IBM® DB2® OLAP Server™ 8.1

Database Administrator's Guide

Volume I: Essbase Basics

SC18-7001-00

Printed in the U.S.A.

Contents

Volume I: Essbase Basics

Chapter 10: Creating and Managing Aliases 261

Chapter 11: Linking Objects to Essbase Data 273

Volume II: Loading, Calculating, and Retrieving Data

Chapter 22: Manipulating Fields Using a Rules File 613

Chapter 23: Performing and Debugging a Data Load 641

Volume III: Maintaining, Automating, and Optimizing Essbase

Preface

Welcome to the *Essbase Database Administrator's Guide*:

Purpose

This guide provides you with strategic and detailed technical information:

- A presentation of the strategies and techniques necessary to implement, design, and maintain an optimized Essbase multidimensional database
- A technical discussion of Essbase concepts to help you think about and manage data in a multidimensional environment and to enable you to design a Hyperion Essbase database on your own
- Step-by-step procedures on how to use Application Manager

Use this guide to complete any of the following tasks:

- Learn about Essbase architecture, philosophy, and concepts
- Design a multidimensional application
- Define users and security

- Perform data loads and outline updates
- Learn techniques for developing calc and report scripts for advanced applications
- Ensure data integrity
- Optimize a database
- Backup and restore data
- Manage large databases

Audience

This guide is primarily for the following:

- Essbase database administrators. An Essbase database administrator is a person who installs, controls, and maintains an Essbase system.
- People who design, create and maintain applications, databases, data load rules, calculation scripts, and report scripts.

Prerequisites

Before you use this guide to create or maintain Essbase databases, you should already know these items:

- The operating system your server uses and of the operating systems your clients use. For information on the supported operating systems, see the *Essbase Installation Guide*.
- The location of the data that your business uses and who is responsible for updating Essbase with current data.
- Your business's data requirements, so you can apply Essbase to your specific application.
- Typical database administration requirements and tasks, including designing security, setting up user accounts, how many users your server can accommodate and how to manage the space on your server.

Document Map

This guide is divided into volumes and parts that describe the major functional areas of Hyperion Essbase:

- Volume I: Essbase Basics
  - Part I, “Understanding Hyperion Essbase”
  - Part II, “Designing and Creating Applications and Databases”
  - Part III, “Designing, Building, and Maintaining Partitions”
  - Part IV, “Designing and Building a Security System”
- Volume II: Loading, Calculating, and Retrieving Data
  - Part V, “Building Dimensions and Loading Data”
  - Part VI, “Calculating Data”
  - Part VII, “Retrieving Data”
- Volume III: Maintaining, Automating, and Optimizing Essbase
  - Part VIII, “Storing and Protecting Data”
  - Part IX, “Maintaining and Automating Hyperion Essbase”
  - Part X, “Optimizing and Troubleshooting Essbase”
  - Appendix A, “Limits”
  - Glossary
  - Index

To find any information not listed above, use the Information Map located in the *`ARBORPATH`*`/docs` directory of your Essbase installation.

Accessing Hyperion Documentation

All Essbase documentation is accessible from the following locations:

- The Information Map is located in the *ARBORPATH*/`docs` directory of your Essbase installation.
- The DB2 OLAP Server site is located at: `www.ibm.com/software/db2/db2olap/library.html`.

Document Conventions

This book uses several formatting styles to indicate actions you should take or types of information you need. The following table lists each document convention.

Table i: Document Conventions

Type	Description
Essbase Installation Guide	Titles of books appear in italics.
parallel calculation	The earliest occurrence of a word from the glossary appears in italics.
n, *x*	You will often see *n* or *x* in formulas. *n* represents a number, and *x* a letter
ARBORPATH (environment variable) *ARBORPATH*/`docs`	You will often see the variable ARBORPATH, which is also an environment variable. When you see it in italics, substitute the value of ARBORPATH from your site.
File > Open	Choices from a menu item use the greater than symbol (>).
Time -> Qtr1	The dash and greater than symbol combined (->) indicate the Essbase cross-dimensional operator.

Table i: Document Conventions (Continued)

Type	Description
➤	Arrows indicate the beginning of a procedure.
Brackets []	In examples, brackets indicate that the enclosed elements are optional.
Bold	• Bold in procedural steps highlights major interface elements. • MaxL commands are presented in bold.
CLEARBLOCK	Capital letters denote commands, functions, and configuration settings, except MaxL commands, which are in lower case and bold.
Ctrl + 0	Keystroke combinations shown with the plus symbol (+) indicate that you should press the first key and hold it while you press the next key. Do not type the + symbol.
`essbase.cfg`	Courier font denotes the names of files, directories, and sample code or example code.
`Courier italics`	Courier italic text indicates a variable in command syntax. Substitute a value in place of the variable shown in Courier italics.
(...)	Ellipsis points indicate that text has been omitted from an example.

This document provides examples and procedures using a right-handed mouse. If you use a left-handed mouse, adjust the procedures accordingly.

Sample Applications

This book provides examples. The examples are based on applications that are provided with the Hyperion Essbase server software and are called Sample, Demo, Samppart, and Sampeast. (Samppart and Sampeast are examples of partitioned applications. These applications are available only if your company has licensed Partitioning.) The individual who installs the server is responsible for making the example applications available to end users:

- Sample contains three databases: Basic, Interntl, and Xchgrate
- Demo contains one database: Basic
- Samppart contains one database: Company
- Sampeast contains one database: East

If, when you connect to the Essbase server, any of the following problems occur, contact your Essbase administrator:

- You cannot find the Sample or Demo applications.
- Your company has licensed Partitioning, and you cannot find the Samppart or Sampeast applications.
- You do not have adequate access to the applications.
- You do not see any data in the databases.

For more information about the sample applications, see the *Essbase Installation Guide*.

Part

I

Understanding Hyperion Essbase

Part I introduces you to the Hyperion® Essbase® OLAP Server by describing general online analytical processing (OLAP) concepts, the steps that you should follow to create a Hyperion Essbase OLAP Server, basic multidimensional concepts, the Essbase architecture, and how to design both single server and partitioned applications. Part I contains the following chapters:

- Chapter 1, "Essbase and Multidimensional Databases," describes the parts of Essbase, high-level Essbase functionality, key architectural features, and how Essbase works in a client-server environment.
- Chapter 2, "Multidimensional Concepts," introduces you to basic multidimensional concepts and terminology, including dimensions and members, data values, and hierarchies. It contains several diagrams and a detailed description of a simple application.
- Chapter 3, "Basic Architectural Elements," describes how the Essbase architecture stores and retrieves information, introducing concepts such as data blocks, indexes, and dense and sparse dimensions.
- Chapter 4, "Quick Start for Implementing Essbase," provides a high-level process map for implementing Essbase in your organization with cross references to further information.

Chapter 1

Essbase and Multidimensional Databases

This chapter introduces Essbase and describes the Essbase environment. Essbase is a multidimensional database server that is optimized for planning, analysis, and management reporting applications. You can access Essbase from a spreadsheet or custom interface on a desktop computer or on a workstation. Managers, analysts, and executives can see useful information on demand with Essbase.

This chapter contains the following sections:

- "About the Essbase Product Family" on page 51
- "Multidimensional Development Features" on page 53
- "Architectural Features of Essbase" on page 53
- "Understanding Client-Server Communications" on page 55

About the Essbase Product Family

The Essbase product family includes the following feature sets:

- Essbase Administration Services

  Essbase Administration Services is the new framework for managing and maintaining Essbase. Essbase Administration Services consists of a client console and a middle tier server that communicate directly with Essbase OLAP Servers. Both of these components can run on any platform supported by Essbase. For more information, see the *Essbase Administration Services Online Help*.

- Essbase Application Manager

  The *Application Manager* is a graphical environment for developing and maintaining Essbase applications on a single OLAP Server. Tasks include building outlines and dimensions, performing data loads and calculations, and defining security access.

- Essbase OLAP Server

  A multidimensional database for storing data with an unlimited number of dimensions such as time, accounts, region, channel, or product. The Essbase server manages analytical data models, data storage, calculations, and data security.

- Essbase Spreadsheet Add-in

  The Spreadsheet Add-in desktop software enables analysis of the data stored in the Essbase server. Essbase Spreadsheet Add-in is seamlessly integrated with Microsoft Excel or Lotus 1-2-3 spreadsheets.

- Essbase application tools

  This suite of tools is used for extending Essbase applications. These tools include Essbase currency conversion, Essbase SQL Interface, Essbase Spreadsheet Toolkit, and Essbase Application Programming Interface.

- Essbase Partitioning

  This suite of features makes it easy to design and administer databases that span Essbase applications or servers. You can copy a slice of a large database to work with locally, or you can link from your database directly to other databases.

- Essbase Hybrid Analysis

  Hybrid Analysis integrates relational databases with Essbase databases to combine the size and scalability of the relational database with the conceptual power and analytic capabilities of the multidimensional database. Hybrid Analysis eliminates the need to store lower-level members and data within Essbase, virtually eliminating size restrictions on outlines and giving you rapid analysis of large amounts of data.

Multidimensional Development Features

Essbase offers many key advantages to help you develop effective multidimensional applications:

- Essbase requires minimal programming experience and no knowledge of query languages. You can design and manage applications using a graphical interface to control most server functions.
- You can add dimensions, change calculations, and modify hierarchies to reflect new business developments. The dynamic dimension builder automatically defines and dynamically loads large amounts of data. You can load spreadsheets, flat files, and SQL tables directly into the database.
- Essbase contains over 100 analytical functions, all of which operate on very large data sets. You can perform key analytical functions, such as trend analysis, ratios, and allocations, without having to write a program.You can define calculations quickly by using pre-defined functions for depreciation, variances, and other common formulas.
- Essbase provides secure data views and limits access to unauthorized areas. You can define security for individuals and groups and customize views and retrieval procedures for each user without writing a program.
- Essbase is easy to deploy and supports standard applications, operating systems, and networking protocols.

Architectural Features of Essbase

Essbase incorporates the following powerful architectural features to handle a wide range of analytic applications across large multi-user environments.

Dynamic Dimensionality

The Essbase server uses a method called dynamic dimensionality for storing and retrieving data and for optimizing analytical performance. This method separates data into sparse and dense dimensions. See Chapter 2, “Multidimensional Concepts” and Chapter 3, “Basic Architectural Elements” to learn how Essbase defines and uses sparse and dense dimensions to optimize data access and to reduce index and storage requirements within the database.

Dynamic dimensionality allows Essbase to provide sophisticated attribute reporting without impact to database storage requirements or batch calculation performance.

Multithreaded Design and SMP

The Essbase server is a 32-bit, multithreaded software application that supports symmetric multiprocessing (SMP) hardware platforms. Multithreaded design creates a separate thread for each user request. A multithreaded software architecture enables multiple users to work on an Essbase database at the same time. Essbase also uses separate threads to support data loads and calculations in the database.

Symmetric multiprocessing allows single servers to run multiple processors concurrently. Essbase automatically supports multiple threads over SMP servers. Thus performance is not significantly degraded, even with a large number of simultaneous users.

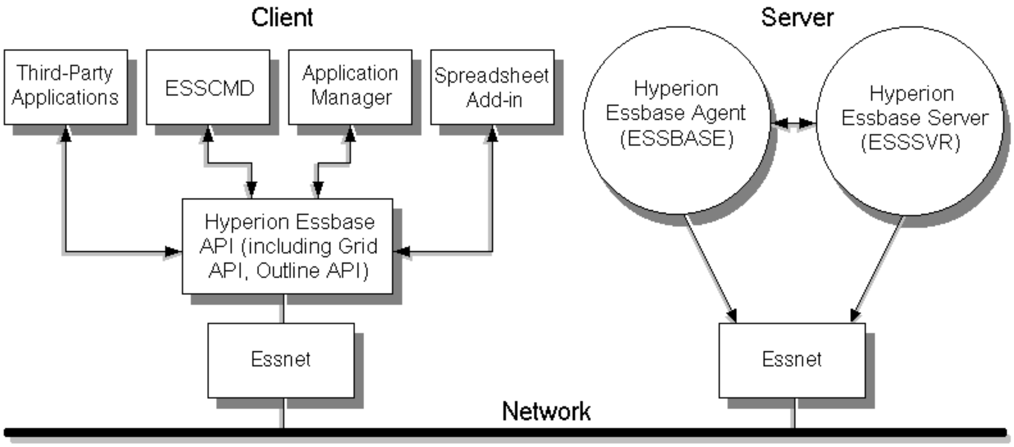
Multiple User Read and Write

The Essbase server supports simultaneous access and update by multiple users. You can implement applications that require iterative changes to data, such as budgeting, forecasting, and planning applications, and allow multiple users to access these applications simultaneously.

Understanding Client-Server Communications

The Essbase client-server architecture supports enterprise analysis applications:

Figure 1: Essbase Client-Server Communications


Client
Server
Third-Party Applications
ESSCMD
Application Manager
Spreadsheet Add-in
Hyperion Essbase Agent (ESSBASE)
Hyperion Essbase Server (ESSSVR)
Hyperion Essbase API (including Grid API, Outline API)
Essnet
Essnet
Network

Note: Essbase Administration Services, MaxL, and other components do not appear in this illustration.

Notice these relationships:

- The server runs a Server Agent (ESSBASE) process that acts as a traffic coordinator for all user requests to Essbase applications.
- The application server (ESSSVR) fields data requests from clients.
- An Essnet library connects Essbase to the network and enables the server and the clients to communicate with each other.

Essbase uses a distributed client-server model. In a distributed model, the database engine typically resides on the server and portions of the database software reside on each client. A typical client-server configuration has one server and multiple clients: the server performs most of the database processing so that the clients can run queries with minimal memory and disk configurations.

Essbase clients often connect to multiple servers to access different databases. Within your organization, you might have multiple servers, each with its own users and databases.

Essbase Server

All Essbase application components, including database outlines and calc scripts, application control, and multidimensional database information, reside on the server. With Essbase you can configure server disk storage to span multiple disk drives, enabling you to store large databases. Essbase requires a server to run a multithreaded operating system so the server can efficiently manage multiple, simultaneous requests. The server also runs a server agent process that acts as a traffic coordinator for all user requests to Essbase applications.

The Essbase server software runs on PCs or UNIX servers. See the *Essbase Installation Guide* for information on the supported operating systems, and for specific information about server configuration requirements.

Essbase Client

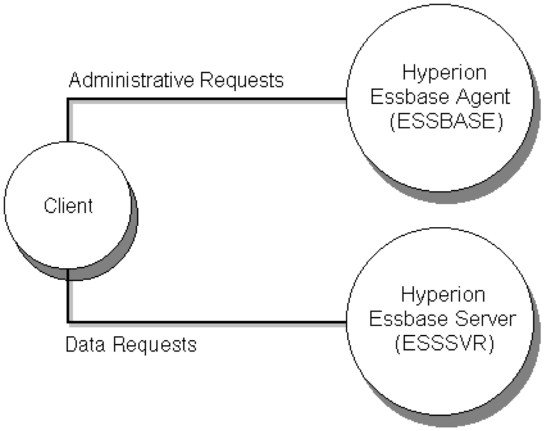
The following Essbase clients access the server:

- Essbase Spreadsheet Add-in, which provides you with seamless data access to the server.
- Essbase Application Manager, which you can use to design, develop, and maintain Essbase Applications.
- ESSCMD, a command-line interface, which you can use to perform operations interactively or through a batch or script file.
- A custom application built with the Essbase Application Programming Interface (API). The *API Reference* in your `docs` directory provides a complete listing of Essbase API functions, platforms, and supported compilers.

Essbase clients retrieve and analyze data from the server using Lotus 1-2-3, Microsoft Excel, or a custom application interface.

Essbase handles client requests differently, depending on the type of the request:

Figure 2: Essbase Client Request Handling


Administrative Requests
Hyperion Essbase Agent (ESSBASE)
Client
Hyperion Essbase Server (ESSSVR)
Data Requests

- Administrative requests, such as logging in and logging out, starting and stopping applications and databases, and viewing user security information, are handled by the Essbase Server Agent (ESSBASE).
- Client requests for data, such as data loads, calculations, spreadsheet reports, and data lock and unlock, are handled by the application server (ESSSVR).

See the *Essbase Installation Guide* for specific information on client configuration requirements and for information about supported platforms for Essbase products.

Chapter 2 Multidimensional Concepts

Essbase OLAP Server contains multidimensional databases that support analysis and management reporting applications that are described as online analytical processing (OLAP) applications. This chapter discusses multidimensional concepts and terminology. This chapter contains the following sections:

Introducing OLAP

In 1993, E. F. Codd, who set the seminal rules describing relational databases, published twelve rules for the analytical functions and performance characteristics that are essential to enterprise-scale planning and analysis applications. He called the new technology online analytical processing (OLAP) to reflect its analytical functionality and to differentiate it from online transaction processing (OLTP).

A *multidimensional database* supports multiple views of data sets for users who need to analyze the relationships between data categories. For example, a marketing analyst might want answers to the following questions:

- How did Product A sell last month? How does this figure compare to sales in the same month over the last five years? How did the product sell by branch, region, and territory?
- Did this product sell better in particular regions? Are there regional trends?

- Did customers return Product A last year? Were the returns due to product defects? Did the company manufacture the products in a specific plant?
- Did commissions and pricing affect how salespeople sold the product? Did particular salespeople do a better job of selling the product?

Multidimensional databases consolidate and calculate data to provide different views. Only the database outline, the structure that defines all elements of the database, limits the number of views. With a multidimensional database, users can *pivot* the data to see information from a different viewpoint, *drill down* to find more detailed information, or drill up to see an overview.

Codd’s twelve rules cover most user aspects of OLAP, including defining the conceptual view of the data (multidimensional), defining user needs (consistent reporting performance), and defining the platform (client-server). Codd also covers the kind of database operations a multidimensional database should support. These operations include the following:

- Unrestricted cross-dimensional operations
- Intuitive data manipulation
- Flexible reporting
- Unlimited dimensions and consolidation levels

Because of Codd’s research, the multidimensional database is a standard in today’s computing environment. In fact, OLTP and OLAP databases often coexist; many companies implement an OLAP database in tandem with an OLTP database.

Introducing Dimensions and Members

If you understand dimensions and members, you are well on your way to understanding the power of a multidimensional database.

Essbase has two types of dimensions: standard dimensions and attribute dimensions.

Standard dimensions represent the core components of a business plan and often relate to departmental functions. Typical standard dimensions are Time, Accounts, Product Line, Market, and Division. Dimensions are static in most databases; database dimensions rarely change over the life of the application.

Attribute dimensions are a special type of dimension and are associated with standard dimensions. Through attribute dimensions, you group and analyze members of your standard dimensions. Your analyses are based on the members' attributes (characteristics). For example, you can compare the profitability of your non-caffeinated products that are packaged in glass to the profitability of your non-caffeinated products that are packaged in cans.

Attribute dimensions must be associated with a base standard dimension. Essbase does not store the data for attribute dimensions, Essbase dynamically calculates the data when a user retrieves it. For more information about attribute dimensions, see Chapter 9, "Working with Attributes."

Members are the individual components of a dimension. For example, Product A, Product B, and Product C might be members of the Product dimension. Each member has a unique name. A dimension can contain an unlimited number of members. Essbase can store the data associated with a member (referred to as a stored member in this chapter) or it can dynamically calculate the data when a user retrieves it. For more information, see Chapter 28, "Dynamically Calculating Data Values."

A dimension represents the highest consolidation level in the database outline. The database outline indents members below one another to indicate a consolidation relationship. For example, in Figure 3, Time is a dimension and Qtr1 is a member. You will learn in later chapters how the hierarchy of members in the outline governs how users drill and pivot data.

Arranging Dimensions into Hierarchies

All Essbase database development begins with creating a database outline. A database *outline* accomplishes the following:

- Defines the structural relationships between members in a Essbase database
- Organizes all the data in the database
- Defines the consolidations and mathematical relationships between items

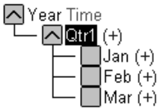
Essbase uses the concept of members to represent data hierarchies. Each dimension consists of one or more members. The members, in turn, may consist of other members. When you create a dimension, you tell Essbase how to *consolidate* the values of its individual members. Within the tree structure of the database outline, a consolidation is a group of members in a branch of the tree.

For example, many businesses summarize their data monthly, then roll up the monthly data to obtain quarterly figures, and roll up the quarterly data to obtain annual figures. Businesses may also summarize data by zip code, then by city, state, and country. Any dimension can be used to consolidate data for reporting purposes.

In the Sample Basic database included in your shipment, for example, the Year dimension consists of five members: the Qtr1, Qtr2, Qtr3, and Qtr4 members, each storing data for an individual quarter, plus Year, storing summary data for the entire year. Qtr1 consists of four members: the Jan, Feb, and Mar members, each storing data for an individual month, plus Qtr1, storing summary data for the entire quarter. Likewise, Qtr2, Qtr3, and Qtr4 consist of the members that represent the individual months plus the member that stores the quarterly totals.

The database outline in Figure 3 uses a hierarchical structure to represent the data consolidations and relationships in Qtr1.

Figure 3: Hierarchical Structure



```
Year Time
  Qtr1 (+)
    Jan (+)
    Feb (+)
    Mar (+)
```

Some dimensions consist of relatively few members, while others may have hundreds or even thousands of members. Essbase does not limit the number of members within a dimension and allows you to add new members as needed.

For information on creating a database outline, see Chapter 7, “Creating and Changing Database Outlines.”

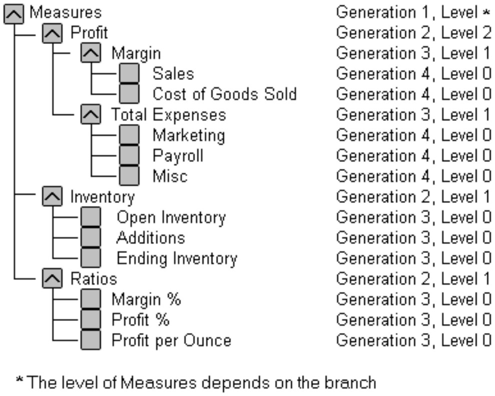
Defining Essbase Terminology

Essbase uses the terms defined in the following sections to describe a database outline. These terms are used throughout this manual.

