバーを隔離する次元タイプです。通貨タイプ(実績、予算、予測など)を特定します。

通貨換算 データベースの通貨の値を別の通貨に変換するプロセスです。たとえば、1 米ドルをユーロに変換するには、ドルに為替レート(たとえば、0.923702)を乗じます(1*0.923702)。変換後のユーロの額は 0.92 になります。

連結 従属するエンティティからのデータを親エンティティに集約するプロセスです。たとえば、次元 Year に Qtr1、Qtr2、Qtr3、および Qtr4 というメンバーが含まれている場合、この連結は Year になります。

連結ファイル(*.cns) 集計のプロセスでチャートまたはツリー・ビューを使用して Strategic Finance ファイルを追加、削除または移動するためのグラフィカル・インタフェースです。集計ファイルを使用して、集計を定義したり変更したりすることも可能です。

連結ファイル(親) 事業部門のすべてのファイルが連結されたファイルです。連結の定義を含みます。

連結比率 親に連結された子の値の割合です。

適応状態 Interactive Reporting Web Client の権限レベルです。

選択リスト レポート設計者がレポートの視点(POV)を定義する際に各次元に指定するメンバーのリストです。定義されたメンバー・リストに指定されたメンバーを選択するか、または動的リストの関数に定義された条件に一致するメンバーを選択するだけで、選択リストを使用する次元の POV を変更できます。

重複する別名 別名テーブルに複数存在し、データベース・アウトラインの複数メンバーに関連付けられている可能性のある名前です。重複する別名は、重複メンバーのアウトラインでのみ使用できます。

重複メンバーのアウトライン 重複メンバー名を格納しているデータベース・アウトラインです。

重複メンバー名 データベース内に複数存在する、それぞれ異なるメンバーを表す同一のメンバー名です。たとえば、ニューヨーク州を示すメンバーとニューヨーク市を示すメンバーが存在する場合、データベースに New York という名前のメンバーが 2 つ含まれることがあります。データベースに New York という名前のメンバーが 2 つ含まれることがあります。

重要事業領域(CBA) 部門、地域、工場、コスト・センター、プロフィット・センター、プロジェクト・チーム、またはプロセスに編成された個人またはグループです。責任チームまたはビジネス領域とも呼ばれます。

重要成功要因(CSF) 戦略目標を達成するために確立および維持する必要のある能力です。戦略目標または重要プロセスにより所有され、1 つ以上のアクションに対する親となります。

関数 Data Relationship Management で、パラメータを受け入れ動的値を返す、派生プロパティ式の構文要素です。

関連勘定科目 メイン勘定科目に関連する、同一のメイン勘定科目番号でグループ化された勘定科目です。勘定科目の構造体では、すべてのメイン勘定科目および関連勘定科目は同一のメイン勘定科目番号にグループ化されます。メイン勘定科目と関連勘定科目は、勘定科目番号の最初の接尾辞により区別されます。

限界税率 税引き後の負債コストを計算するために使用されるレートです。最近計上された所得に適用される税率(所得額に適用される最高の税率区分の税率)を示し、連邦税、州税、および地方税を含みます。課税対象所得と税率区分の現在のレベルに基づいて、限界税率を予測できます。

障害回復 地理的に離れたスタンバイ・サイトへのアプリケーションおよびデータに対する回復戦略を備えることで、運用サイトでの自然災害や予定外の停止から守る機能です。

隣接する四角形 Interactive Reporting ドキュメントのセクションを個人用ページに埋め込む場合に、Interactive Reporting ドキュメントのコンテンツをカプセル化する必須のパラメータです。高さと幅を表すピクセル、または 1 ページ当たりの行数により指定されます。

集約 集約ストレージ・データベースの値をロール・アップおよび格納するプロセスです。または集約プロセスによって格納された結果を指します。

集約スクリプト 集約を構築するための集約ビューの選択を定義するファイルです。集約ストレージ・データベースのみで使用されます。

集約ストレージ・データベース 潜在的に大きな多数の次元に分類される疎に分散した大規模なデータをサポートするように設計されたデータベースのストレージ・モデルです。上位のメンバーと式は動的に計算され、選択されたデータ値は集約、格納されます。通常、集約の合計所要時間が改善されます。

集約セル 複数のセルから構成されるセルです。たとえば、Children(Year)を使用するデータ・セルは、Quarter 1、Quarter 2、Quarter 3、および Quarter 4 のデータを含む 4 つのセルに展開されます。

集約ビュー 各次元内のメンバーのレベルに基づく集約セルの集合です。計算時間を短縮するため、値は事前に集約されて集約ビューとして格納されています。取得は集約ビューの合計から開始され、合計に追加されます。

集約制約 集約要求ライン・アイテムや集約メタトピック・アイテムに設定する制約です。

集約関数 関数の一種です。合計、平均の計算など、データの要約や分析を実行します。

集計ルール 階層のノードを集計する際に実行されるルールです。親の残高が正しく集計されるように、顧客固有の適切な式を含めることができます。消去プロセスは、このルール内で制御できます。

非アクティブ・グループ 管理者によりシステムへのアクセスが非アクティブにされているグループです。

非アクティブ・ユーザー 管理者によりアカウントが非アクティブ化されているユーザーです。

非対称トポロジ 運用サイトとスタンバイ・サイトの層全体で異なる Oracle Fusion Middleware 障害回復構成です。たとえば、非対称トポロジでは、運用サイトよりも少ないホストとインスタンスがあるスタンバイ・サイトが含まれることがあります。

非次元モデル Shared Services のモデル・タイプの 1 つです。セキュリティ・ファイル、メンバー・リスト、計算スクリプト、Web フォームなどのアプリケーション・オブジェクトが含まれます。

領域 メンバーおよび値の定義済のセットであり、パーティションを構成します。

高可用性 障害が発生した場合でもアプリケーションが継続してサービスを提供できるようにするシステム属性です。これは、シングル・ポイント障害の除去、フォルト・トレラント・ハードウェアおよびサーバー・クラスタにより実現されます。1 つのサーバーで障害が発生すると、処理要求は別のサーバーにルーティングされます。

高機能計算 最後に実行された計算以降に更新されたデータ・ブロックを追跡する計算方法です。

索引

記号

A - Z

あ行

か行

さ行

た行

な行

は行

ま行

や行

ら行

わ行