Member Relationships, Generations, and Levels

Essbase uses hierarchical and family history terms to describe the roles and relationships of the members in an outline. You can describe the position of the members of the branches in Figure 4 in several ways.

Figure 4: Member Generation and Level Numbers



```
Measures                        Generation 1, Level *
    Profit                      Generation 2, Level 2
        Margin                  Generation 3, Level 1
            Sales               Generation 4, Level 0
            Cost of Goods Sold  Generation 4, Level 0
        Total Expenses          Generation 3, Level 1
            Marketing           Generation 4, Level 0
            Payroll             Generation 4, Level 0
            Misc                Generation 4, Level 0
    Inventory                   Generation 2, Level 1
        Open Inventory          Generation 3, Level 0
        Additions               Generation 3, Level 0
        Ending Inventory        Generation 3, Level 0
    Ratios                      Generation 2, Level 1
        Margin %                Generation 3, Level 0
        Profit %                Generation 3, Level 0
        Profit per Ounce        Generation 3, Level 0
```

* The level of Measures depends on the branch

Parents, Children, and Siblings

Figure 4 illustrates the following parent, child, and sibling relationships:

- A *parent* is a member that has a branch below it. For example, Margin is a parent member for Sales and Cost of Goods Sold.
- A *child* is a member that has a parent above it. For example, Sales and Cost of Goods Sold are children of the parent Margin.
- *Siblings* are child members of the same immediate parent, at the same generation. For example, Sales and Cost of Goods Sold are siblings (they both have the parent Margin), but Marketing (at the same branch level) is not a sibling because its parent is Total Expenses.

Descendants and Ancestors

Figure 4 illustrates the following descendant and ancestral relationships:

- *Descendants* are all the members in branches below a parent. For example, Profit, Inventory, and Ratios are descendants of Measures. The children of Profit, Inventory, and Ratios are also descendants of Measures.
- *Ancestors* are all the members in branches above a member. For example, Margin, Profit, and Measures are ancestors of Sales.

Roots and Leaves

Figure 4 illustrates the following root and leaf member relationships:

- The *root* is the top member in a branch. Measures is the root for Profit, Inventory, Ratios, and the children of Profit, Inventory, and Ratios.
- *Leaf* members have no children. They are also referred to as detail members, level 0 members, and leaf nodes. For example, Opening Inventory, Additions, and Ending Inventory are leaf members.

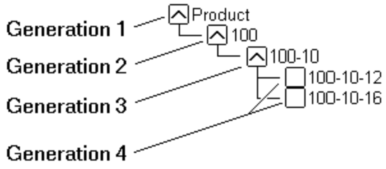
Generations and Levels

Figure 4 illustrates the following generations and branch levels:

- *Generation* numbers refer to consolidation levels within a dimension. A root branch of the tree is generation 1. Generation numbers increase as you count from the root toward the leaf member. In Figure 4, Measures is generation 1, Profit is generation 2, and Margin is generation 3. All siblings of each level belong to the same generation; for example, Inventory and Ratios are also generation 2.

  Figure 5 shows part of the Product dimension with its generations numbered.

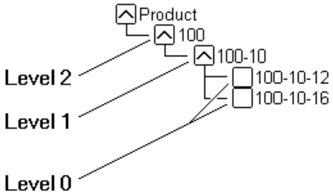
  Figure 5: Generations



- *Level* also refers to a branch within a dimension; however, levels reverse the numerical ordering that Essbase uses for generations. The levels count up from the leaf member toward the root. The root level number varies depending on the depth of the branch. In the example in Figure 4, Sales and Cost of Goods Sold are level 0. All other leaf members are also level 0. Margin is level 1 and Profit is level 2. Notice that the level number of Measures varies depending on the branch. For the Ratios branch, Measures is level 2. For the Total Expenses branch, Measures is level 3.

  Figure 6 shows part of the Product dimension with its levels numbered.

  Figure 6: Levels



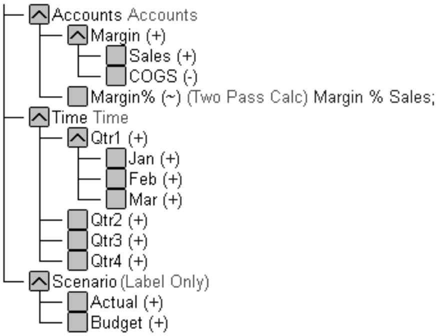
Note: You can assign a name to a generation or level and then use the name as a shorthand for all the members in that generation or level.

Identifying Values in a Multidimensional Database

This section describes how data is stored in a multidimensional database—a cube of cells containing data values. Each *data value* is stored in a single cell in the database. You refer to a particular data value by specifying its coordinates along *each* standard dimension.

Consider the simplified database shown in Figure 7.

Figure 7: A Multidimensional Database Outline



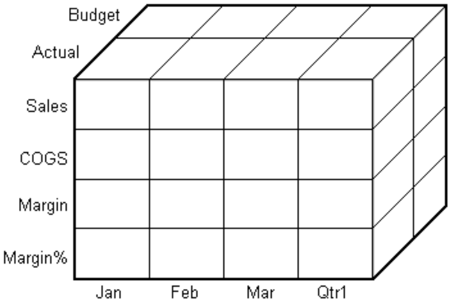
```
Accounts Accounts
    Margin (+)
        Sales (+)
        COGS (-)
    Margin% (~) (Two Pass Calc) Margin % Sales;
Time Time
    Qtr1 (+)
        Jan (+)
        Feb (+)
        Mar (+)
    Qtr2 (+)
    Qtr3 (+)
    Qtr4 (+)
Scenario (Label Only)
    Actual (+)
    Budget (+)
```

This database has three dimensions: Accounts, Time, and Scenario:

- The Accounts dimension has four members: Sales, COGS, Margin, and Margin%.
- The Time dimension shown in Figure 7 has four quarter members. Figure 8 shows only the members in Qtr1.
- The Scenario dimension has two child members: Budget for budget values and Actual for actual values.

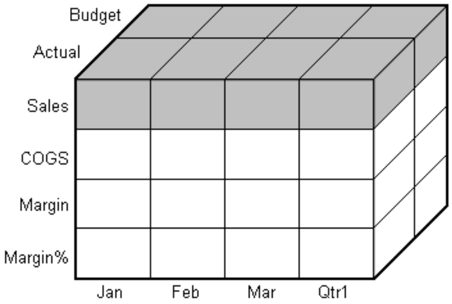
Each intersection of members (one member from each dimension) represents a data value. The example in Figure 8 has three dimensions; thus, the dimensions and data values in the database can be represented in a cube.

Figure 8: Three-Dimensional Database



The shaded cells in Figure 9 called a slice illustrate that, when you refer to Sales, you are referring to the portion of the database containing eight Sales values.

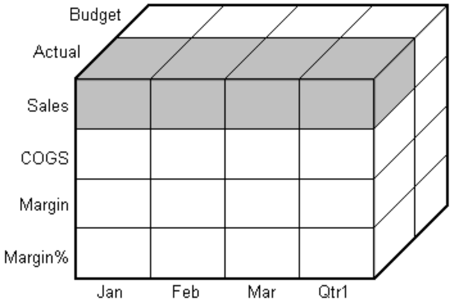
Figure 9: Sales Slice of the Database



Slicing a database amounts to fixing one or more dimensions at a constant value while allowing the other dimensions to vary.

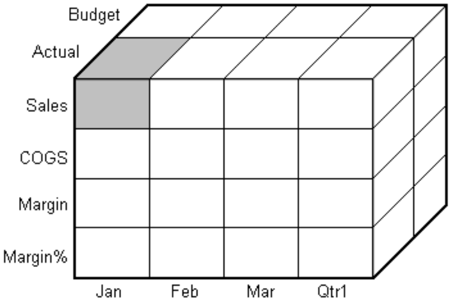
When you refer to Actual Sales, you are referring to the four Sales values where Actual and Sales intersect as shown by the shaded area in Figure 10.

Figure 10: Actual, Sales Slice of the Database



A data value is stored in a single cell in the database. To refer to a specific data value in a multidimensional database, you specify its member on each dimension. In Figure 11, the cell containing the data value for Sales, Jan, Actual is shaded. The data value can also be expressed using the cross-dimensional operator (->) as Sales -> Actual -> Jan.

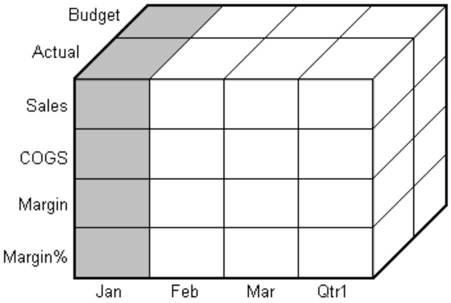
Figure 11: Sales -> Jan -> Actual Slice of the Database



Looking at Data from Different Perspectives

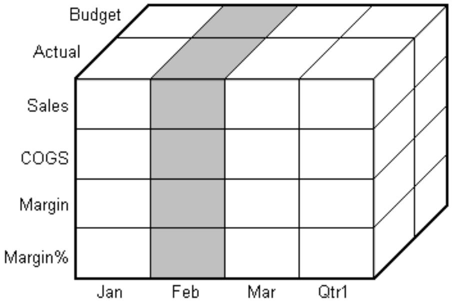
Slicing the database in different ways gives you different perspectives of the data. The slice of January in Figure 12, for example, examines all data values for which the Year dimension is fixed at Jan.

Figure 12: Data for January



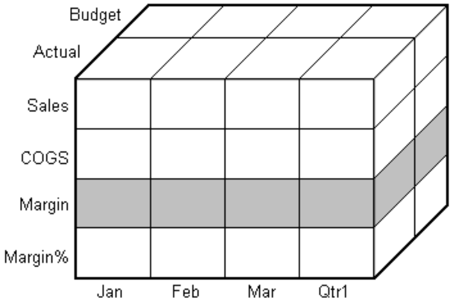
The slice in Figure 13 shows data for the month of February:

Figure 13: Data for February



The slice in Figure 14 shows data for profit margin:

Figure 14: Data for Profit Margin



Designing and Creating a Simple Application

Use this section to build and analyze a sample application called Simple. This application is for a company called The Car Company (TCC). TCC manufactures, markets, and distributes cars and trucks across the United States. This is not a sample application supplied with the software but is an example used to illustrate how you can quickly create your own first application.

These steps are similar to those you would use to create any Essbase application:

- "Analyzing the Application Requirements" on page 70
- "Organizing Multidimensional Data" on page 70
- "Adding and Deleting Standard Dimensions and Stored Members" on page 74

Analyzing the Application Requirements

Before you design an application, you must analyze the company data and determine requirements. Analysts at TCC prepare budget forecasts and track performance on a monthly basis.

Because TCC plans and tracks a variety of products over several markets, the process of deriving and analyzing data is quite tedious. Last month, analysts spent the majority of their time entering, rekeying, and preparing reports.

TCC needs a centralized repository for financial data that allows administrators to load data from different sources. The data repository should reside on a server accessible to analysts throughout the organization. Because all users have access to the server, they can retrieve data at will, regardless of its origin. To accommodate their needs, TCC chooses Essbase.

Organizing Multidimensional Data

To organize your data in a multidimensional structure, create a database outline. The outline defines the structure of the database, including the dimensions and members that it contains. It is important to remember that Essbase stores the database outline separately from the data in the database. Each time you make a significant change to the database outline, Essbase restructures the data to support the change. For more information on how to build an outline, see Chapter 7, "Creating and Changing Database Outlines."

This section contains information to help you create and understand the outline created for TCC:

- "Understanding the Outline" on page 70
- "Determining a Logical Structure" on page 71
- "Using the Logical Structure to Answer Questions" on page 72

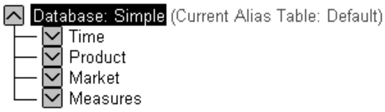
Understanding the Outline

A database outline contains standard dimensions, attribute dimensions, and members. The members can be stored or they can be dynamically calculated upon retrieval. The following example uses only standard dimensions and stored members. For more information about attribute dimensions and dynamically calculated members, see "Introducing Dimensions and Members" on page 60.

Determining a Logical Structure

To determine a logical structure for the data, remember that standard dimensions often parallel a company's organization. In Figure 15 the TCC Simple database has four standard dimensions: Time, Product, Market, and Measures.

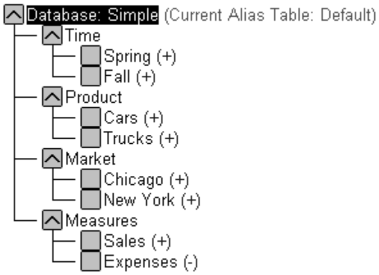
Figure 15: TCC Simple Database Showing Four Standard Dimensions



```
Database: Simple (Current Alias Table: Default)
├─ Time
├─ Product
├─ Market
└─ Measures
```

Give each dimension two members as shown in Figure 16.

Figure 16: TCC Simple Database with Standard Dimensions and Members



```
Database: Simple (Current Alias Table: Default)
├─ Time
│   ├─ Spring (+)
│   └─ Fall (+)
├─ Product
│   ├─ Cars (+)
│   └─ Trucks (+)
├─ Market
│   ├─ Chicago (+)
│   └─ New York (+)
└─ Measures
    ├─ Sales (+)
    └─ Expenses (-)
```

To appreciate the power of the multidimensional database, you need to understand how Essbase organizes members and standard dimensions. Consider a traditional spreadsheet, where a cell is at the intersection of a row and column. In a multidimensional database, a cell, or data value, is defined by the intersection of all the standard dimensions in the database.

For example, a spreadsheet cell is the intersection of a certain row and a certain column; for instance, row 3 and column 4. An Essbase data value is defined by the intersection of one member on each of the standard dimensions. For example, a data value can be at the intersection of Spring, Cars, Chicago, and Sales. See Chapter 3, "Basic Architectural Elements" for more information on how Essbase stores data values.

The database size is defined by its standard dimensions and members. For example, TCC is a four-dimensional database with three members in each dimension, including the root member (the member that is the name of the dimension). To determine the maximum number of values in the database, multiply the number of members in each dimension. For TCC, the maximum number of values is 3 x 3 x 3 x 3, so TCC has 81 potential data values. It is easy to see from this example how fast a multidimensional database can grow.

Using the Logical Structure to Answer Questions

Look at a subset of the 81 data values for the TCC database to determine how Essbase structures data. A typical query might be: *How many cars and trucks did TCC sell in the spring?* Begin with the member Spring, and list combinations of standard dimensions and members from the database outline for Spring. Remember, a data value must be defined by one member from each standard dimension. Table 1 contains all of the values that constitutes the Spring list, with the consolidated members in bold.

Table 1: Data Values and Consolidation Points for TCC

Time	Product	Market	Measures	Data Value
Spring	Cars	Chicago	Sales	800
Spring	Cars	Chicago	Expenses	600
Spring	Cars	Chicago	Measures	200
Spring	Cars	New York	Sales	500
Spring	Cars	New York	Expenses	200
Spring	Cars	New York	Measures	300
Spring	Cars	Market	Sales	1300
Spring	Cars	Market	Expenses	800
Spring	**Cars**	**Market**	**Measures**	**500**
Spring	Trucks	Chicago	Sales	700
Spring	Trucks	Chicago	Expenses	400
Spring	Trucks	Chicago	Measures	300
Spring	Trucks	New York	Sales	550
Spring	Trucks	New York	Expenses	150
Spring	Trucks	New York	Measures	400
Spring	Trucks	Market	Sales	1250

Table 1: Data Values and Consolidation Points for TCC (Continued)

Time	Product	Market	Measures	Data Value
Spring	Trucks	Market	Expenses	550
Spring	**Trucks**	**Market**	**Measures**	**700**
Spring	Product	Chicago	Sales	1500
Spring	Product	Chicago	Expenses	1000
Spring	Product	Chicago	Measures	500
Spring	Product	New York	Sales	1050
Spring	Product	New York	Expenses	350
Spring	Product	New York	Measures	700
Spring	Product	Market	Sales	2550
Spring	Product	Market	Expenses	1350
Spring	**Product**	**Market**	**Measures**	**1200**

Here are just a few of the questions you can answer using this data:

- How many cars did TCC sell in New York in the spring?
- Did TCC lose money on truck sales in New York in the spring?
- How many trucks did TCC sell in Chicago in the spring?
- What were the associated profits and revenues during that period?

A typical database contains associated formulas and a Essbase script to analyze the data. For more information on developing formulas and Essbase scripts, see Chapter 24, “Introduction to Database Calculations.”

Adding and Deleting Standard Dimensions and Stored Members

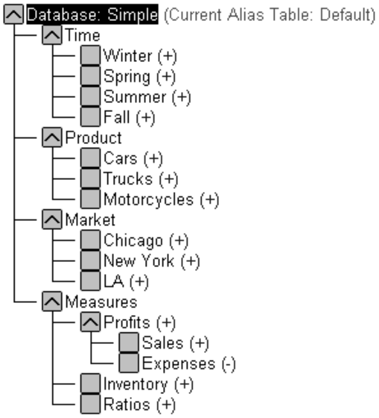
In this section, you will apply a few hypothetical situations to the TCC database. You will consider what happens to the multidimensional database when you add and delete members and standard dimensions:

- "Adding Stored Members" on page 74
- "Adding a Standard Dimension" on page 76
- "Removing a Standard Dimension" on page 77

Adding Stored Members

In this example, you will append additional stored members to the database under each standard dimension. Add two seasons under Time; one product, Motorcycles; one market, LA; and three members, Profits, Inventory, and Ratios, under the Measures dimension. Refer to Figure 17. For more information on how to build an outline, see Chapter 7, "Creating and Changing Database Outlines."

Figure 17: Adding Stored Members to the Outline



If you tried to create a table similar to Table 1, your table would look like the one shown in Table 2.

Table 2: Data Values and Consolidation Points of Stored Members

Time	Product	Market	Measures	Data Value
Winter	Cars	Chicago	Sales	1000
Winter	Cars	Chicago	Expenses	600
Winter	Cars	Chicago	Profits	
Winter	Cars	Chicago	Inventory	1600
and so on	...	...	...	...

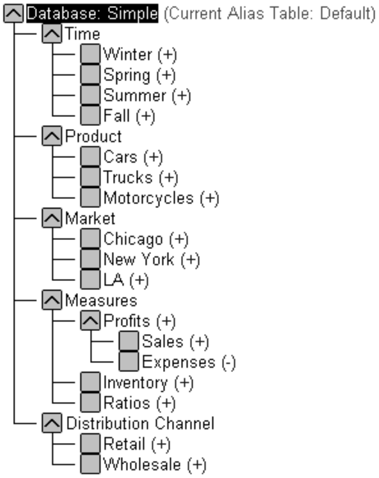
The new database is going to be much larger than the old one. There are now five members in the Time dimension, four members in each of the Product and Market dimensions, and six members in the Measures dimension.

To determine the maximum potential number of values in the database, multiply the number of stored members in each standard dimension: 5 x 4 x 4 x 6 = 480. So, by adding seven new members, the Simple multidimensional database has grown to a potential 480 values.

Adding a Standard Dimension

Add a new standard dimension called Distribution Channel to the database outline. Give the new dimension two stored members, Retail and Wholesale as shown in Figure 18. For more information, see "Adding Dimensions and Members to Outlines" on page 179.

Figure 18: Adding a Standard Dimension to the Outline



```
Database: Simple (Current Alias Table: Default)
    Time
        Winter (+)
        Spring (+)
        Summer (+)
        Fall (+)
    Product
        Cars (+)
        Trucks (+)
        Motorcycles (+)
    Market
        Chicago (+)
        New York (+)
        LA (+)
    Measures
        Profits (+)
            Sales (+)
            Expenses (-)
        Inventory (+)
        Ratios (+)
    Distribution Channel
        Retail (+)
        Wholesale (+)
```

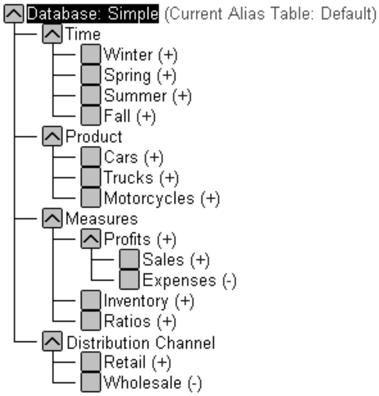
Because the Simple database now has 5 standard dimensions and 17 stored members, the maximum potential number of values is 5 x 4 x 4 x 6 x 3 = 1,440.

When you add a new standard dimension to an outline, you must associate any data in the database with one of the members of the new dimension. For example, in the Simple database, you would have to specify whether the existing data represented Retail or Wholesale. You would then need to load data and calculate the database. For more information, see Chapter 20, "Introducing Data Loading" and Chapter 24, "Introduction to Database Calculations."

Removing a Standard Dimension

The TCC company wants the Simple database to represent the LA market only, so there is no need for a Market dimension. Delete the Market dimension from the database outline. The changed outline is shown in Figure 19. For more information, see Chapter 7, "Creating and Changing Database Outlines."

Figure 19: Deleting a Standard Dimension and Members



```
Database: Simple (Current Alias Table: Default)
  Time
    Winter (+)
    Spring (+)
    Summer (+)
    Fall (+)
  Product
    Cars (+)
    Trucks (+)
    Motorcycles (+)
  Measures
    Profits (+)
      Sales (+)
      Expenses (-)
    Inventory (+)
    Ratios (+)
  Distribution Channel
    Retail (+)
    Wholesale (-)
```

This decision impacts the database in that one less standard dimension diminishes the overall size of the database. However, data for all members in the Market dimension still exists. If you delete a standard dimension from a database outline, the data associated with one member of the deleted standard dimension is retained. You must choose which member's data to retain.

For example, removing the Market dimension from the outline implies that you want to retain data for one member of the Market dimension. In this case, you choose to retain the LA data.

When you delete a standard dimension, you need to recalculate data to reflect changes to the relationships.

Chapter 3

Basic Architectural Elements

In this chapter, you will learn how Essbase OLAP Server improves performance by reducing storage space and speeding up data retrieval for multidimensional databases. This chapter contains the following sections:

Attribute Dimensions and Standard Dimensions

Essbase has two types of dimensions: attribute dimensions and standard dimensions (non-attribute dimensions). This chapter primarily considers standard dimensions because Essbase does not allocate storage for attribute dimension members. Instead it dynamically calculates the members when the user requests data associated with them.

An attribute dimension is a special type of dimension that is associated with a standard dimension. For more information about attribute dimensions, see Chapter 9, "Working with Attributes."

Sparse and Dense Dimensions

Most data sets of multidimensional applications have two characteristics:

- Data is *not* smoothly and uniformly distributed.
- Data does *not* exist for the majority of member combinations. For example, all products may not be sold in all areas of the country.

Essbase maximizes performance by dividing the standard dimensions of an application into two types: *dense dimensions* and *sparse dimensions*. This division allows Essbase to cope with data that is not smoothly distributed, without losing the advantages of matrix-style access to the data. Essbase speeds up data retrieval while minimizing the memory and disk requirements.

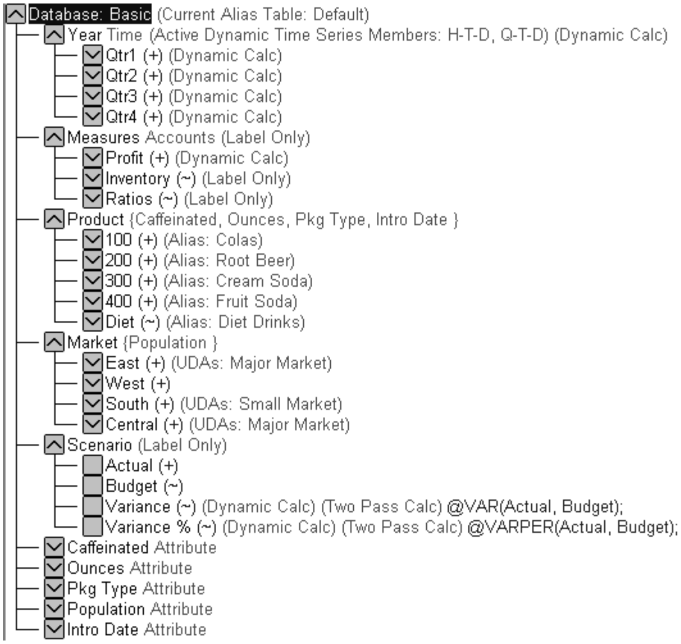
Most multidimensional databases are inherently sparse: they lack data values for the majority of member combinations. A sparse dimension is a dimension with a low percentage of available data positions filled.

For example, the Sample Basic database shown in Figure 20 includes the Year, Product, Market, Measures, and Scenario dimensions. Product represents the product units, Market represents the geographical regions in which the products are sold, and Measures represents the accounts data. Because not every product is sold in every market, Market and Product are chosen as sparse dimensions.

Most multidimensional databases also contain dense dimensions. A dense dimension is a dimension with a high probability that one or more data points is occupied in every combination of dimensions. For example, in the Sample Basic database, accounts data exists for almost all products in all markets, so Measures is chosen as a dense dimension. Year and Scenario are also chosen as dense dimensions. Year represents time in months, and Scenario represents whether the accounts values are budget or actual values.

Note: Caffeinated, Intro Date, Ounces, and Pkg Type are attribute dimensions that are associated with the Product dimension. Population is an attribute dimension that is associated with the Market dimension. Members of attribute dimensions describe characteristics of the members of the dimensions with which they are associated. For example, each product has a size in ounces. Attribute dimensions are always sparse dimensions and must be associated with a sparse standard dimension. Essbase does not store the data for attribute dimensions, Essbase dynamically calculates the data when a user retrieves it. For more information about attribute dimensions, see Chapter 9, “Working with Attributes.”

Figure 20: Sample Basic Database Outline



```
Database: Basic (Current Alias Table: Default)
    Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
        Qtr1 (+) (Dynamic Calc)
        Qtr2 (+) (Dynamic Calc)
        Qtr3 (+) (Dynamic Calc)
        Qtr4 (+) (Dynamic Calc)
    Measures Accounts (Label Only)
        Profit (+) (Dynamic Calc)
        Inventory (~) (Label Only)
        Ratios (~) (Label Only)
    Product {Caffeinated, Ounces, Pkg Type, Intro Date }
        100 (+) (Alias: Colas)
        200 (+) (Alias: Root Beer)
        300 (+) (Alias: Cream Soda)
        400 (+) (Alias: Fruit Soda)
        Diet (~) (Alias: Diet Drinks)
    Market {Population }
        East (+) (UDAs: Major Market)
        West (+)
        South (+) (UDAs: Small Market)
        Central (+) (UDAs: Major 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
```

Data Blocks and the Index System

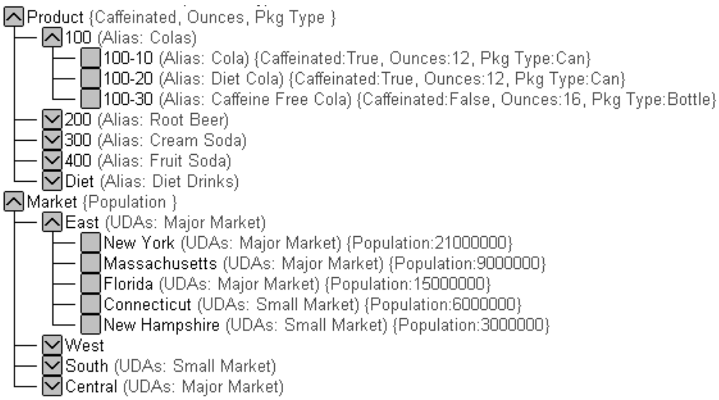
Essbase uses two types of internal structures to store and access data: data blocks and the index system.

Essbase creates a *data block* for each unique combination of sparse standard dimension members (providing that at least one data value exists for the sparse dimension member combination). The data block represents all the dense dimension members for its combination of sparse dimension members.

Essbase creates an *index entry* for each data block. The index represents the combinations of sparse standard dimension members. It contains an entry for each unique combination of sparse standard dimension members for which at least one data value exists.

For example, in the Sample Basic database outline shown in Figure 21, Product and Market are sparse dimensions.

Figure 21: Product and Market Dimensions from the Sample Basic Database



```
Product {Caffeinated, Ounces, Pkg Type }
  100 (Alias: Colas)
    100-10 (Alias: Cola) {Caffeinated:True, Ounces:12, Pkg Type:Can}
    100-20 (Alias: Diet Cola) {Caffeinated:True, Ounces:12, Pkg Type:Can}
    100-30 (Alias: Caffeine Free Cola) {Caffeinated:False, Ounces:16, Pkg Type:Bottle}
  200 (Alias: Root Beer)
  300 (Alias: Cream Soda)
  400 (Alias: Fruit Soda)
  Diet (Alias: Diet Drinks)
Market {Population }
  East (UDAs: Major Market)
    New York (UDAs: Major Market) {Population:21000000}
    Massachusetts (UDAs: Major Market) {Population:9000000}
    Florida (UDAs: Major Market) {Population:15000000}
    Connecticut (UDAs: Small Market) {Population:6000000}
    New Hampshire (UDAs: Small Market) {Population:3000000}
  West
  South (UDAs: Small Market)
  Central (UDAs: Major Market)
```

If data exists for Caffeine Free Cola in New York, then Essbase creates a data block and an index entry for the sparse member combination of Caffeine Free Cola (100-30) -> New York. If Caffeine Free Cola is *not* sold in Florida, then Essbase does *not* create a data block or an index entry for the sparse member combination of Caffeine Free Cola (100-30) -> Florida.

The data block Caffeine Free Cola (100-30) -> New York represents all the Year, Measures, and Scenario dimensions for Caffeine Free Cola (100-30) -> New York.

Each unique data value can be considered to exist in a cell in a data block. When Essbase searches for a data value, it uses the index to locate the appropriate data block asd shown in Figure 22. Then, within the data block, it locates the cell containing the data value. The index entry provides a pointer to the data block. The index handles sparse data efficiently because it includes only pointers to existing data blocks.

Figure 22: Simplified Index and Data Blocks

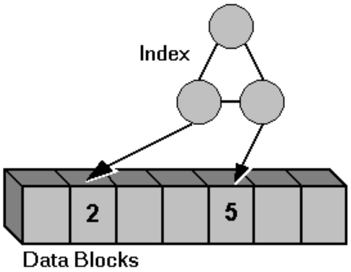
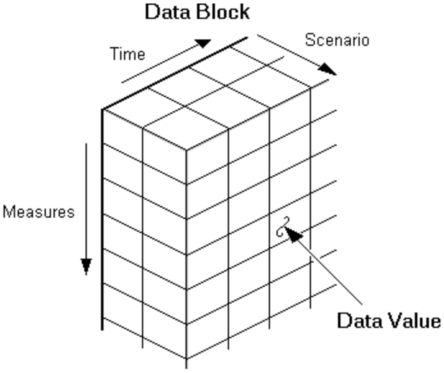


Figure 23 shows part of a data block for the Sample Basic database. Each dimension of the block represents a dense dimension in the Sample Basic database: Time, Measures, and Scenario. A data block exists for each unique combination of members of the Product and Market sparse dimensions (providing that at least one data value exists for the combination).

Figure 23: Part of a Data Block for the Sample Basic Database



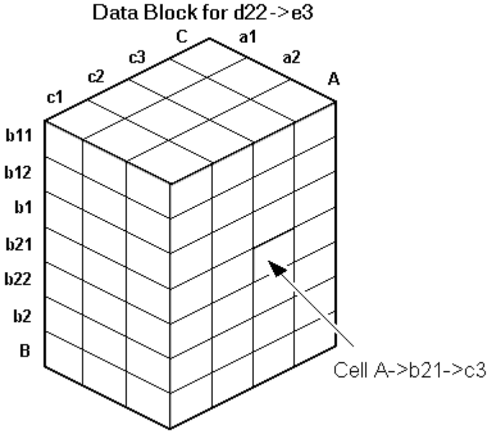
Each data block is a multidimensional array that contains a fixed, ordered location for each possible combination of dense dimension members. Accessing a cell in the block does not involve sequential or index searches. The search is almost instantaneous, resulting in optimal retrieval and calculation speed.

Essbase orders the cells in a data block according to the order of the members in the dense dimensions of the database outline.

```
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
```

The block in Figure 24 represents the three dense dimensions from within the combination of the sparse members d22 and e3 in the preceding database outline. In Essbase, member combinations are denoted by the cross-dimensional operator. The symbol for the cross-dimensional operator is ->. So d22, e3 is written d22 -> e3. A, b21, c3 is written A -> b21 -> c3.

Figure 24: Data Block Representing Dense Dimensions for d22 -> e3



Essbase creates a data block for every unique combination of the members of the sparse dimensions D and E (providing that at least one data value exists for the combination).

Data blocks, such as the one shown in Figure 24, may include cells that do not contain data values. A data block is created if at least one data value exists in the block. Essbase compresses data blocks with missing values on disk, expanding each block fully as it brings the block into memory. Data compression is optional, but is enabled by default. For more information, see “Data Compression” on page 1124.

By carefully selecting dense and sparse standard dimensions, you can ensure that data blocks do not contain many empty cells. In Essbase, empty cells are known as missing or `#MISSING` data. You can also minimize disk storage requirements and maximize performance.

Selection of Sparse and Dense Dimensions

In most data sets, existing data tends to follow predictable patterns of density and sparsity. If you match patterns correctly, you can store the existing data in a reasonable number of fairly dense data blocks, rather than in many highly sparse data blocks.

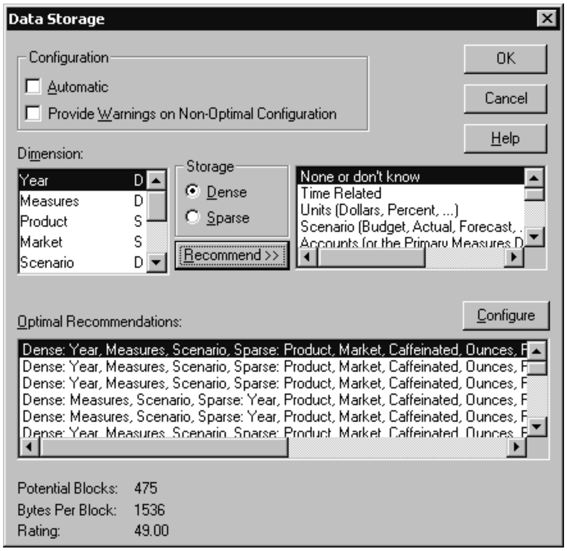
Changing the Sparse-Dense Configuration

When you add a dimension to an outline in Outline Editor, Essbase automatically sets the dimension as sparse.

To help you determine whether dimensions should be sparse or dense, Essbase provides data storage information in the Data Storage dialog box in Application Manager.

➤ To open the dialog box shown in Figure 25, open the database outline and choose Settings > Data Storage.

Figure 25: Data Storage Dialog Box



☐ Automatic
☐ Provide Warnings on Non-Optimal Configuration

Dimension:

Year	D
Measures	D
Product	S
Market	S
Scenario	D

● Dense
○ Sparse

None or don't know
Time Related
Units (Dollars, Percent, ...)
Scenario (Budget, Actual, Forecast, .
Accounts (or the Primary Measures D

Optimal Recommendations:

Dense: Year, Measures, Scenario, Sparse: Product, Market, Caffeinated, Ounces, F
Dense: Year, Measures, Scenario, Sparse: Product, Market, Caffeinated, Ounces, F
Dense: Year, Measures, Scenario, Sparse: Product, Market, Caffeinated, Ounces, F
Dense: Measures, Scenario, Sparse: Year, Product, Market, Caffeinated, Ounces, F
Dense: Measures, Scenario, Sparse: Year, Product, Market, Caffeinated, Ounces, F
Dense: Year, Measures, Scenario, Sparse: Product, Market, Caffeinated, Ounces, F

Potential Blocks:	475
Bytes Per Block:	1536
Rating:	49.00

Essbase can make recommendations for the sparse-dense configuration of dimensions based on the following factors:

- The time and accounts tags on dimensions
- The probable size of the data blocks
- Characteristics that you attribute to the dimensions in this dialog box

Click the Recommend button to open the recommendations portion of the dialog box. For more information on time and accounts tags, see "Setting the Dimension Type" on page 194. For more information on using the Outline Editor to create database outlines, see Chapter 7, "Creating and Changing Database Outlines."

You can apply a recommended configuration or you can turn off automatic configuration and manually set the sparse or dense property for each dimension. Attribute dimensions are always sparse dimensions. Keep in mind that you can associate attribute dimensions only with sparse standard dimensions.

Note: The automatic configuration of dense and sparse dimensions provides only an estimate. It cannot take into account the nature of the data you will load into your database or multiple user considerations.

Determining the Sparse-Dense Configuration for Sample Basic

Consider the Sample Basic database that is shipped with Essbase. The Sample Basic database represents data for The Beverage Company (TBC).

TBC does not sell every product in every market; therefore, the data set is reasonably sparse. Data values do not exist for many combinations of members in the Product and Market dimensions. For example, if Caffeine Free Cola is not sold in Florida, then data values do not exist for the combination Caffeine Free Cola (100-30)->Florida. So, Product and Market are sparse dimensions. Therefore, if no data values exist for a specific combination of members in these dimensions, Essbase does not create a data block for the combination.

However, consider combinations of members in the Year, Measures, and Scenario dimensions. Data values almost always exist for some member combinations on these dimensions. For example, data values exist for the member combination Sales->January->Actual because at least some products are sold in January. Thus, Year and, similarly, Measures and Scenario are dense dimensions.

The sparse-dense configuration of the standard dimensions in the Sample Basic database may be summarized as follows:

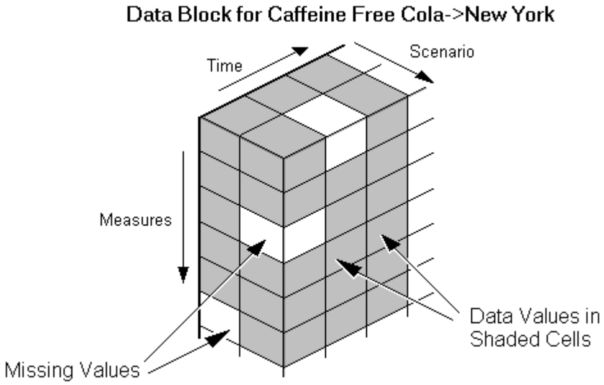
- The sparse standard dimension are Product and Market.
- The dense standard dimensions are Year, Measures, and Scenario.

Essbase creates a data block for each unique combination of members in the Product and Market dimensions. Each data block represents data from the dense dimensions. The data blocks are likely to have few empty cells.

For example, consider the sparse member combination Caffeine Free Cola (100-30), New York, illustrated by Figure 26:

- If accounts data (represented by the Measures dimension) exists for this combination for January, it probably exists for February and for all members in the Year dimension.
- If a data value exists for one member on the Measures dimension, then it is likely that other accounts data values exist for other members in the Measures dimension.
- If Actual accounts data values exist, then it is likely that Budget accounts data values exist.

Figure 26: Dense Data Block for Sample Basic Database


Data Block for Caffeine Free Cola->New York
Time
Scenario
Measures
Data Values in Shaded Cells
Missing Values

Dense and Sparse Selection Scenarios

The following scenarios show how a database is affected when you select different dense and sparse standard dimensions. Assume that these scenarios are based on typical databases with at least seven dimensions and several hundred members:

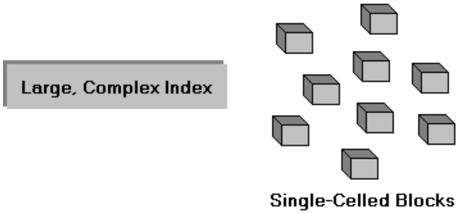
- "Scenario 1: All Sparse Standard Dimensions" on page 89
- "Scenario 2: All Dense Standard Dimensions" on page 90
- "Scenario 3: Dense and Sparse Standard Dimensions" on page 90
- "Scenario 4: A Typical Multidimensional Problem" on page 91

Scenario 1: All Sparse Standard Dimensions

If you make all dimensions sparse, Essbase creates data blocks that consist of single data cells that contain single data values. An index entry is created for each data block and, therefore, in this scenario, for each existing data value.

This configuration produces a huge index that requires a large amount of memory. The more index entries, the longer Essbase searches to find a specific block.

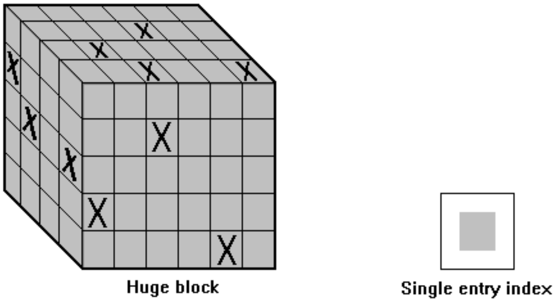
Figure 27: Database with All Sparse Standard Dimensions



Scenario 2: All Dense Standard Dimensions

If you make all dimensions dense, as shown in Figure 28, Essbase creates one index entry and one very large, very sparse block. In most applications, this configuration requires thousands of times more storage than other configurations. Essbase needs to load the entire block into memory when it searches for a data value, which requires enormous amounts of memory.

Figure 28: Database with All Dense Standard Dimensions

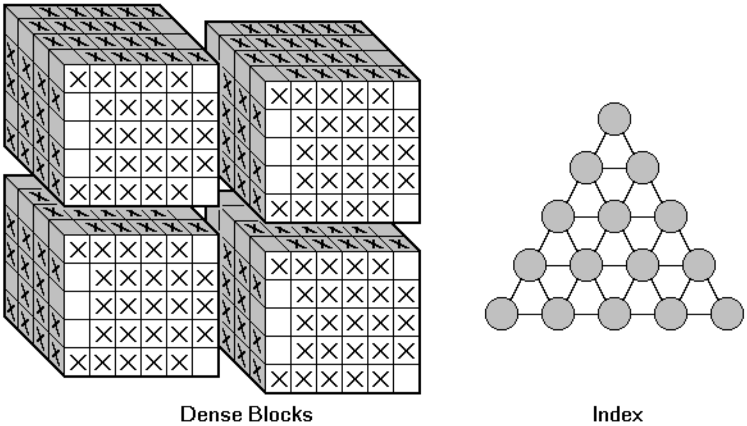


Scenario 3: Dense and Sparse Standard Dimensions

Based upon your knowledge of your company's data, you have identified all your sparse and dense standard dimensions. Ideally, you have approximately equal numbers of sparse and dense standard dimensions. If not, you are probably working with a non-typical data set and you need to do more tuning to define the dimensions.

Essbase creates dense blocks that can fit into memory easily and creates a relatively small index as shown in Figure 29. Your database runs efficiently using minimal resources.

Figure 29: An Ideal Configuration with Combination of Dense and Sparse Dimensions

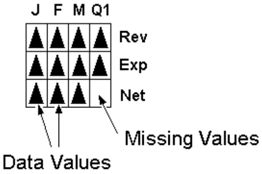


Scenario 4: A Typical Multidimensional Problem

Consider a database with four standard dimensions: Time, Accounts, Region, and Product. In the following example, Time and Accounts are dense dimensions, and Region and Product are sparse dimensions.

The two-dimensional data blocks shown in Figure 30 represent data values from the dense dimensions: Time and Accounts. The members in the Time dimension are J, F, M and Q1. The members in the Accounts dimension are Rev, Exp, and Net.

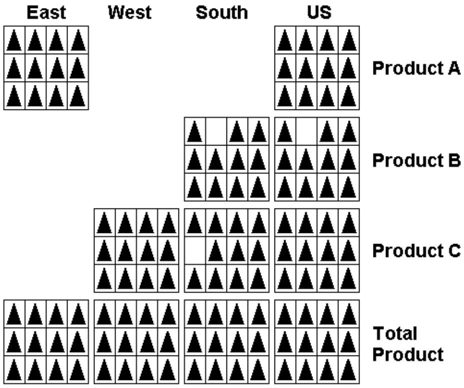
Figure 30: Two-dimensional Data Block for Time and Accounts



Essbase creates data blocks for combinations of members in the sparse standard dimensions (providing at least one data value exists for the member combination). The sparse dimensions are Region and Product. The members of the Region dimension are East, West, South, and Total US. The members in the Product dimension are Product A, Product B, Product C, and Total Product.

Figure 31 shows 11 data blocks. No data values exist for Product A in the West and South, for Product B in the East and West, and for Product C in the East. Therefore, Essbase has not created data blocks for these member combinations. The data blocks that Essbase has created have very few empty cells.

Figure 31: Data Blocks Created for Sparse Members on Region and Product

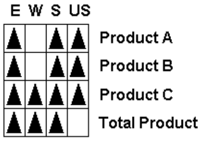


This example effectively concentrates all the sparseness into the index and concentrates all the data into fully utilized blocks. This configuration provides efficient data storage and retrieval.

Now consider a reversal of the dense and sparse dimension selections. In the following example, Region and Product are dense dimensions, and Time and Accounts are sparse dimensions.

As shown in Figure 32, the two-dimensional data blocks represent data values from the dense dimensions: Region and Product.

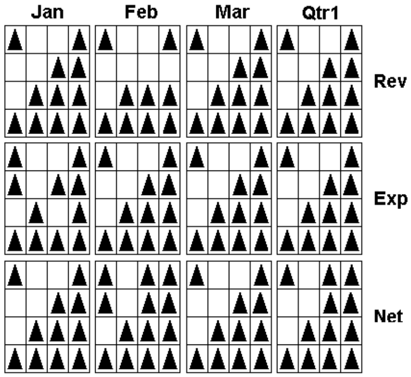
Figure 32: Two-Dimensional Data Block for Region and Product



Essbase creates data blocks for combinations of members in the sparse standard dimensions (providing at least one data value exists for the member combination). The sparse standard dimensions are Time and Accounts.

Figure 33 shows 12 data blocks. Data values exist for all combinations of members in the Time and Accounts dimensions; therefore, Essbase creates data blocks for all the member combinations. Because data values do not exist for all products in all regions, the data blocks have many empty cells. Data blocks with many empty cells store data inefficiently.

Figure 33: Data Blocks Created for Sparse Members on Time and Accounts



The Essbase Solution

When you create an optimized Essbase database, you need to consider carefully the following questions:

- How does your company use the data?
- How do you plan to build and order the dimensions?
- Which data compression scheme will you use?
- How do you want to create and order calculations?

For more information on:

- Planning the development of your multidimensional database, see Chapter 5, “Case Study: Designing a Single-Server Application.”
- Selecting dense and sparse dimensions, see “Sparse and Dense Dimensions” on page 3-80.
- Loading data, see Chapter 20, “Introducing Data Loading.”
- Compressing data and optimizing your database, see “Data Compression” on page 1124.
- Calculating your database, see Chapter 24, “Introduction to Database Calculations.”

Chapter 4

Quick Start for Implementing Essbase

If you want to get started as quickly as possible, use Table 3 to identify the steps to get up and running with Essbase. This table also tells you where you can find more information about each step. Unless otherwise specified, the chapters refer to this *Essbase Database Administrator's Guide*.

Note: This chapter assumes you are a new Essbase user. If you used Version 5.x, you need to migrate your applications and databases. See the *Essbase Installation Guide* for important migration information.

Table 3: A Process Map

Process step	Reference
Install Essbase. Decide what components you want to install. Be aware that the license your company purchased might not include all options.	*Essbase Installation Guide*
Assess your needs and requirements. Have a clear idea of your data analysis needs and of what types of calculations and reports you want to run.	Your budget, forecasting, and other financial reports with notes on how you want to improve them

Table 3: A Process Map (Continued)

Process step	**Reference**
Analyze your data from a multidimensional perspective. Consider: • Where are your data sources? • What type is the data? Is it detailed, relational data or is it higher-level, hierarchical data that can be used for analysis? • In what format is the data? • How will you access the data? If you need to access relational data, you may need SQL Interface or Integration Server (a separately purchasable product).	• Chapter 2, “Multidimensional Concepts” • *Essbase SQL Interface Guide* • *ODBC drivers documentation* • Essbase Integration Services documentation
Learn the fundamentals of Essbase and distributed OLAP.	• Chapter 1, “Essbase and Multidimensional Databases” • Chapter 2, “Multidimensional Concepts” • Chapter 3, “Basic Architectural Elements” • Chapter 6, “Creating Applications and Databases” • Chapter 9, “Working with Attributes” • Chapter 13, “Designing Partitioned Applications” • Attend an Essbase training class; contact your software provider for details.
Design your application and database. Think about which dimensions you will designate as sparse and which as dense, which dimensions you will designate as time and which as account, and where you will include attribute dimensions.	Chapter 5, “Case Study: Designing a Single-Server Application”
Estimate the size of your database.	Appendix C, “Estimating Disk and Memory Requirements”

Table 3: A Process Map (Continued)

Process step	**Reference**
Allocate storage and specify Essbase kernel settings for your database.	• Chapter 38, "Managing Essbase Kernel Settings" • Chapter 39, "Allocating Storage and Compressing Data"
Learn about Partitioning. Think about whether your data can benefit from being decentralized into connected databases.	• Chapter 13, "Designing Partitioned Applications" • Chapter 14, "Building and Maintaining Partitions"
Learn about dynamic calculations and how they can greatly improve performance.	Chapter 28, "Dynamically Calculating Data Values"
Create an application and a database.	Chapter 6, "Creating Applications and Databases"
Design security for your database. Think about who needs access to the data, who should have update authority, and who should have read-only access?	Part IV, "Designing and Building a Security System"
Build an outline for your database.	Chapter 7, "Creating and Changing Database Outlines"
Build the dimensions. Decide whether your data loads will introduce new members into the outline. If so, set up dynamic dimension building. If not, set up regular data loads.	• Chapter 18, "Introducing Dynamic Dimension Building" • Chapter 19, "Building Dimensions Using a Rules File"
Load your data.	Part V, "Building Dimensions and Loading Data"
Calculate your database.	Part VI, "Calculating Data"
Run a report.	Part VIII, "Storing and Protecting Data"
Use Spreadsheet Add-in to retrieve data.	The *Essbase Spreadsheet Add-in User's Guide* for your spreadsheet application
Link files or cell notes to data cells.	Chapter 11, "Linking Objects to Essbase Data"
Assign alias names to your members.	Chapter 10, "Creating and Managing Aliases"

Table 3: A Process Map (Continued)

Process step	Reference
Copy or export data subsets.	Chapter 37, “Copying Data Subsets and Exporting Data to Other Programs”
Back up and restore your data.	Chapter 44, “Backing Up and Restoring Data”
Design a currency application.	Chapter 12, “Designing and Building Currency Conversion Applications”
Fine-tune your database performance and storage settings.	• Chapter 38, “Managing Essbase Kernel Settings” • Chapter 39, “Allocating Storage and Compressing Data” • Chapter 46, “Monitoring Performance”
Automate routine operations by using ESSCMD.	Chapter 45, “Automating the Production Environment”
Maintain your applications.	Part IX, “Maintaining and Automating Hyperion Essbase”
Analyze and improve performance and troubleshoot errors if they occur.	Part X, “Optimizing and Troubleshooting Essbase”

Part

II

Designing and Creating Applications and Databases

Part II describes how to create single-server Essbase applications and databases. This part includes information on attributes, aliases, linked reporting objects, and currency conversion applications. Part II contains the following chapters:

- Chapter 5, “Case Study: Designing a Single-Server Application,” describes the rules used to design a single-server, multidimensional database solution for your organization. It contains detailed examples that show how to apply these rules.
- Chapter 6, “Creating Applications and Databases,” describes how to create and manage applications and databases.
- Chapter 7, “Creating and Changing Database Outlines,” describes how to create and modify database outlines using the Application Manager.
- Chapter 8, “Setting Dimension and Member Properties,” illustrates how to set attributes for the dimensions and members in an outline using the Application Manager.
- Chapter 9, “Working with Attributes,” describes attribute dimensions and members and how to define them using the Application Manager.
- Chapter 10, “Creating and Managing Aliases,” describes how to create one or more aliases for dimensions and members in an outline using the Application Manager.
- Chapter 11, “Linking Objects to Essbase Data,” describes how to link various kinds of data with any cell in a Essbase database.
- Chapter 12, “Designing and Building Currency Conversion Applications,” introduces you to currency applications, describes how to create them, and explains how to calculate currency conversions.

Chapter

5

Case Study: Designing a Single-Server Application

To implement a multidimensional database, first you install Essbase, and then you design and create an application. You analyze data sources and define requirements very carefully and then decide whether a single-server application or a partitioned, distributed approach best serves your needs. For criteria that you can review to decide whether to partition an application, see Chapter 13, “Designing Partitioned Applications.”

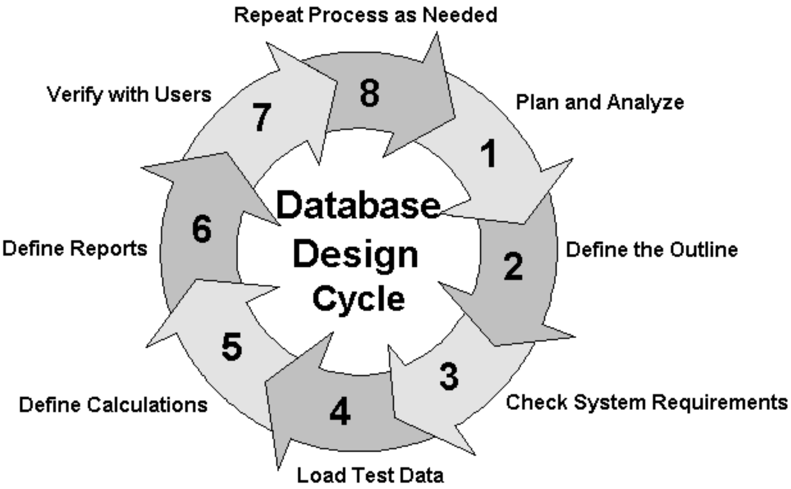
Using a case study, this chapter provides an overview of the database planning process and discusses working rules that you can follow to design a single-server, multidimensional database solution for your organization. For detailed information about building applications and databases, see Chapter 6, “Creating Applications and Databases.”

This chapter includes the following sections:

Designing an Application

As illustrated in Figure 34, designing an application is a cyclic process that moves from a planning stage to verification stage.

Figure 34: The Database Design Cycle



The database design process includes the following basic steps:

1. **Analyze business needs and plan the database.** In this step you determine the information needs that the application and database must satisfy. You identify source data, user information, and access needs.
2. **Define the database outline.** The *outline* determines the structure of the database—what information is stored and how different pieces of information relate to one another.
3. **Check system requirements.** How you meet system requirements and define system parameters affects the efficiency and performance of the database. Make sure that you have allocated enough disk space for the database and that the index and data file caches in memory are of adequate size (see Appendix C, “Estimating Disk and Memory Requirements” and Chapter 39, “Allocating Storage and Compressing Data”).
4. **Load test data into the database.** After an outline and a security plan are in place, you load the database with test data to enable the later steps of the process.

5. **Define calculations.** Now you test the consolidations in the outline and write and test formulas and calculation scripts for specialized calculations.
6. **Define reports.** Users access data through printed or online reports and spreadsheets and even on the World Wide Web. If you plan to provide predefined reports to users, design the report layouts and run the reports.
7. **Verify with users.** It is important to ensure that the database satisfies the goals of the application. You should solicit and carefully consider the opinions of the users. Do the calculations give them the information they need? Are they able to generate reports quickly? Are they satisfied with consolidation times? In short, ask users if the database works for them.
8. **Repeat the process.** To fine-tune the design, repeat steps 1 through 7.

Case Study: The Beverage Company

This chapter bases the database planning process on the needs of a fictitious company called *The Beverage Company* (TBC) and uses TBC as an example to demonstrate how to build an Essbase database. The examples follow a variation of the Sample Basic application that is included with the Essbase installation.

TBC manufactures, markets, and distributes soft drink products internationally. Analysts at TBC prepare budget forecasts and compare performance to budget forecasts on a monthly basis. The financial measures the analysts track are profit and loss and inventory data.

TBC uses spreadsheet packages to prepare budget data and perform variance reporting. Because TBC plans and tracks a variety of products over several markets, the process of deriving and analyzing data is tedious. Last month, analysts spent most of their time entering and rekeying data and preparing reports.

TBC has determined that Essbase is the best tool for creating a centralized repository for financial data. The data repository will reside on a server that is accessible to analysts throughout the organization. Users will have access to the server and will be able to load data from various sources and retrieve data when they need it. TBC has a variety of users, so TBC expects that different users will have different security levels for accessing data.

Planning and Analyzing

The design and operation of an Essbase multidimensional database plays a key role in achieving a well-tuned system that enables you to analyze a business efficiently. Given the size and performance volatility of multidimensional databases, developing an optimized database is critical.A detailed plan that outlines data sources, user needs, and prospective database elements can save you development and implementation time.

The planning and analysis phase involves three tasks:

- "Identifying Source Data" on page 104
- "Identifying User Requirements" on page 105
- "Creating a Business Model" on page 106

When designing a multidimensional application, consider these factors:

- How information flows within the company—who uses what data for what purposes
- The types of reporting the company does—what types of data must be included in the outline to serve user reporting needs

Before you create a database and build its outline, you must create an Essbase application to contain it. Applications that use the optional Essbase currency conversion module generally consist of a main database and a separate currency database (see Chapter 12, "Designing and Building Currency Conversion Applications").

Identifying Source Data

First, you need to evaluate the source data that you want to include in the database. Think about where the data resides and how often you plan to update the database with the data. This up-front research saves time when you create the database outline and load data into the Essbase database.

Determine the scope of the database. If an organization has thousands of product families containing hundreds of thousands of products, you may want to store data values only for product families. Interview members from each user department to find out what data they process, how they process data today, and how they want to process data in the future.

Carefully define reporting and analysis needs.

- Does the data support these needs?
- If not, what additional data do you need and where will you find the needed data?

Where does each department currently store data?

- Is it in a form that Essbase can use?
- Do departments store data in a DB2 database on an IBM mainframe, in a relational database on a UNIX-based server, or in a PC-based database or spreadsheet?
- Who updates the database and how frequently?
- Do the individuals who need to update data have access to the data?

Finally, make sure that all the data you want is ready to load in the necessary format. You can save hours of time by making sure that data is readily available and easy for Essbase to import. For a list of valid data sources that you can import into Essbase, see Chapter 23, “Performing and Debugging a Data Load.”

Use this checklist to select data for the Essbase database

- How much detail should the database contain? Does the data support the desired analysis and reporting goals?
- Where is the data? Does it come from a single source or from multiple sources?
- Is all the data you want to use readily available?
- Is the data in a format that Essbase can import?

Identifying User Requirements

Given the data sources, what types of analysis do users require? What summary and detail levels of information do they need? Do some users require access to information that others should not see?

Be sure to discuss information needs with users. Review the information they use now and the reports they must generate for review by others.

Creating a Business Model

You are now ready to create a model of the business on paper. To build the model, you need to identify the perspectives and views that are important to the business. These views translate into the dimensions of the database model.

Most businesses choose to analyze include the following areas:

- Time periods
- Accounts
- Scenarios
- Products
- Distribution channels
- Geographical regions
- Business units

Use these sections to help you gather information:

- "Identifying Analysis Objectives" on page 106
- "Determining Dimensions and Members" on page 107
- "Analyzing Database Design" on page 110
- "Planning for Security in a Multi-User Environment" on page 118

Identifying Analysis Objectives

After you identify the major areas of information in a business, the next step in designing a Essbase application is deciding how the application enables analysis of data:

- If by time, which time periods are needed? Does the analysis need to include just the current year or multiple years? Quarterly and monthly data? By season?
- If by geographical region, how do you define these regions? By sales territories? By geographical boundaries such as states and cities?
- If by product line, do you need to review data for each specific product or can you summarize data into product classes?

Regardless of the business area, you need to determine the perspective and detail needed in the analysis. Each business area you analyze provides a different view of the data.

Determining Dimensions and Members

You can represent each of the business views as a separate dimension in the database. If you need to analyze a business area by classification or attribute, such as by the size or color of products, you can use attribute dimensions to represent the classification views. For information about attribute dimensions, see Chapter 9, “Working with Attributes.”

The dimensions that you choose determine what types of analysis you can perform on the data. With Essbase, you can use as many dimensions as you need for your analysis. A typical Essbase database contains at least seven standard dimensions and many more attribute dimensions. Dimensions that are not attribute dimensions are called *standard dimensions*.

After you determine the dimensions of the business model, choose the elements or items within the perspective of each dimension. These elements become the members within the dimensions. For example, a perspective of time may include the time periods that you want to analyze, such as quarters, and within quarters, months. Each quarter and month becomes a member of the dimension that you create for time. Quarters and months represent a two-level hierarchy of members and their children. Months within a quarter consolidate to a total for each quarter.

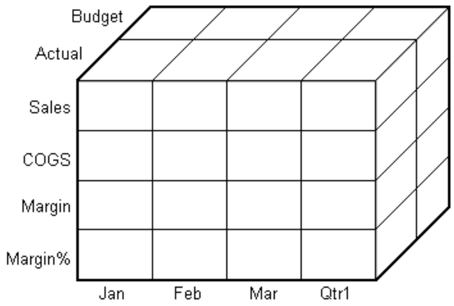
Next, consider the relationships among the business areas. The structure of a Essbase database makes it easy for users to analyze information from many different perspectives. A financial analyst, for example, may ask the following questions:

- What are sales for a particular month? How does this figure compare to sales in the same month over the last five years?
- By what percentage is profit margin increasing?
- How close are actual values to budgeted values?

In other words, the analyst may want to examine information from three different perspectives: time, account, and scenario. The sample database shown in Figure 35 represents these three perspectives as three dimensions, with one dimension represented along each of the three axes:

- A time dimension, which consists of the individual months Jan, Feb, and Mar and the total for Qtr1, is displayed along the X-axis.
- An accounts dimension, which consists of accounting figures such as Sales, COGS, Margin, and Margin%, is displayed along the Y-axis.
- Another dimension which provides a different point of view, such as Budget for budget values and Actual for actual values, is displayed along the Z-axis.

Figure 35: Cube Representing Three Database Dimensions



The cells within the cube, where the members intersect, contain the data relevant to all three intersecting members; for example, the actual sales in January.

Table 4 shows a summary of TBC's business areas that the planner determined would be dimensions. These are the major business areas to be analyzed. The planner created three columns, with the dimensions in the left column and members in the two right columns. The members in column 3 are subcategories of the members in column 2. In some cases, members in column 3 are broken into another level of subcategories; for example, the Margin and Total Expenses members of the Measures dimension.

Table 4: TBC Sample Dimensions

Dimensions	Members	Child Members
Year	Qtr1	Jan, Feb, Mar
	Qtr2	Apr, May, Jun
	Qtr3	Jul, Aug, Sep
	Qtr4	Oct, Nov, Dec

Table 4: TBC Sample Dimensions (Continued)

Dimensions	**Members**	**Child Members**
Measures	Profit	Margin: Sales, COGS Total Expenses: Marketing, Payroll, Miscellaneous
	Inventory	Opening Inventory, Additions, Ending Inventory
	Ratios	Margin %, Profit %, Profit per Ounce
Product	Colas (100)	Cola (100-10), Diet Cola (100-20), Caffeine Free Cola (100-30)
	Root Beer (200)	Old Fashioned (200-10), Diet Root Beer (200-20), Sarsaparilla (200-30), Birch Beer (200-40)
	Cream Soda (300)	Dark Cream (300-10), Vanilla Cream (300-20), Diet Cream Soda (300-30)
	Fruit Soda (400)	Grape (400-10), Orange (400-20), Strawberry (400-30)
Market	East	New York, Massachusetts, Connecticut, Florida, New Hampshire
	West	Oregon, Washington, California, Utah, Nevada
	South	Texas, Louisiana, New Mexico, Oklahoma
	Central	Illinois, Ohio, Wisconsin, Missouri, Iowa, Colorado
Scenario	Actual	
	Budget	
	Variance	
	Variance %	

In addition, some desired views are actually ways to view product information. For these views, the planner added these two attribute dimensions to enable product analysis based on size and packaging:

Table 5: TBC Sample Attribute Dimensions

Dimensions	Members	Child Members
Ounces	Large Small	64, 32, 20 16, 12
Pkg Type	Bottle Can	

Use this checklist to create a business model

- What are the candidates for dimensions?
- Do any of the dimensions classify or describe other dimensions? These are candidates for attribute dimensions.
- Do users want to qualify their view of a dimension? The categories by which they qualify a dimension are candidates for attribute dimensions.
- What are candidates for members?
- How many levels does the data require?
- How does the data consolidate?

Analyzing Database Design

While the initial dimension design is still on paper, you should review the design according to a set of guidelines. These guidelines help you to fine-tune the database and leverage the multidimensional technology. These rules are processes or questions that help you achieve an efficient design and meet consolidation and calculation goals.

Keep in mind that the number of members needed to describe a potential data point should determine the number of dimensions. As you analyze the design, if you are not sure that you should delete a dimension, keep it and apply more analysis rules until you feel confident about deleting or keeping it.

Use these sections to analyze your database design:

- "Examine dimension relationships" on page 111
- "Examine dimension combinations" on page 113
- "Avoid repetition in the outline" on page 115
- "Avoid interdimensional irrelevance" on page 116
- "Split databases if necessary" on page 117
- "Checklist to analyze the database" on page 118

Examine dimension relationships

For simplicity, the examples in this section show alternative arrangements for what was initially designed as two dimensions. You can apply the same logic to all combinations of dimensions.

Consider the design for a company that sells products to multiple customers over multiple markets; the markets are unique to each customer:

```
             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 is only in New York, Cust B is only in Illinois, and Cust C is only in California. The company can define the data in one standard dimension:

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

However, if you look at a larger sampling of data, you may see that there can be many customers in each market. Cust A and Cust E are in New York; Cust B, Cust M, and Cust P are in Illinois; Cust C and Cust F are in California. In this situation, the company typically defines the large dimension, Customer, as a standard dimension and Market as an attribute dimension. The company associates the members of the Market dimension as attributes of the members of the Customer dimension. The members of the Market dimension describe locations of the customers.

```
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
```

Consider another situation. Again, the company sells products to multiple customers over multiple markets. This time, the company can ship to a customer that has locations in different markets:

	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 is in New York and California. Cust B is in New York and Illinois. Cust C is only in California. Using an attribute dimension does not work in this situation; a customer cannot have more than one attribute. Therefore, the company designs the data in two standard dimensions:

```
Customer
       Cust A
       Cust B
       Cust C

Market
       New York
       Illinois
       California
```

Examine dimension combinations

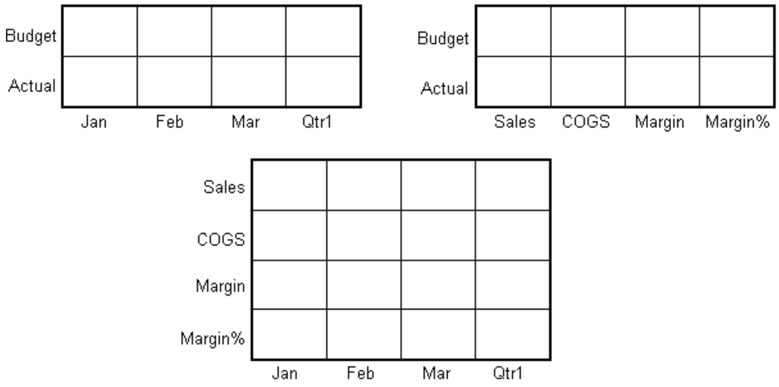
Break each combination of two dimensions into a two-dimensional matrix. For example, proposed dimensions at TBC (as listed in Table 4) include the following combinations:

- Year vs. Measures
- Year vs. Product
- Year vs. Market
- Year vs. Scenario
- Measures vs. Product
- Measures vs. Market
- Measures vs. Scenario
- Market vs. Product
- Market vs. Scenario
- Scenario vs. Product
- Ounces vs. Pkg Type

As attribute dimensions associated with the Product dimension, Ounces and Pkg Type should be considered with the Product dimension.

To help visualize each dimension, you can draw a matrix and include a few of the first generation members. Figure 36 shows a simplified set of matrixes for three dimensions.

Figure 36: Analyzing Dimensional Relationships



For each combination of dimensions, ask three questions:

- Does it add analytic value?
- Does it add utility for reporting?
- Are there many unused combinations?

For each combination, the answers to the questions help determine if the combination is valid for the database. Ideally, the answers to all of the questions should be yes. If all answers are not yes, you should consider rearranging the data into dimensions that are more meaningful. As you go through this process, be sure to discuss information needs with users.

Avoid repetition in the outline

The repetition of elements in an outline often indicates a need to split dimensions. Here is an example of repetition and a solution:

```
Repetition              No Repetition
Accounts                Accounts
  Budget                  Profit
    Profit                  Margin
      Margin                  Sales
        Sales                 COGS
        COGS                Expenses
      Expenses          Scenario
  Actual                  Budget
    Profit                Actual
      Margin
        Sales
        COGS
      Expenses
```

Separating Budget and Actual and placing them into another dimension simplifies the outline and provides a simpler view of the budget and actual figures of the other dimensions in the database.

This left column of this table uses shared members to analyze diet beverages. You can avoid the repetition of the left column and simplify the design of the outline by creating a Diet attribute dimension, as shown in the second example.

Repetition

```
Product
   100 (Alias: Colas)
       100-10 (Alias: Cola)
       100-20 (Alias: Diet Cola)
   200 (Alias: Root Beer)
       200-20 (Alias: Diet Root Beer)
       200-30 (Alias: Birch Beer)
   300 (Alias Cream Soda)
       300-10 (Alias: Dark Cream)
       300-20 (Alias: Diet Cream)
   Diet (Alias: Diet Drinks)
       100-20 (Alias: Diet Cola)
       200-20 (Alias: Diet Root Beer
       300-20 (Alias: Diet Cream)
```

No Repetition

```
Product (Diet)
   100 (Alias: Colas)
       100-10 (Alias: Cola) (Diet: False)
       100-20 (Alias: Diet Cola) (Diet: True)
   200 (Alias: Root Beer)
       200-20 (Alias: Diet Root Beer) (Diet: True)
       200-30 (Alias: Birch Beer) (Diet: False)
   300 (Alias Cream Soda)
       300-10 (Alias: Dark Cream) (Diet: False)
       300-20 (Alias: Diet Cream) (Diet: True)
Diet  Attribute (Type: Boolean)
    True
    False
```

Attribute dimensions also provide additional analytic capabilities. For guidelines on when to use attribute dimensions, see “Attribute Design Considerations” on page 9-236.

Avoid interdimensional irrelevance

Interdimensional irrelevance occurs when many members of a dimension are irrelevant across other dimensions. Essbase defines irrelevant data as data that Essbase stores only at the summary (dimension) level. In such a situation, you may be able to remove a dimension from the database and add its members to another dimension or split the model into separate databases.

For example, TBC considered analyzing salaries as a member of the Measures dimension. But salary information often proves to be irrelevant in the context of a corporate database. Most salaries are confidential and apply to specific individuals. The individual and the salary typically represent one cell, with no reason to intersect with any other dimension.

TBC considered separating employees into a separate dimension. Table 6 shows an example of how TBC analyzed the proposed Employee dimension for interdimensional irrelevance. Members of the proposed Employee dimension are compared with members of the Measures dimension. Only the Salary measure is relevant to individual employees.

Table 6: Interdimensional Irrelevance Example

	Joe Smith	Mary Jones	Mike Garcia	All Employees
Revenue				x
Variable Costs				x
COGS				x
Advertising				x
Salaries	x	x	x	x
Fixed Costs				x
Expenses				x
Profit				x

Split databases if necessary

TBC agreed that in context with other dimensions, individual employees were irrelevant. They also agreed that adding an Employee dimension substantially increases database storage needs. Consequently, they decided to create a separate Human Resources (HR) database. The new HR database contains a group of related dimensions and includes salaries, benefits, insurance, and 401(k) plans.

There are many reasons for splitting a database; for example, suppose that a company maintains an organizational database that contains several international subsidiaries located in several time zones. Each subsidiary relies on time-sensitive financial calculations. You may want to split the database for groups of subsidiaries in the same time zone to ensure that their financial calculations are timely. You can also use a partitioned application to separate information by subsidiary.

Checklist to analyze the database

Use this checklist to analyze your database:

- Have you minimized the number of dimensions?
- For each dimensional combination, did you ask:
  - Does it add analytic value?
  - Does it add utility for reporting?
  - Are there any unused combinations?
- Did you avoid repetition in the outline?
- Did you avoid interdimensional irrelevance?
- Did you split the databases as necessary?

Planning for Security in a Multi-User Environment

The time to think about the type of security privileges you plan to issue for an Essbase database is when you consider user information needs. As an Essbase administrator, you frequently load external data into the database. Determine who else should have privileges for these operations. End the plan with a list of users and privileges. See Chapter 15, “Managing Security for Users and Applications” to learn more about assigning user privileges.

Use this checklist to plan for security

- Who are the users and what privileges should they have?
- Who should have load data privileges?
- Which users can be grouped, and as a group, given similar privileges?

Drafting an Outline

At this point, you can create the application and database and build the first draft of the outline in Essbase. The draft defines all dimensions, members, and consolidations. Use the outline to design consolidation requirements and identify where you need formulas and calculation scripts.

The TBC planners issued the following draft for a database outline. In this plan, the bold headings are the dimensions: Year, Measures, Product, Market, Scenario, Pkg Type, and Ounces. Observe how TBC anticipated consolidations, calculations and formulas, and reporting requirements. The planners also used product codes rather than product names to describe products.

- **Year**—TBC needs to collect data monthly and summarize the monthly data by quarter and year. Monthly data, stored in members such as Jan, Feb, and Mar, consolidates to quarters. Quarterly data, stored in members such as Qtr1 and Qtr2, consolidates into Year.
- **Measures**—Sales, Cost of Goods Sold, Marketing, Payroll, Miscellaneous, Opening Inventory, Additions, and Ending Inventory are standard measures. Essbase can calculate Margin, Total Expenses, Profit, Total Inventory, Profit %, Margin %, and Profit per Ounce from these measures. TBC needs to calculate Measures on a monthly, quarterly, and yearly basis.
- **Product**—The Product codes are 100-10, 100-20, 100-30, 200-10, 200-20, 200-30, 200-40, 300-10, 300-20, 300-30, 400-10, 400-20, and 400-30. Each product consolidates to its respective family (100, 200, 300, and 400). This consolidation allows TBC to analyze by size and package, by associating each product with members of the Ounces and Pkg Type attribute dimensions.
- **Market**—Several states make up a region, and four regions make up a market. The states are New York, Massachusetts, Connecticut, Florida, New Hampshire, Oregon, Washington, California, Utah, Nevada, Texas, Louisiana, New Mexico, Oklahoma, Illinois, Ohio, Wisconsin, Missouri, Iowa, and Colorado. Each state consolidates into its respective region: East, West, South, and Central. Each region consolidates into Market.
- **Scenario**—TBC derives and tracks budget data versus actual data. Managers must monitor and track budgets and actuals, as well as the variance and variance percentage between them.
- **Pkg Type**—TBC wants to see the effect that product packaging has on sales and profit. Establishing the Pkg Type attribute dimension enables users to analyze product information based on whether a product is packaged in bottles or cans.
- **Ounces**—TBC sells products in different sizes in ounces in different market areas. Establishing the Ounces attribute dimension helps users to monitor which sizes sell better in which markets.

Use these sections to understand the basics of dimension, member, and member storage properties, and how to design an outline for performance:

- "Building an Outline" on page 120
- "Defining Dimension and Member Properties" on page 121
- "Defining Member Storage Properties" on page 122
- "Checklist to Define Dimension and Member Properties" on page 124
- "Designing an Outline to Optimize Performance" on page 124

Building an Outline

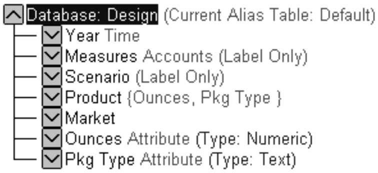
An outline includes more than dimensions and members. Dimensions and members have specific properties that provide access to built-in functionality. The properties of dimensions and members define the roles of the dimensions and members in the design of the multidimensional structure. These properties include the following:

- Dimension types
- Generation and level names
- Aliases
- Properties
- Field types
- Consolidation operators
- Formulas
- Attribute associations

Defining Dimension and Member Properties

In the outline, the TBC database dimensions and members have special tags. Essbase calls these tags *properties*:

Figure 37: TBC Dimensions and Related Properties



```
Database: Design (Current Alias Table: Default)
├── Year Time
├── Measures Accounts (Label Only)
├── Scenario (Label Only)
├── Product {Ounces, Pkg Type }
├── Market
├── Ounces Attribute (Type: Numeric)
└── Pkg Type Attribute (Type: Text)
```

- The Year dimension is tagged as time.
- The Measures dimension is tagged as accounts and label only.
- The Scenario dimension is tagged as label only.
- Ounces and Pkg Type are attribute dimensions. The Products dimension is associated with two attribute dimensions, Ounces and Pkg Type.

A dimension type is a property that Essbase provides that adds special functionality to a dimension. The most commonly used dimension types are time, accounts, and attribute. Table 7 defines each of the Essbase dimension types.

Table 7: Dimension Types

Dimension Types	Description
None	Specifies no particular dimension type.
Time	Defines the time period for which you report and update data. You can tag only one dimension as time. The time dimension enables several accounts dimension functions, such as first and last time balances.
Accounts	Contains items that you want to measure, such as profit and inventory and makes Essbase built-in accounting functionality available. (For example, see "Accounts Dimension Calculation" on page 5-131.) Only one dimension can be defined as accounts.

5

Table 7: Dimension Types (Continued)

Dimension Types	Description
Attribute	Contains members that can be used to classify members of another, associated dimension. For example, the Pkg Type attribute dimension contains a member for each type of packaging, such as bottle or can, that applies to members of the Product dimension.
Country	Contains data about where business activities take place. In a country dimension, you can specify the type of currency used in each member. For example, Canada has three markets: Vancouver, Toronto, and Montreal. They use the same currency type, Canadian dollars.
Currency partition	Separates local currency members from a base currency defined in the application. This dimension type is used only in the main database and is used for currency conversion applications. If the base currency for analysis is US dollars, the local currency members contain values that are based on the currency type of the region. See Chapter 12, "Designing and Building Currency Conversion Applications."

When you define members in standard dimensions, Essbase automatically tags the members with the addition (+) consolidation, meaning that during consolidation members are added. For example, Jan, Feb, and Mar figures are added, and the result stored in the parent, Qtr1.

As appropriate, you can change a member consolidation property to one of the following operators: +, -, *, /, %, and ~ (no consolidation). For more information on consolidation properties, see "Consolidation Paths" on page 5-128.

Defining Member Storage Properties

With Essbase, you can specify data storage properties for members; these properties define where and when consolidations are stored. For example, by default, members are tagged as store data. Essbase sums store data members and stores the result at the parent level. You can change the default logic for each member by changing the data storage property tag for the member. Members with the label only tag, for example, do not have data associated with them. Members with the label only tag exist only for purposes of data grouping or navigation. Because members of this type contain no data, they cannot be consolidated.

For example, in the Measures dimension, the member Ratios has three children, Margin%, Profit%, and Profit per Ounce. The member Ratios defines a category of members. When consolidated, Margin%, Profit%, and Profit per Ounce do not roll up to a meaningful figure for Ratios. Hence, Ratios is tagged as label only.

Table 8 describes Essbase data storage properties.

Table 8: Essbase Data Storage Properties

Storage Properties	Effects on Members
Store data	The member stores data. Store data is the default storage property.
Dynamic Calc	The data associated with the member is not calculated until requested by a user. The calculated data is not stored; it is discarded after the request is completed.
Dynamic Calc and Store	The data associated with the member is not calculated until it is requested by a user. The calculated data is then stored.
Shared member	The data associated with the member comes from another member with the same name.
Never share	The data associated with the member is duplicated with its parent or child if an implied shared relationship exists.
Label only	Although a label only member has no data associated with it, it can still display a value. The label only tag groups members and eases navigation and reporting. Typically, label only members have a no consolidation property. (See "Consolidation Ordering" on page 5-129.)

Attribute dimensions and members are tagged as Dynamic Calc. Attribute dimensions and members have no data values of their own. The data values you see associated with attribute dimensions and members are dynamically calculated from the associated base dimension members. You cannot change the storage property for attribute dimensions and members.

For more details on the types of properties you can assign to dimensions and members and for instructions for creating the components of a database, see the following chapters:

- Chapter 6, “Creating Applications and Databases”
- Chapter 7, “Creating and Changing Database Outlines”
- Chapter 8, “Setting Dimension and Member Properties”
- Chapter 9, “Working with Attributes”

Checklist to Define Dimension and Member Properties

- Can you identify a time dimension?
- Can you identify an accounts dimension?
- Does the data include foreign currencies? If so, did you identify a currency partition dimension? See Chapter 12, “Designing and Building Currency Conversion Applications.”
- Can you identify qualities or characteristics of dimensions that should be defined as separate attribute dimensions?
- What members require special data storage properties?

Designing an Outline to Optimize Performance

When you design an outline, you must position attribute dimensions at the end of the outline. You should position dense dimensions before sparse dimensions.

The position of dimensions in an outline and the storage properties of dimensions can affect two areas of performance: how quickly calculations are run and how long it takes users to retrieve information.

Use these sections to understand performance optimization basics:

- “Optimizing Query Performance” on page 125
- “Optimizing Calculation Performance” on page 125
- “Meeting the Needs of Both Calculation and Retrieval” on page 126

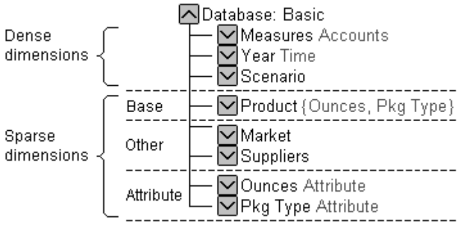
Optimizing Query Performance

To optimize query performance, use the following guidelines when you design an outline:

- If the outline contains attribute dimensions, make sure that the attribute dimensions are the only sparse Dynamic Calc dimensions in the outline.
- In the outline, place the most queried sparse dimensions before the less queried sparse dimensions.

The outline shown in Figure 38 is designed for optimum query performance:

Figure 38: Designing an Outline for Optimized Query Times



- Because the outline contains attribute dimensions, the storage property for the all of the standard dimensions and all of their members is set as store data.
- As the most-queried sparse dimension, the Product dimension is the first of the sparse dimensions. Base dimensions are typically queried more than other dimensions.

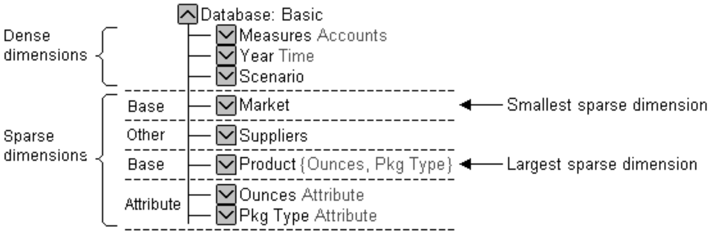
Optimizing Calculation Performance

To optimize calculation performance, order the sparse dimensions in the outline by their number of members, starting with the dimension that contains the fewest members.

For more information about factors that affect calculation performance, see "Designing for Calculation Performance" on page 1370.

The outline shown in Figure 39 is designed for optimum calculation performance:

Figure 39: Designing an Outline for Optimized Calculation Times



- The smallest standard dimension that is sparse, Market, is the first of the sparse dimensions in the outline.
- The largest standard dimension that is sparse, Product, is immediately above the first attribute dimension. If the outline did not contain attribute dimensions, the Product dimension would be at the end of the outline.

Meeting the Needs of Both Calculation and Retrieval

Even though they contain the same dimensions, the example outlines are different. To determine the best outline sequence for a situation, you must prioritize the data retrieval requirements of the users against the time needed to run calculations on the database. How often do you expect to update and recalculate the database? What is the nature of user queries? What is the expected volume of user queries?

A possible workaround is to position the dimensions in the outline initially to optimize calculations. After you run the calculations, you can manually resequence the dimensions to optimize retrieval. When you save the outline after you reposition its dimensions, choose to restructure the database by index only. Before you run calculations again, remember to resequence the dimensions in the outline to optimize calculations.

Loading Test Data

Before you can test calculations, consolidations, and reports, you need data in the database. During the design process, loading mocked-up data or a subset of actual business data provides flexibility and shortens the time it takes to test and analyze results.

Detailed instructions for loading data are in the following chapters:

- Chapter 20, "Introducing Data Loading"
- Chapter 21, "Setting up a Rules File to Manipulate Records"
- Chapter 22, "Manipulating Fields Using a Rules File"
- Chapter 23, "Performing and Debugging a Data Load"

When you are satisfied with a database design, test the loading of the complete set of real data with which you will populate the final database, using the test rules files if possible. This final test may reveal problems with the source data that you did not anticipate during earlier phases of the database design process.

Defining and Testing Calculations

Calculations are essential to derive certain types of data. Data that is derived from a calculation is called *calculated data*; basic noncalculated data is called *input data*.

This section uses the Product and Measures dimensions in TBC's application to illustrate several types of common calculations, Such calculations are found in many Essbase applications. You learn about the following:

- "Consolidation Paths" on page 128
- "Consolidation Ordering" on page 129
- "Shared Members and Consolidation" on page 130
- "Time and Accounts Calculations" on page 130
- "Accounts Dimension Calculation" on page 131
- "Formulas" on page 133
- "Dynamic Calculations" on page 135
- "Two-Pass Calculations" on page 136

Consolidation Paths

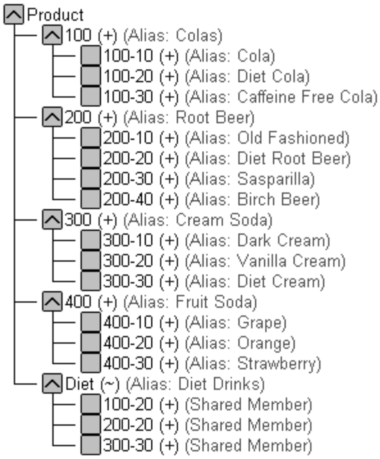
Consolidation is the most frequently used calculation in Essbase. This section uses the Product dimension to illustrate consolidations.

TBC's application has several consolidation paths:

- Individual products roll up to product families, and product families consolidate into Product. The TBC outline also requires multiple consolidation paths; some products must consolidate in multiple categories.
- States roll up to regions, and regions consolidate into Market.
- Months roll up into quarters, and quarters consolidate into Year.

Consolidation operators define how Essbase rolls up data for each member in a branch to the parent. Essbase gives members a default operator of addition (+), meaning that it adds the value of a member to the values of other members in the branch. For example, Essbase adds 100-10, 100-20, and 100-30 and stores the result in their parent, 100, as shown in Figure 40.

Figure 40: TBC Product Dimension



```
Product
    100 (+) (Alias: Colas)
        100-10 (+) (Alias: Cola)
        100-20 (+) (Alias: Diet Cola)
        100-30 (+) (Alias: Caffeine Free Cola)
    200 (+) (Alias: Root Beer)
        200-10 (+) (Alias: Old Fashioned)
        200-20 (+) (Alias: Diet Root Beer)
        200-30 (+) (Alias: Sasparilla)
        200-40 (+) (Alias: Birch Beer)
    300 (+) (Alias: Cream Soda)
        300-10 (+) (Alias: Dark Cream)
        300-20 (+) (Alias: Vanilla Cream)
        300-30 (+) (Alias: Diet Cream)
    400 (+) (Alias: Fruit Soda)
        400-10 (+) (Alias: Grape)
        400-20 (+) (Alias: Orange)
        400-30 (+) (Alias: Strawberry)
    Diet (~) (Alias: Diet Drinks)
        100-20 (+) (Shared Member)
        200-20 (+) (Shared Member)
        300-30 (+) (Shared Member)
```

The Product dimension contains mostly (+), operators, which indicate that each group of members is added together and rolled up to the parent. Diet has a tilde (~), which indicates that Essbase does not include the Diet member in the consolidation to the parent, Product. The Diet member consists entirely of members it shares or duplicates. The TBC product management group wants to be able to isolate Diet drinks in reports, so TBC created a separate Diet member that does not impact the overall consolidation.

For more information on consolidation and calculations, see Chapter 8, "Setting Dimension and Member Properties."

Consolidation Ordering

Essbase calculates the data in a branch in top-down order. For example, if you have, in order, two members tagged with an addition symbol (+) and a third member tagged with a multiplication symbol (*). Essbase adds the first two and multiplies the sum by the third. Be aware that Essbase always begins with the top member when it consolidates, so the order and the labels of the members is very important.

Table 9 shows the Essbase consolidation operators. For more information, see "Introducing Member Consolidation Properties" on page 8-208.

Table 9: Consolidation Operations

Operator	Description
+	The default operator. When a member has the + operator, Essbase adds that member to the result of previous calculations performed on other members.
–	When a member has the – operator, Essbase multiplies the member by -1 and then adds the product to the result of previous calculations performed on other members.
*	When a member has the * operator, Essbase multiplies the member by the result of previous calculations performed on other members.
/	When a member has the / operator, Essbase divides the member into the result of previous calculations performed on other members.
%	When a member has the % operator, Essbase divides the member into the sum of previous calculations performed on other members. The result is multiplied by 100.
~	When a member has the ~ operator, Essbase does not use it in the consolidation to its parent.

Shared Members and Consolidation

Shared members also affect consolidation paths. The shared member concept lets two members with the same name share the same data. The shared member stores a pointer to data contained in the other member, so Essbase only stores the data once. Shared members must be in the same dimension. You can share data with two or more members.

Use this checklist to help define consolidations:

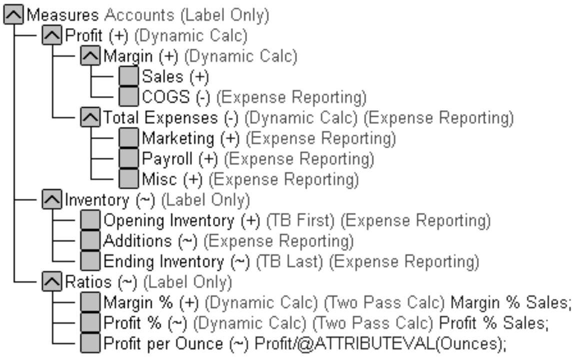
- Have you identified the consolidations in the outline?
- Did you tag each member with the proper consolidation operator?
- Did you specify a shared member tag for designated members?
- Would shared members be more efficient if designed as an attribute dimension?

Time and Accounts Calculations

The Measures dimension is the most complex dimension in the TBC outline because it uses both time and accounts data. It also contains several formulas and special tags to help Essbase calculate the outline. This section discusses the formulas and tags that TBC included in the Measures dimension (the dimension tagged as accounts).

Take a moment to look closely at the Measures dimension tags defined by TBC (in Figure 41). You see that you already know about many of the properties in the Measures dimension. You have learned about positive (+), negative (–), and tilde (~) consolidation operators, as well as accounts and label only tags:

Figure 41: TBC Measures Dimension



```
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);
```

- The Inventory and Ratios member names assist the user in data navigation and do not contain data and, therefore, receive a label only tag.
- The Measures dimension itself also has a label only tag. Some members of Measures have a Dynamic Calc tag.
- Dynamic calculations are discussed in "Dynamic Calculations" on page 135.
- Some members of Measures have a time balance tag (TB First or TB Last). Time balance tags are discussed in "Introducing Time Balance Properties" on page 196.

For details on Essbase calculations, see these sections:

- Chapter 24, "Introduction to Database Calculations" through Chapter 33, "Developing Custom-Defined Calculation Functions,"
- Chapter 52, "Optimizing with Intelligent Calculation," and Chapter 52, "Optimizing with Intelligent Calculation."

Accounts Dimension Calculation

This section discusses two forms of calculations for a dimension tagged as accounts:.

- "Time Balance Properties" on page 131
- "Variance Reporting" on page 133

Time Balance Properties

Note the two tags in the Measures dimension: TB first and TB last. These tags, called time balance tags or properties, provide instructions to Essbase about how to calculate the data in a dimension tagged as accounts. To use these tags, you must have a dimension tagged as accounts and a dimension tagged as time. The first, last, average, and expense tags are available exclusively for use with accounts dimension members.

In the TBC Measures dimension, Opening Inventory data represents the inventory that TBC carries at the beginning of each month. The quarterly value for Opening Inventory is equal to the Opening value for the quarter. Opening Inventory requires the time balance tag, TB first.

Ending Inventory data represents the inventory that TBC carries at the end of each month. The quarterly value for Ending Inventory is equal to the ending value for the quarter. Ending Inventory requires the time balance tag, TB last. Table 10 shows the time balance tags for the accounts dimension.

Table 10: Accounts Member Tags

Tags	Description
Time Balance Last	The value for the last child member is carried to the parent. For example, March is carried up to Qtr1.
Time Balance First	The value for the first child is carried to the parent. For example, Jan is carried up to Qtr1.

Table 11 shows how consolidation in the time dimension is affected by time balance properties in the accounts dimension, showing details for the first quarter only for clarity:

Table 11: TBC Consolidations Affected by Time Balance Properties

Accounts -> Time	Jan	Feb	Mar	Qtr1	Year
Accounts Member1	11	12	13	36	Qtr1 + Qtr2 + Qtr3 + Qtr4
Accounts Member2 (TB First)	20	25	21	20	20
Accounts Member3 (TB Last)	25	21	30	30	Value of Qtr4

Normally, the calculation of a parent in the time dimension is based on the consolidation and formulas of its children. However, if a member in an accounts branch is marked as TB First, then any parent in the time dimension matches the member marked as TB First.

For more information on time balance tags, see “Introducing Time Balance Properties” on page 8-196.

Variance Reporting

One of TBC's Essbase requirements is the ability to perform variance reporting on actual versus budget data. The variance reporting calculation requires that any item that represents an expense to the company must have an expense reporting tag. Inventory members, Total Expense members, and the COGS member each receive an expense reporting tag for variance reporting.

Essbase provides two variance reporting properties: expense and non-expense. The default is non-expense. Variance reporting properties define how Essbase calculates the difference between actual and budget data in members with the @VAR or @VARPER function in their member formulas.

When you tag a member as expense, the @VAR function calculates Budget - Actual. For example, if the budgeted amount is $100 and the actual amount is $110, the variance is -10.

Without the expense reporting tag, the @VAR function calculates Actual - Budget. For example, if the budgeted amount is $100 and the actual amount is $110, the variance is 10.

Formulas

You can define formulas to calculate relationships between members in the database outline. You can either apply the formulas to members in the outline, or you can place the formulas in a calc script. In this section, you learn how TBC optimized the performance of its database by using formulas.

Functions are predefined routines that perform specialized calculations and return sets of members or sets of data values. Formulas are composed of operators and functions, as well as dimension names, member names, and numeric constants.

The operators include the following:

- Mathematical operators that perform arithmetic operations
- Conditional operators that build logical conditions into calculations
- Cross-dimensional operators that point to data values of specific database member combinations

The Essbase functions include over 100 predefined routines to extend the calculation capabilities of Essbase. The main functions include the following:

- Boolean functions, which provide a conditional test by returning a TRUE or FALSE value
- Mathematical functions, which perform specialized mathematical calculations
- Relationship functions, which look up data values within a database during a calculation based on the current member's position
- Range functions, which declare a range of members as an argument to another function or command
- Financial functions, which perform specialized financial calculations
- Member set functions, which generate lists of members based on a specified member
- Allocation functions, which allocate values that are input at a parent level across child members
- Forecasting functions, which manipulate data for the purposes of smoothing data, interpolating data, or calculating future values
- Statistical functions, which calculate advanced statistics
- Date and time functions, which use date and time characteristics in calculation formulas
- Calculation mode functions, which specifies the calculation mode that Essbase uses to calculate a formula

The Measures dimension uses the following formulas:

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

Essbase uses consolidation operators to calculate the Margin, Total Expenses, and Profit members. The Margin% formula uses a % operator, which means "express Margin as a percentage of Sales." The Profit% formula uses the same % operator. The Profit per Ounce formula uses a division operator (/) and a function (@ATTRIBUTEVAL) to calculate profitability by ounce for products sized in ounces.

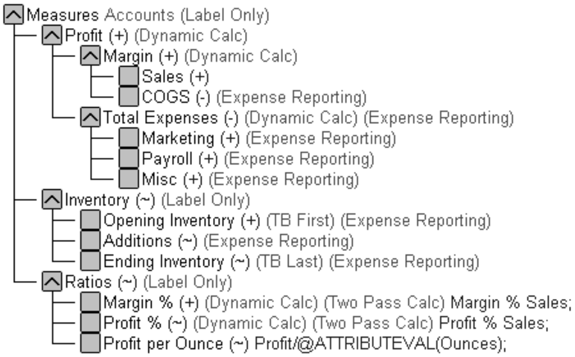
For a complete list of operators, functions, and syntax, see the *Technical Reference* in the `docs` directory. For a discussion of how to use formulas, see Chapter 25, "Developing Formulas."

Dynamic Calculations

When you design the overall database calculation, you may want to define a member as a Dynamic Calc member. When you tag a member as Dynamic Calc, Essbase calculates the combinations of that member when you retrieve the data, instead of pre-calculating the member combinations during the regular database calculation. Dynamic calculations shorten regular database calculation time, but may increase retrieval time for dynamically calculated data values.

As shown in Figure 42, TBC's Measures dimension contains several members that are tagged as Dynamic Calc: Profit, Margin, Total Expenses, Margin %, and Profit %.

Figure 42: TBC Measures Dimension, Dynamic Calc Tags



```
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);
```

When an overall database calculation is performed, these members and their corresponding formulas are not calculated. Rather, these members are calculated when a user requests them, for example, from Spreadsheet Add-in. Essbase does not store the calculated values; it recalculates the values for any subsequent retrieval. However, you can choose to store dynamically calculated values after the first retrieval. For more information, see "Understanding Dynamic Calculations" on page 770.

To decide when to calculate data values dynamically, consider your needs for the following:

- Optimum regular calculation time (batch calculation)
- Low disk space usage
- Reduced database restructure time
- Speedy data retrieval for users
- Reduced backup time

For more information about dynamic calculations, see Chapter 28, "Dynamically Calculating Data Values."

Two-Pass Calculations

In the TBC database, both Margin % and Profit % contain the label two-pass. This default label indicates that some member formulas need to be calculated twice to produce the desired value. The two-pass property works only on members from the dimension tagged as accounts or on Dynamic Calc and Dynamic Calc And Store members. The following examples illustrate why Profit % has a two-pass tag.

Essbase loads data into the system as follows:

Measures -> Year	**Jan**	**Feb**	**Mar**	**Qtr1**
Profit	100	100	100	
Sales	1000	1000	1000	
Profit % Profit % Sales (Two-Pass Calc)				

Essbase calculates Measures first. The data then looks like this:

Measures -> Year	Jan	Feb	Mar	Qtr1
Profit	100	100	100	
Sales	1000	1000	1000	
Profit % Profit % Sales (Two-Pass Calc)	10%	10%	10%	

Next, Essbase calculates the Year dimension. The data rolls up across the dimension.

Measures -> Year	Jan	Feb	Mar	Qtr1
Profit	100	100	100	300
Sales	1000	1000	1000	3000
Profit% Profit % Sales (Two-Pass Calc)	10%	10%	10%	30%

The result in Profit % -> Qtr1 of 30% is not correct. However, because TBC tagged Profit% as two-pass calculation, Essbase recalculates profit percent at each occurrence of the member Profit %. The data is then correct and is displayed as follows:

Measures -> Year	Jan	Feb	Mar	Qtr1
Profit	100	100	100	300
Sales	1000	1000	1000	3000
Profit % Profit % Sales (Two-Pass Calc)	10%	10%	10%	10%

Use this checklist when you define a calculations

- Which dimensions require formulas or calculations?
- Does the default calculation logic achieve accurate results?
- What type of calculations are required?
- Which members require variance reporting?
- Do members that require variance reporting have a time or accounts tag?

Defining Reports

To be sure the design meets user information requirements, you need to view data as users view it. Users typically view data through spreadsheets, printed reports, or reports published on the Web. There are many tools available through Hyperion and Hyperion partners for producing the reporting systems that users use.

Essbase provides several tools that can help you during the design process to display and format the data quickly and test whether the database design meets user needs. You can use the Report Writer of Application Manager or the Essbase Administration Services Console Report Script Editor to write report scripts quickly. Those familiar with spreadsheets can use the spreadsheet add-ins for Microsoft Excel and Lotus 1-2-3. See Chapter 34, “Quick Start to Report Scripts” and the appropriate spreadsheet user’s guide.

During the design phase, you should check for such things as the following:

- Grouping and sequencing of data. Do the intersections enabled by the design provide the data that users need?
- Levels of totals. In spreadsheets this term refers to the consolidation levels that are available to a user who drills down and up through the hierarchy of the outline design.
- Attribute reporting. Does the database design facilitate an analysis that is based on the characteristics or attributes of specific dimensions or members? For example, do you need to compare sales by specific combinations of size and packaging, such as comparing the sales of 16-ounce bottled colas with the sales of 32-ounce bottled colas? How can you take advantage of the Attribute Calculations dimension that Essbase provides for use with attribute dimensions? For more information about attributes, see Chapter 9, “Working with Attributes.”

If you provide predesigned reports for users, this is the time to use the appropriate tool to create those reports against the test data. The reports that you design should provide the information that meets the original objectives of the application. The reports should be easy to use. They should provide the right combinations of data and the right amount of data. Reports with too many columns and rows are very hard to use. It may be better to create a number of different reports instead of one or two all-inclusive ones.

Verifying the Design

After you have analyzed the data and created a preliminary design, you need to check all aspects of the design with the users. You should have already checked to see if the database satisfies the users' analysis and reporting needs. Make sure you check with the users to ensure that the database satisfies all goals of the application.

Near the end of the design cycle, you need to test with real data. Did the outline build correctly? Did all the data load? If the database fails in any area, repeat the steps of the design cycle to identify the cause of the problem.

Essbase provides several sources of information to help you isolate problems. Sources include application and OLAP Server logs, exception logs, and the information displayed on the Database Information dialog box. This manual, *Essbase Database Administrator's Guide*, also provides helpful information. Look in sections relevant to the problem; for example, sections about security, calculations, or reports. You can also use the index of this guide to find help for solving problems. Look up such terms as troubleshooting, logs, optimizing, performance, recovery, resources, errors, and warnings.

Most likely, you will need to repeat one or more steps of the design process to arrive at the ideal database solution.

Chapter

6

Creating Applications and Databases

An Essbase application is a container for a database and its related files. This chapter provides an overview of Essbase applications and databases and explains how to create them. For information on everyday management of applications, databases, and their associated files, see Part IX, “Maintaining and Automating Hyperion Essbase.”

This chapter includes the following sections:

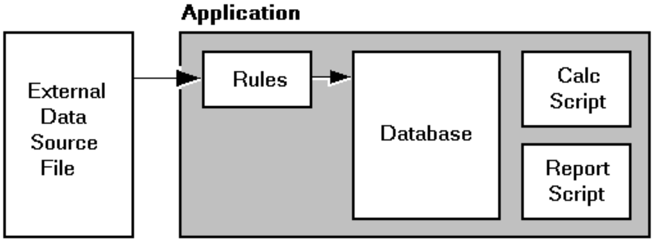
About Applications and Databases

An Essbase application is a management structure that contains one or more Essbase databases and related files. Essbase applications and databases usually reside on OLAP Server. The server machine can store multiple applications. Applications and databases created on client machines are used only to store database objects, such as outlines and calculation scripts. You cannot load data or calculate data on a client machine.

An Essbase database is a data repository that contains a multidimensional data storage array. A multidimensional database supports multiple views of data so that users can analyze the data and make meaningful business decisions. For more information about how Essbase stores data, see "Storage Allocation" on page 1115.

This diagram shows the relationships among the parts of an application:

Figure 43: Parts of an Essbase Application



With Application Manager, you can create and maintain Essbase applications. Application development includes creating databases, building database outlines, loading and calculating data, and defining security access. For information about everyday management of applications and databases, see Part IX, "Maintaining and Automating Hyperion Essbase."

Database Objects

Files that are related to databases are called *objects*. Database objects perform actions against one or more Essbase databases, such as defining calculations or reporting against data. By default, objects are stored in their associated database folder on the server. They can also be saved to a client machine or to other available network directories.

In Essbase, the common types of objects include the following:

- A database outline (a storage structure definition)
- Data sources
- Rules for loading data and building dimensions dynamically (“rules files”)
- Scripts that define how to calculate data (“calculation scripts”)
- Scripts that generate reports on data (“report scripts”)
- Security definitions
- Linked reporting objects
- Partition definitions

Some objects are optional, such as calculation scripts and linked reporting objects.

For a complete list of application and database file types, see “Application and Database File Types” on page 1185.

Database Outlines

Database outlines define the structure of a multidimensional database, including all the dimensions, members, aliases, tags, types, consolidations, and mathematical relationships. The structure defined in the outline determines how data is stored in the database.

When a database is created, Essbase creates an outline for that database automatically. The outline has the same name as the database (*dbname*.OTL). For example, when the Basic database is created within the Sample application, an outline is created within the following directory:

```
ARBORPATH\APP\SAMPLE\BASIC\BASIC.OTL
```

For information about creating outlines, see Chapter 7, “Creating and Changing Database Outlines.”

Data Sources

A data source is external data that is loaded into an Essbase database. The common types of data sources include the following:

- Text files
- Spreadsheet files
- External databases, such as an SQL database

For complete information about data sources, see Chapter 20, “Introducing Data Loading.”

You can integrate a relational database with a multidimensional database so that much of the data remains in the relational database. For complete information, see Appendix D, “Accessing Relational Data with Hybrid Analysis.”

Rules Files for Data Load and Dimension Build

An Essbase database contains no data when it is first created. *Data load rules files* are sets of operations that Essbase performs on data from an external data source file as it is loaded, or copied, into the Essbase database. Specifying data load rules is the most common way to load data into the database. Data load rules files are typically associated with a particular database, but you can define rules for use with multiple databases.

Dimension build rules files create or modify an outline dynamically based on data in an external data source file.

A single rules file can be used for both data loads and dimension builds. Rules files have the `.RUL` extension.

For information about creating data load rules files, see “Introducing Data Loading” on page 565. For information about creating dimension build rules files, see “Building Dimensions Using a Rules File” on page 533.

Calculation Scripts

Calculation scripts are text files that contain sets of instructions telling Essbase how to calculate data in the database. Calculation scripts, also called calculation scripts, perform different calculations than the consolidations and mathematical operations that are defined in the database outline. Because calculation scripts perform specific mathematical operations on members, they are typically associated with a particular database. You can, however, define a calculation script for use with multiple databases. Calculation scripts files have the `.CSC` extension.

For information about creating calculation scripts, see Chapter 30, "Developing Calculation Scripts."

Report Scripts

Report scripts are text files that contain data retrieval, formatting, and output instructions to create a report from the database. Report scripts are typically associated with a particular database, but you can define a report script for use with multiple databases. Report scripts have the `.REP` extension.

For information about creating report scripts, see Chapter 35, "Developing Report Scripts."

Security Definitions

Essbase provides a comprehensive system for managing access to applications, databases, and other objects. Each application and database contains its own security definitions that restrict user access.

For more information about setting up and maintaining security information, see Chapter 15, "Managing Security for Users and Applications."

Linked Reporting Objects

A linked reporting object is an object associated with a specific data cell in an Essbase database. Linked reporting objects can enhance data analysis capabilities by providing additional information on a data point.

A linked reporting object can be any of the following:

- A paragraph of descriptive text (a "cell note")
- A separate file that contains text, audio, video, or graphics
- A Uniform Resource Locator (URL) for a Web site
- A link to data in another Essbase database

For more information about using linked reporting objects, see Chapter 11, "Linking Objects to Essbase Data."

Spreadsheet Queries

Within Spreadsheet Add-in, users can create queries using Query Designer (EQD). Users can save the reports in the form of queries (`.EQD` files). The queries can be accessed at a later time by the user who created the report or by other users who have access to the query.

For more information, see the *Essbase Spreadsheet Add-in User's Guide*.

Member Select Definitions

With Spreadsheet Add-in, users can define member retrievals with the member select feature. If users want to save the member select specification, they can do so with an `.SEL` file.

For more information, see the *Essbase Spreadsheet Add-in User's Guide*.

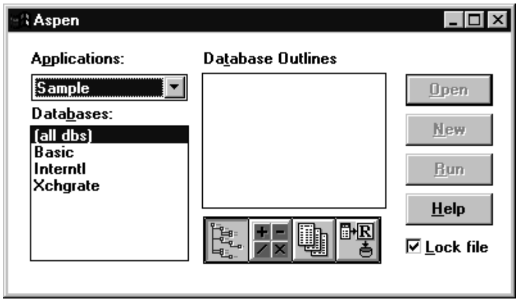
Process for Creating Databases

To implement a multidimensional database you need to complete a number of steps, from creating an application through optimizing your database performance. For a complete list of these steps, see Chapter 4, "Quick Start for Implementing Essbase."

Using the Application Desktop Window

When you start Application Manager, the Desktop opens and a closed window is displayed at the bottom of the Desktop. This is the Application Desktop window for client-based applications. When you connect to OLAP Server, a window is displayed that lists all the applications on the server. This is the Application Desktop window for server-based applications:

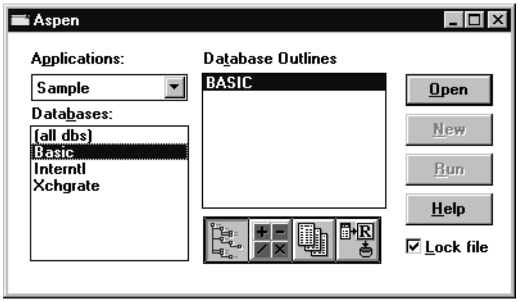
Figure 44: Application Desktop Window



➤ To work with database outlines, calculation scripts, report scripts, data loading or dimension building rules using the Application Dekstop, use this procedure:

1. Select the application name from the Applications list box. The Databases list box then displays the names of all the databases in the selected application.
2. Select a database name from the list box. The Application Desktop window is displayed:

Figure 45: Application Desktop Window with Database Selected



3. Select one of the following buttons to work with outlines, scripts, or rules files:

   The Outline button lists all database outlines for the selected database. This button performs no action if `(all dbs)` is selected.

   The Calc Script button lists all calculation scripts available for the selected application or database.

   The Report Script button lists all report scripts available for the selected application or database.

   The Data Load Rules button lists all rules files available for the selected application or database.

4. Do one of the following, depending on your task:

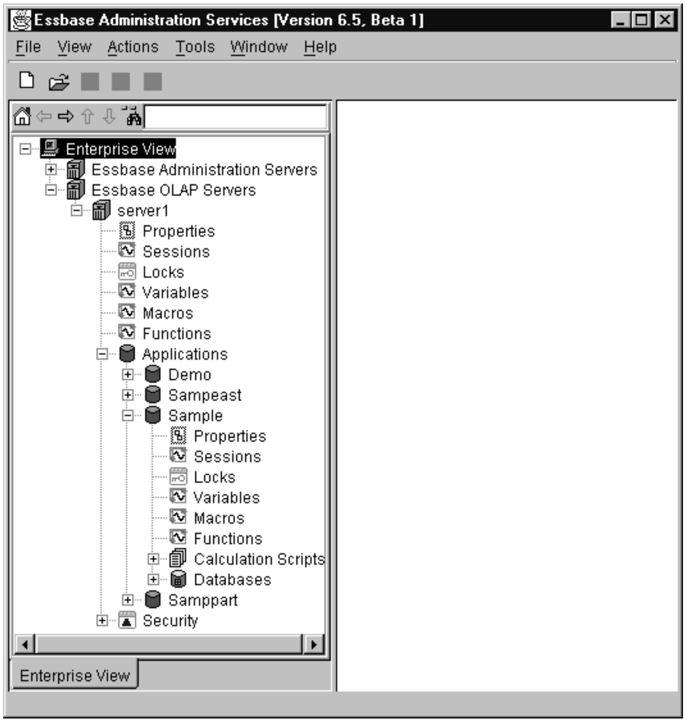
   - To open one of these files, select it from the list, and then select Open.
   - To create a new calculation script, report script, or rules file, select the appropriate button, and click New.
   - To create a new outline, see "Creating a New Database" on page 152.
   - To run a calculation script or report script, select the file from the list, and then select Run.
   - Click Help for details about this window.

   Note: When working on scripts associated with a particular database, make sure the database name is selected in the Databases list box. When working on scripts not associated with any particular database, make sure "(all dbs)" is selected in the Databases list box.

Using the Administration Services Console

You can also use Administration Services Console to manage applications and databases. The console displays a graphical view of the Essbase environment called Enterprise View as shown in Figure 46.

Figure 46: Administration Services Console Enterprise View



From this view, you can operate directly on applications and databases. For more information, see the *Essbase Administration Services Online Help*.

Deciding Where to Build an Application

Before creating an application and database, you should decide:

- Whether to build the application and database on OLAP Server, where other users can access them, or on your client machine.

  You can create an application on your client machine, or you can connect to OLAP Server and build it there. Building an application on the client allows you to develop and test it in an isolated environment. However, it does limit what can be done with the application.

  You must build your application on the server if you want to give other users access to the application or start the application automatically when the server starts.

- Whether to copy an existing application and database or to create a new one.
- What tools to use—Application Manager's graphical user interface, Essbase Administration Services Console, or ESSCMD commands.

Creating Applications and Databases

You can use either Application Manager, Essbase Administration Services Console, or ESSCMD to create a new application or database.

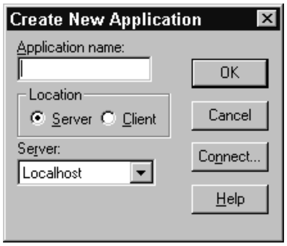
The application name will be created exactly as you enter it. If you enter the name as all capital letters, for instance NEWSAMP, Essbase will not automatically convert it to upper and lower case; for instance Newsamp.

Creating a New Application

➤ To create a new application with Application Manager:

1. Select File > New > Application. This dialog box is displayed:

   Figure 47: Create New Application Dialog Box

   Depending on your license agreement, you may see a Storage Type list box.

2. Type the name of the new application in the Application Name text box. This must be a unique name of 8 characters or less.

3. Select the client or server on which to create the application:

   - To create a client-based application, click Client. Essbase creates a subdirectory for the application in the `ARBORPATH\client` directory. The new subdirectory has the same name as the application.
   - To create a server-based application, click Server and select one of the servers listed in the Server list box. If the Server list box is empty, follow the directions for connecting to a server. Essbase creates a subdirectory for the application within the server's APP directory. The new subdirectory has the same name as the application.

Tip: You can also create applications without using Application Manager:

Tool	Instructions	For More Information
Administration Services	Use the Create Application dialog box	*Essbase Administration Services Online Help*
MaxL	**create application**	*Technical Reference* in the `docs` directory
ESSCMD	CREATEAPP	

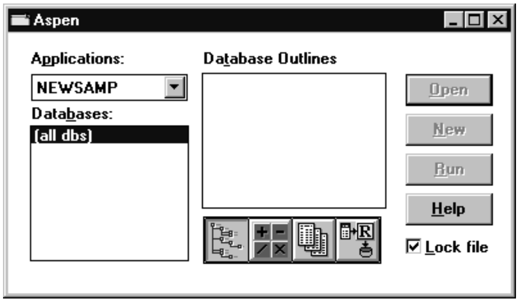
Creating a New Database

➤ To create a database using Application Manager:

1. Open the Application Desktop window:
   - For client-based applications, click the icon at the bottom of the Application Manager window.
   - For server-based applications, connect to the appropriate server.

   The Application Desktop window is displayed.

   Figure 48: Application Desktop Window

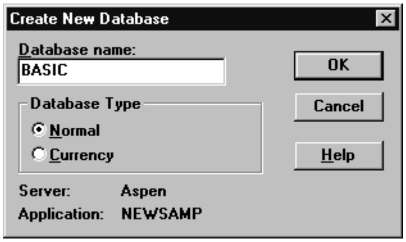


Applications: NEWSAMP
Databases: (all dbs)
☑ Lock file

2. From the Applications list box, select the name of the application in which you want to create the database.

3. Select File > New > Database. The following dialog box shown in Figure 49 is displayed.

   Figure 49: Create New Database Dialog Box



Database name: BASIC
Database Type
● Normal
○ Currency
Server: Aspen
Application: NEWSAMP

4. Type the name of the new database in the Database name text box. The Application Manager limits the database name to 8 characters.

   Note: The database name will be created exactly as you enter it. If you enter the name as all capital letters, for instance BASIC, Essbase will not automatically convert it to upper and lower case; for instance Basic.

   In most cases, the database type is Normal. To create a currency database, select a database type of Currency. For more information on currency databases, see Chapter 12, "Designing and Building Currency Conversion Applications."

5. Click OK to create the database. Essbase creates a subdirectory for the database within the application directory on the server or client. For information on the application directory, see "Creating Applications and Databases" on page 150. The new subdirectory has the same name as the application.

Tip: You can also create databases without using Application Manager:

Tool	Instructions	For More Information
Administration Services	Use the Create Database dialog box	*Essbase Administration Services Online Help*
MaxL	**create database**	*Technical Reference* in the `docs` directory
ESSCMD	CREATEDB	

Rules for Naming Applications and Databases

When naming applications and databases, follow these rules:

- The length must be 8 characters or fewer.
- Do not use spaces anywhere in the name.
- Do not use the following special characters anywhere in the name:

```
*   (asterisks)
\   (backslashes)
[]  (brackets)
:   (colons)
,   (commas)
=   (equal signs)
>   (greater than signs)
<   (less than signs)
.   (periods)
+   (plus signs)
?   (question marks)
"   (double quotation marks)
;   (semicolons)
`   (single quotation marks)
/   (forward slashes)
tabs
|   (vertical bars)
```

Annotating a Database

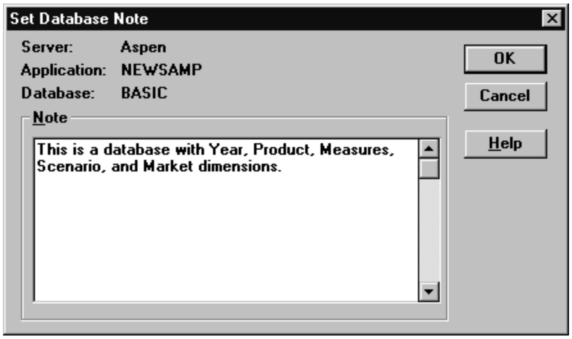
When you have created a database, you should annotate it. A database note can provide useful information in situations where you need to broadcast messages to users about the status of a database, deadlines for updates, and so on.

Spreadsheet Add-in users can view the database note from Spreadsheet Add-in. In Excel, for example, the Note button in the Connect dialog box lets you view database information.

➤ To annotate a database using Application Manager:

1. Select the database you want to annotate from the Application Desktop window.
2. Select Database > Set Note. This dialog box is displayed:

Figure 50: Set Database Note Dialog Box



Set Database Note

Server: Aspen
Application: NEWSAMP
Database: BASIC

Note

This is a database with Year, Product, Measures, Scenario, and Market dimensions.

3. Type some text in the Note text box.
4. Click OK.

You can also use the Set Database Note dialog box in Administration Services to annotate a database. Refer to the *Essbase Administration Services Online Help*.

Using Dynamic Calculations

When designing and creating your Essbase database, you might want to make use of dynamic calculations. Dynamically calculating some data values in your database can significantly improve the overall calculation performance of your database. For more information, see Chapter 28, “Dynamically Calculating Data Values” and Chapter 29, “Calculating Time Series Data.”

Using Substitution Variables

When designing your Essbase application, you may want to make use of *substitution variables*. Substitution variables act as global placeholders for information that changes regularly; each variable has a value assigned to it. The value can be changed at any time by the database designer; thus, manual changes are reduced.

For example, many reports depend on reporting periods; if you generate a report based on the current month, you have to update the report script manually every month. With a substitution variable, such as `CurMnth`, set on the server, you can change the assigned value each month to the appropriate time period. When you use the variable name in your report script, the information is dynamically updated when you run the final report.

You can use substitution variables in these components:

- Calculation scripts. For more information, see Chapter 30, “Developing Calculation Scripts.”
- Report Writer. For more information, see Chapter 35, “Developing Report Scripts.”
- Spreadsheet Add-in. For more information, see the *Essbase Spreadsheet Add-in User’s Guide*.
- SQL Interface. For more information, see the *Essbase SQL Interface Guide*.

You cannot use substitution variables in formulas that you apply to the database outline.

You can set substitution variables on the server using Application Manager, Administration Services, MaxL, or ESSCMD. Set the variable at any of the following levels:

- Server level provides access to the variable from all applications and databases on the server.
- Application level provides access to the variable from all databases within the application.
- Database level provides access to the variable within the specified database.

Guidelines for Setting Substitution Variables

Keep in mind the following guidelines when setting substitution variables:

- The substitution variable name must be composed of alphanumeric characters or underscores (_) and cannot the limit specified in Appendix A, "Limits."
- The substitution variable name cannot include non-alphanumeric characters, such as hyphens (-), asterisks (*), and slashes (/).
- The value of the substitution variable cannot exceed 256 characters. You can use any combination of characters in the value name. The value may contain any character except the leading ampersand (&).
- If the value of the substitution variable is numeric, you must enclose it in quotes. For example, if the variable name is Month and its corresponding value is 01 (corresponding to January), place quotes around 01 ("01").

Setting a Substitution Variable

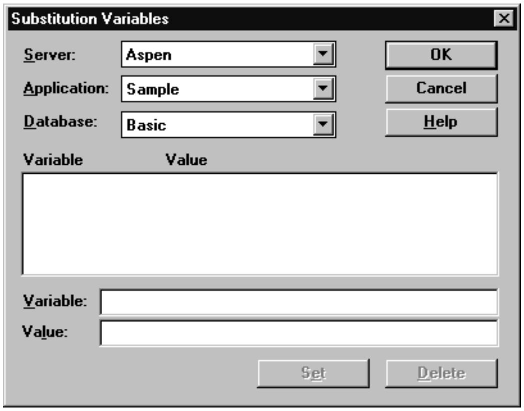
You can set substitution variables on the server at the server, application, or database level.

➤ To set a substitution variable using Application Manager:

1. From the Application Desktop window, select Server > Substitution Variables.

   Essbase displays the Substitution Variables dialog box, as shown in Figure 51.

   Figure 51: Creating a Substitution Variable

   Server: Aspen
   Application: Sample
   Database: Basic

2. In the Server list box, select the server to apply the variable to.
3. In the Application list box, select the application to apply the variable to. Select “(all apps)” to apply the variable to all applications on the server.
4. In the Database list box, select the database to apply the variable to. Select “(all dbs)” to apply the variable to all databases in the selected application.
5. In the Variable box, type the new variable name. See “Guidelines for Setting Substitution Variables” on page 157.
6. In the Value box, type the value name. See “Guidelines for Setting Substitution Variables” on page 157.
7. Click Set to apply the new variable and value.

Tip: You can also set substitution variables without using Application Manager:

Tool	Instructions	For More Information
Administration Services	Use the Substitution Variables window	*Essbase Administration Services Online Help*
MaxL	**alter system** **alter application** **alter database**	*Technical Reference* in the `docs` directory
ESSCMD	CREATEVARIABLE	

Deleting a Substitution Variable

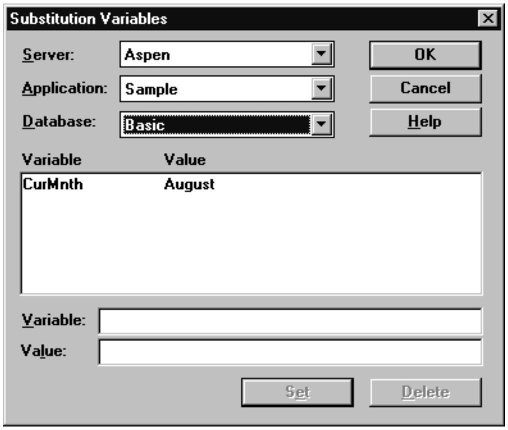
You may need to delete a substitution variable that is no longer used.

➤ To delete a substitution variable using Application Manager:

1. From the Application Desktop window, select Server > Substitution Variables.

   Essbase displays the Substitution Variables dialog box:

   Figure 52: Deleting a Substitution Variable



Substitution Variables

Server: Aspen

Application: Sample

Database: Basic

Variable	Value
CurMnth	August

Variable:

Value:

2. In the Server list box, select the server to delete the variable from.

3. In the Application list box, select the application to delete the variable from. If the variable applies to all applications on the server, select “(all apps).”
4. In the Database list box, select the database to delete the variable from. If the variable applies to all databases on the server, select “(all dbs).”
5. In the Variable list, select the variable you want to delete.
6. Click Delete to delete the variable and value.

Tip: You can also delete substitution variables without using Application Manager:

Tool	Instructions	For More Information
Administration Services	Use the Substitution Variables window	*Essbase Administration Services Online Help*
MaxL	**alter system** **alter application** **alter database**	*Technical Reference* in the `docs` directory
ESSCMD	DELETEVARIABLE	

Updating a Substitution Variable

You can modify or update existing substitution variables.

➤ To update a substitution variable using Application Manager:

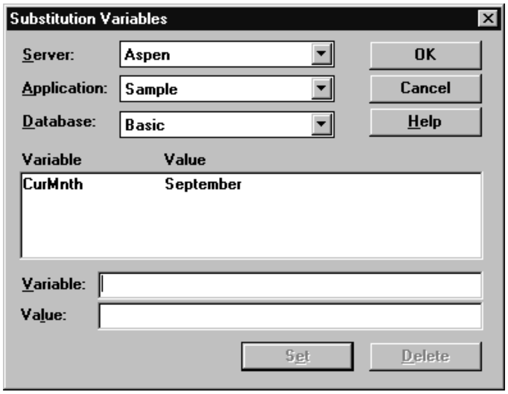
1. From the Application Desktop window, select Server > Substitution Variables.
2. In the Server list box, select the server where the variable is set.
3. In the Application list box, select the application where the variable is set. If the variable applies to all applications on the server, select “(all apps).”
4. In the Database list box, select the database where the variable is set. If the variable applies to all databases in the server, select “(all apps).”

   The name of the existing variable and its current value are displayed in the list.
5. In the Variable box, type the name of the existing variable exactly as it is displayed in the Variable list.

6. In the Value box, type the value name. See "Guidelines for Setting Substitution Variables" on page 157.
7. Click Set to apply the new value to the existing variable.

   In the Value list, the previous value is replaced with the new value:

Figure 53: Updating a Substitution Variable



Substitution Variables

Server: Aspen

Application: Sample

Database: Basic

Variable	Value
CurMnth	September

Variable:

Value:

Tip: You can also update substitution variables without using Application Manager:

Tool	Instructions	For More Information
Administration Services	Use the Substitution Variables window	*Essbase Administration Services Online Help*
MaxL	**alter system** **alter application** **alter database**	*Technical Reference* in the `docs` directory
ESSCMD	UPDATEVARIABLE	

Using Location Aliases

A location alias is a descriptor for a data source. A location alias maps an alias name for a database to the location of that database. A location alias is set at the database level and specifies an alias, a server, an application, a database, a username, and a password. You need database designer privileges to set location aliases.

After you create a location alias, you can use the alias to refer to that database. If the location of the database changes, you can edit the location definition accordingly.

Note: You can use location aliases only with the @XREF function. With this function, you can retrieve a data value from another database to include in a calculation on the current database. In this case, the location alias points to the database from which the value is to be retrieved. For more information on @XREF, see the *Technical Reference* in the `docs` directory.

Creating Location Aliases

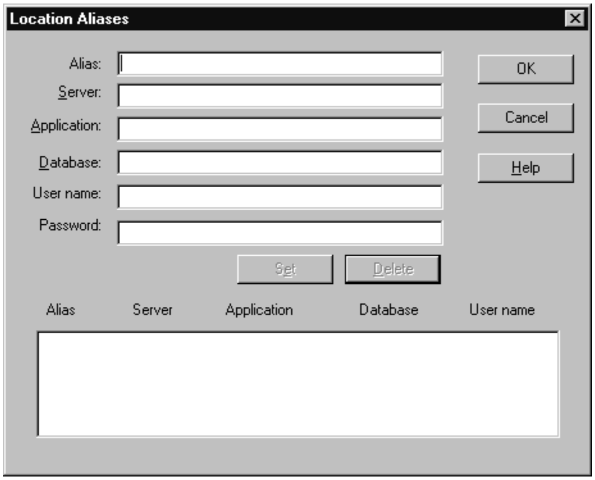
You can create a location alias for a particular database.

➤ To create a location alias using Application Manager:

1. Click the Application Desktop window.
2. Select Database > Location Aliases.

   Essbase displays the Location Aliases dialog box.

Figure 54: Location Aliases Dialog Box

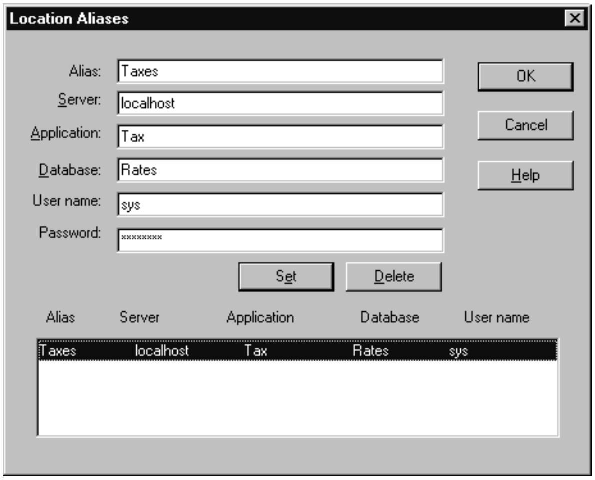


3. Enter the appropriate information in the following text boxes:
   - In the Alias text box, enter the alias, or alternate name, for the remote database.
   - In the Server text box, enter either the name or the IP address for the server that contains the remote database.
   - In the Application text box, enter the name of the application.
   - In the Database text box, enter the name of the database.
   - In the User name text box, enter the user name.
   - In the Password text box, enter the password.

4. Click Set to apply the settings.

   Essbase displays the settings in the lower text box.

Figure 55: Creating a Location Alias



Location Aliases

Alias: Taxes
Server: localhost
Application: Tax
Database: Rates
User name: sys
Password: ********

Alias	Server	Application	Database	User name
Taxes	localhost	Tax	Rates	sys

5. Click OK.

Tip: You can also create location aliases without using Application Manager:

Tool	Instructions	For more information
MaxL	**create location alias**	*Technical Reference* in the `docs` directory
ESSCMD	CREATELOCATION	

Editing or Deleting Location Aliases

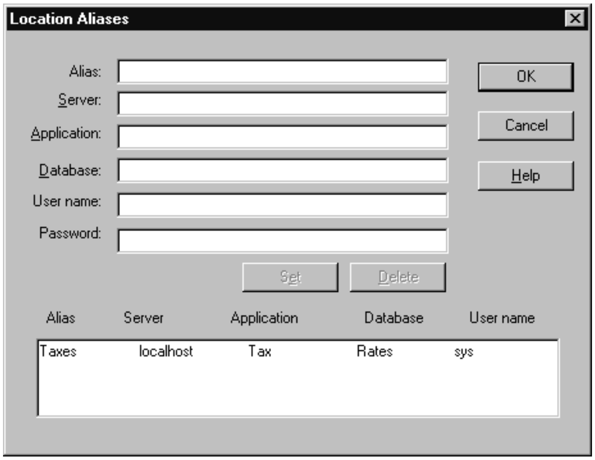
You can edit or delete location aliases that you previously created.

➤ To edit or delete a location alias using Application Manager:

1. Click the Application Desktop window.
2. Select Database > Location Aliases.

   Essbase displays the Location Aliases dialog box.

Figure 56: Editing or Deleting a Location Alias



Alias	Server	Application	Database	User name
Taxes	localhost	Tax	Rates	sys

3. From the lower text box, select the location alias that you want to edit, and follow one of these steps:
   - To edit the location alias, edit the text in the text boxes.
   - To delete the entire location alias, click **Delete**.
4. Click **OK**.

Tip: You can also edit or delete location aliases without using Application Manager:

Tool	Instructions	For More Information
MaxL	**display location alias** **drop location alias**	*Technical Reference* in the `docs` directory
ESSCMD	LISTLOCATIONS DELETELOCATION	

6

Chapter

7

Creating and Changing Database Outlines

A database outline contains dimensions and members organized in a hierarchy.

This chapter explains how to create an Essbase database outline using the Outline Editor. You can also change outlines using data sources and rules files. For more information, see Chapter 18, “Introducing Dynamic Dimension Building.” All examples in this chapter are based on the Sample Basic database shipped with Essbase.

This chapter contains the following sections:

For basic information about outlines, see Chapter 2, “Multidimensional Concepts.”

For information on setting properties in an outline, see Chapter 8, “Setting Dimension and Member Properties.”

For information about optimizing restructuring when outlines change, see Chapter 49, “Optimizing Database Restructuring.”

For information on using the Outline Editor with Hybrid Analysis, see Appendix D, “Accessing Relational Data with Hybrid Analysis”

Creating Outlines

The database outline defines the structure of the database. The Outline Editor displays the dimension hierarchy of an outline visually. Each dimension and consolidated level in a database appears in a collapsible tree diagram. The branches immediately below the database name are the dimensions, and the branches below a dimension name are members. For more information about outlines, see "Arranging Dimensions into Hierarchies" on page 61.

➤ To create an outline:

1. Create a new database. See "Creating a New Database" on page 152.
2. Open the new, blank outline. See "Opening Outlines" on page 168.
3. Add dimensions and members to the outline using one of the following means:
   - Manually add the dimensions and members. See "Adding Dimensions and Members to Outlines" on page 179.
   - Copy an existing outline. See "Copying Outlines" on page 170.
4. Verify and save the outline. See "Verifying and Saving Outlines" on page 172.

CAUTION: If you open the same outline with two instances of the Application Manager using the same login ID, each Save will overwrite the changes of the other instance. Because it can be difficult to keep track of what changes are saved or overwritten, we do not recommend this practice.

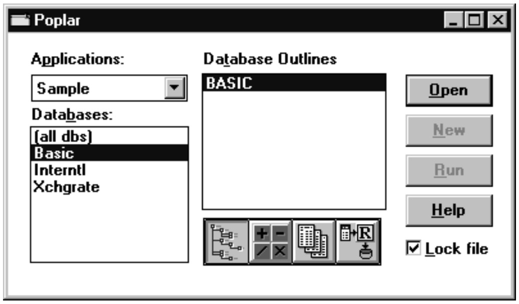
Opening Outlines

When you open an outline in the Outline Editor, Essbase loads the outline into memory on your client machine and you can view and manipulate its dimensions and members graphically.

➤ To open an outline:

1. In the Application Desktop window, select the application to view from the Applications list box; for example, the Sample application.
2. Select the database from the Databases list; for example, Basic:

Figure 57: Application Desktop Window



Poplar

Applications: Sample

Databases:
(all dbs)
Basic
Interntl
Xchgrate

Database Outlines
BASIC

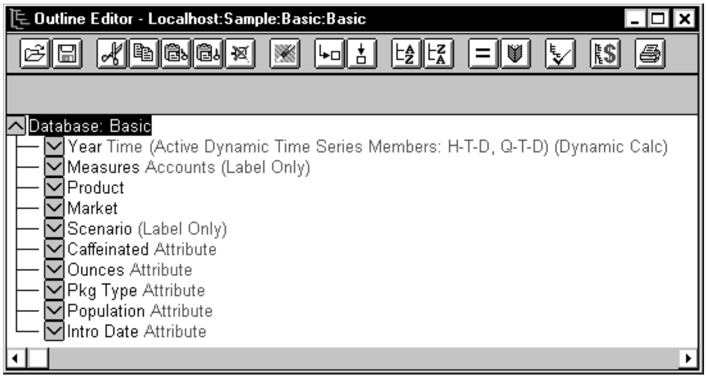
☑ Lock file

If Lock file is selected, other users cannot edit the outline until you close the outline again.

3. Click the Outline button, .
4. Click Open to open the selected outline.

Figure 58 shows the Sample Basic outline in the Outline Editor.

Figure 58: Outline Editor



Outline Editor - Localhost:Sample:Basic:Basic

```
Database: Basic
├─ Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
├─ Measures Accounts (Label Only)
├─ Product
├─ Market
├─ Scenario (Label Only)
├─ Caffeinated Attribute
├─ Ounces Attribute
├─ Pkg Type Attribute
├─ Population Attribute
└─ Intro Date Attribute
```

Each dimension and member is listed after a tree node. The node indicates the dimension or member's position in the hierarchy:

⌄ Indicates that the branch can be expanded. For example, double-clicking on Qtr2 expands the branch to show Apr, May, and Jun.

You can also expand a member by selecting the member and choosing Outline > Expand to Children (to expand the outline one level) or by choosing Outline > Expand to Descendants (to expand the outline to the lowest level).

⌃ Indicates that the branch is expanded and can be collapsed. For example, double-clicking on Qtr1 collapses the branch to show Qtr1 but not its children.

You can also collapse a member by selecting the member and choosing Outline > Collapse to Ancestor.

□ Indicates that the member has no children.

When you open the Outline Editor, the Application Manager's menu items change. The Outline Editor contains a toolbar that provides shortcuts to Outline Editor commands and an editing area that displays the current outline. For help with any of the toolbar buttons or menu commands, use the Help menu.

You can also use the Outline Viewer window in Administration Services to open an outline for viewing.

Copying Outlines

➤ To copy a database outline, its data, and database-related files (such as calculation scripts):

1. Select the database to copy in the Application Desktop window.
2. Select Database > Copy. Essbase copies the entire database.

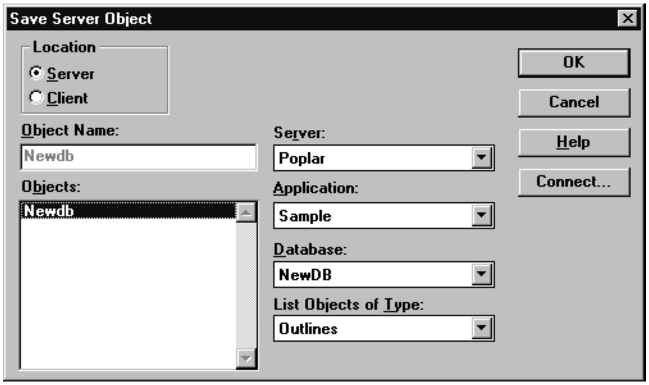
Tip: You can also copy outlines without using Application Manager:

Tool	Instructions	For More Information
Administration Services	Copy Database dialog box	*Essbase Administration Services Online Help*
MaxL	**create database as**	*Technical Reference* in the `docs` directory
ESSCMD	COPYDB	

➤ To copy only the outline without copying data or objects:

1. Create a new database by choosing File > New > Database.
2. Open the outline you want to copy in the Outline Editor. If you don't know how to do this, see "Opening Outlines" on page 168.
3. Select File > Save As to open the Save Server Object dialog box.

Figure 59: Save Server Object Dialog Box



Save Server Object

Location
● Server
○ Client

Object Name: Newdb

Objects: Newdb

Server: Poplar

Application: Sample

Database: NewDB

List Objects of Type: Outlines

4. In the Database list box, select the name of the *new* database that you created in Step 1. A dialog box is displayed, asking if you should overwrite the existing database.
5. Click Yes.

Verifying and Saving Outlines

When you save an outline to the server, Application Manager verifies the outline:

- All member and alias names are valid. Members and aliases cannot have the same name as other members, aliases, generations, or levels. The section “Rules for Naming Dimensions and Members” on page 179 presents conventions for member names.
- Only one dimension is tagged as accounts, time, currency type, or country.
- Shared members are valid as described in “Creating Shared Members” on page 215.
- Level 0 members are not tagged as label only.
- Label-only members cannot be assigned formulas.
- The currency category and currency name are valid for the currency outline.
- Dynamic Calc members do not have more than 100 children.
- Boolean attribute dimensions have only two members. Boolean attribute names are the same as the Boolean attribute member names defined for the database.
- The level 0 member name of a date attribute dimension must match the date format name setting (mm-dd-yyyy or dd-mm-yyyy). If the dimension has no members, because the dimension name is the level 0 member, the dimension name must match the setting.
- The level 0 member name of a numeric attribute dimension is a numeric value. If the dimension has no members, because the dimension name is the level 0 member, the dimension name must be a numeric value.
- Attribute dimensions are located at the end of the outline, following all standard dimensions.
- Level 0 Dynamic Calc members of standard dimensions have a formula.
- Formulas for members are valid. Essbase performs formula validation only when the Outline Editor menu toggle option Options > Server Formula Validation is set ON. (A check mark next to the option name indicates server formula validation is set ON.)
- In a Hybrid Analysis outline, only the lowest level members of a Hybrid Analysis-enabled dimension are allowed.

During outline verify, Application Manager also performs the following conversions to appropriate numeric attribute dimension member names and displays them in the outline:

- It moves minus signs in member names from the front to the end of the name; for example, -1 becomes 1-.
- It strips out leading or trailing zeroes in member names; for example, 1.0 becomes 1, and 00.1 becomes 0.1.

For more information about numeric attribute dimensions, see "Attribute Types" on page 230.

These sections explain the procedures required to verify and save outlines, and related tasks:

- "Verifying Outlines" on page 173
- "Saving Outlines" on page 174
- "Saving an Outline with Added Standard Dimensions" on page 175
- "Saving an Outline with One or More Deleted Standard Dimensions" on page 176
- "Creating Sub-Databases Using Deleted Members" on page 176
- "Setting Dense and Sparse Data Storage" on page 177

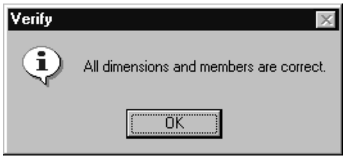
Verifying Outlines

➤ To verify an outline:

1. Select Outline > Verify or click the Verify button, ☑.

   If the outline has no errors, the Verify dialog box is displayed.

   Figure 60: Verify Dialog Box



All dimensions and members are correct.

2. Click OK.

   If the outline is not valid, a dialog box is displayed listing the errors that the outline contains.

3. Fix these errors and verify the outline again.

Saving Outlines

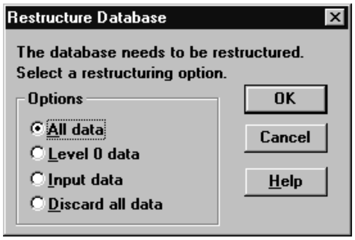
You can save outlines to the client or the server. By default, Essbase saves outlines to the server directory.

➤ To save an outline:

1. Select File > Save. Essbase saves the outline.
2. If you are saving changes to an existing outline, Essbase may restructure the outline. For example, if you change a member name from Market to Region, Essbase moves data stored in reference to Market to Region.

   If Essbase needs to restructure the database, the Restructure Database dialog box is displayed:

   Figure 61: Restructure Database Dialog Box



Restructure Database

The database needs to be restructured.
Select a restructuring option.

Options
● All data
○ Level 0 data
○ Input data
○ Discard all data

3. Select the data that Essbase should restructure.

   - Select All data when you want to keep all changes that pertain to the modified outline.
   - Select Level 0 data when you expect to recalculate the database. If all data in the outline is at level 0, choosing Level 0 data can save space and improve calculation performance.

- Select Input data when you expect to calculate and you have loaded data into non-level 0 members.
- Select Discard all data when you expect to reload the data or when your outlines is so radically changed that no existing data applies.

For more information, see Chapter 49, “Optimizing Database Restructuring.”

4. Click OK.

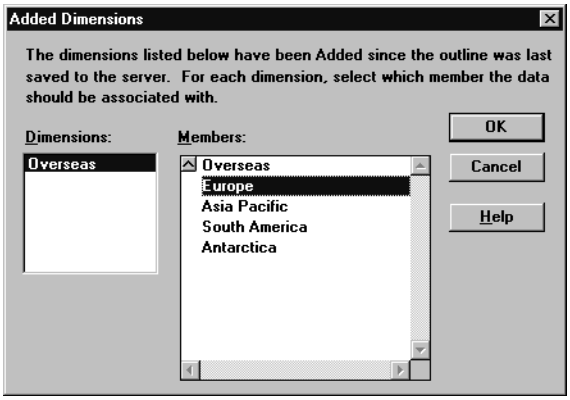
Note: Be careful when using older versions of Application Manager to open and save outlines that are created with newer versions of Essbase. If the outline contains features that are specific to the newer version, the new features are deleted from the outline when it is saved in an older version of Application Manager. For more information, see the *Essbase Installation Guide*.

Saving an Outline with Added Standard Dimensions

If you add one or more new standard (non-attribute) dimensions, then any data that existed previously in the database must be associated with a member of each new dimension. For example, adding a dimension called Channel to the Sample Basic outline implies that all previous data in Sample Basic is associated with a particular channel or the sum of all channels. See “Adding and Deleting Standard Dimensions and Stored Members” on page 74 for more information.

If you add one or more new standard dimensions, the Added Dimensions dialog box is displayed.

Figure 62: Added Dimensions Dialog Box



Added Dimensions

The dimensions listed below have been Added since the outline was last saved to the server. For each dimension, select which member the data should be associated with.

Dimensions: Overseas

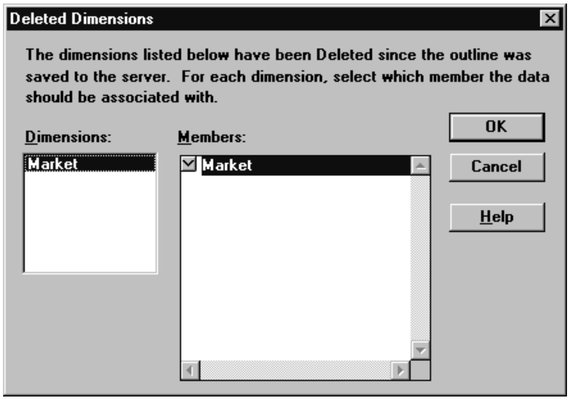
Members: Overseas, Europe, Asia Pacific, South America, Antarctica

Saving an Outline with One or More Deleted Standard Dimensions

If you delete one or more standard (non-attribute) dimensions, the data associated with only one member of each deleted dimension can be retained. For example, removing a dimension called Market from the outline implies that the data that remains in the database after the restructure operation is associated with a specified member of the Market dimension. See "Adding and Deleting Standard Dimensions and Stored Members" on page 74 for more information.

If you delete one or more dimensions, the Deleted Dimensions dialog box is displayed.

Figure 63: Deleted Dimensions Dialog Box



Deleted Dimensions

The dimensions listed below have been Deleted since the outline was saved to the server. For each dimension, select which member the data should be associated with.

Dimensions: Market

Members: Market

Select a member and click OK.

If you delete an attribute dimension, Essbase deletes the associations to its base dimension. See Chapter 9, "Working with Attributes."

Creating Sub-Databases Using Deleted Members

You can easily create sub-databases using the feature described in "Saving an Outline with One or More Deleted Standard Dimensions" on page 176.

➤ To create a sub-database by deleting a dimension from an existing outline, use this procedure:

1. Delete a dimension from the database, such as Market.
2. Save the database as another name, such as New York, and specify the member to keep, such as New York.

The new database is a working database for the New York market, and analysts in New York can use it for their analysis.

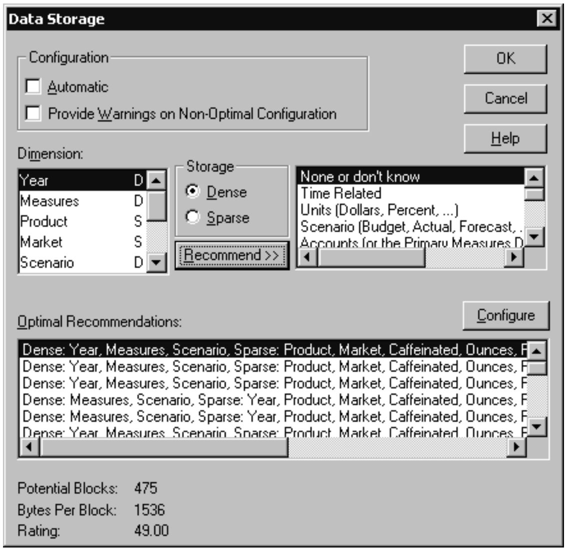
Setting Dense and Sparse Data Storage

When you create new dimensions and save the outline, Essbase automatically sets the new dimensions in the outline as sparse.

➤ To change the storage configuration of a dimension:

1. Choose Settings > Data Storage to open the Data Storage dialog box.

Figure 64: Data Storage Dialog Box



Data Storage

Configuration
□ Automatic
□ Provide Warnings on Non-Optimal Configuration

Dimension:

Year	D
Measures	D
Product	S
Market	S
Scenario	D

Storage
● Dense
○ Sparse

None or don't know
Time Related
Units (Dollars, Percent, ...)
Scenario (Budget, Actual, Forecast, .
Accounts (or the Primary Measures D

Optimal Recommendations:

Dense: Year, Measures, Scenario, Sparse: Product, Market, Caffeinated, Ounces, F
Dense: Year, Measures, Scenario, Sparse: Product, Market, Caffeinated, Ounces, F
Dense: Year, Measures, Scenario, Sparse: Product, Market, Caffeinated, Ounces, F
Dense: Measures, Scenario, Sparse: Year, Product, Market, Caffeinated, Ounces, F
Dense: Measures, Scenario, Sparse: Year, Product, Market, Caffeinated, Ounces, F
Dense: Year, Measures, Scenario, Sparse: Product, Market, Caffeinated, Ounces, F

Potential Blocks: 475
Bytes Per Block: 1536
Rating: 49.00

2. Clear Automatic.

3. Select the dimension name in the Dimension list box.
4. Select Dense or Sparse.
5. Click OK.

You must set the standard dimensions with which you plan to associate attribute dimensions as sparse. Application Manager automatically sets attribute dimensions as sparse.

For information on choosing dense or sparse storage, see “Selection of Sparse and Dense Dimensions” on page 86.

Renaming Dimensions and Members

You can change the names of existing dimensions or members.

➤ To change a dimension or member name in the Outline Editor:

1. Select the dimension or member name with a right mouse click. A member edit text box is displayed.
2. Enter the new name.
3. Press the Enter key or select another member.
4. If you have the Outline Editor set up to prompt you when you rename a dimension or member, a dialog box is displayed. Click Yes to rename the dimension or member.

➤ To rename a member using the Member Properties dialog box:

1. Select the dimension or member name in the outline.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Member Properties dialog box.
3. Enter the new name in the Name text field.
4. Click OK.

Adding Dimensions and Members to Outlines

This section describes how to add dimensions and members to outlines. It contains the following sections:

- “Rules for Naming Dimensions and Members” on page 179
- “Adding Dimensions to Outlines” on page 181
- “Adding Members to Dimensions” on page 183

Note: If you add, delete, or move non-attribute dimensions or members, Essbase restructures your database, and you must recalculate your data.

You must position attribute dimensions at the end of the outline.

Rules for Naming Dimensions and Members

When naming dimensions and members in the database outline, follow these rules.

- The maximum length is specified in Appendix A, “Limits.”
- Names are not case-sensitive unless there is a check mark next to Settings > Case Sensitive Members.
- Do not use " (quotation marks) or tabs anywhere in a name.
- Do not use the following characters at the beginning of a name:

```
@   (at sign)
\   (backslash)
{}  (braces)
,   (comma)
-   (dash, hyphen, or minus sign)
=   (equal sign)
<   (less than sign)
()  (parentheses)
.   (period)
+   (plus sign)
'   (single quotation mark)
_   (underscore)
|   (vertical bar)
```

- Do not place spaces at the beginning or end of a name. Essbase ignores spaces at the beginning or end of a name.
- Do not use the following words as dimension or member names:
  - `$$$UNIVERSE$$$` or `#MISSING` or `#MI`
  - Calculator script commands, operators, and keywords. For a list of commands, see the *Technical Reference* in the `docs` directory.
  - Report writer commands. For a list of commands, see the *Technical Reference* in the `docs` directory.
  - Function names and function arguments. For a list of functions, see the *Technical Reference* in the `docs` directory.
  - Names of other dimensions, members (unless the member is shared), generation names, level names, aliases, and aliase combinations in the database.

Note: If you enable Dynamic Time Series members, do not use the associated generation names: History, Year, Season, Period, Quarter, Month, Week, or Day. See "Applying Predefined Generation Names to Dynamic Time Series Members" on page 812.

- In calculation scripts, report scripts, filter definitions, partition definitions, or formulas, you must enclose member names in quotation marks ("") in the following situations:
  - The name contains one or more numerals at the beginning of the name (for example, 100-Blue) .
  - The name contains spaces or any of the following characters:

```
&   (ampersand)
*   (asterisk)
@   (at sign)
\   (backslash)
{}  (braces)
[]  (brackets)
:   (colon)
,   (comma)
-   (dash, hyphen, or minus sign)
=   (equal sign)
!   (exclamation point)
>   (greater than sign)
<   (less than sign)
```

```
() (parentheses)
%  (percent sign)
.  (period)
+  (plus sign)
;  (semicolon)
/  (slash)
~  (tilde)
```

- In calculation scripts and formulas, you must enclose the following member names in quotation marks (""):

```
AND       IF        BEGIN
MACRO     DOUBLE    MBR
DYNAMIC   MEMBER    ELSE
NOT       ELSEIF    OR
END       RANGE     ENDIF
STRING    FUNCTION  THEN
```

Tips to make names unique:

- Concatenate the member and alias names; for example, 100-10_Cola and 100-10_Smith.
- Add prefixes or suffixes to member names. For example, if the parent is the state and several states have a city called Jackson, appending the state abbreviation creates the unique names Jackson_CA, Jackson_IL, and Jackson_MI.

Adding Dimensions to Outlines

Dimensions are the highest level of organization in an outline. Dimensions are made up of members. For more information on dimensions, see Chapter 2, “Multidimensional Concepts.”

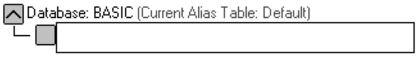
➤ To add dimensions as children of the outline:

1. Select the database name, for example, Basic.
2. Select Edit > Add Child or click the Add Child button, ↳□, to open a member edit text box in the Outline Editor.

➤ To add dimensions as siblings of other dimensions:

1. Select the dimension after which you want to add the new dimension.
2. Select Edit > Add Sibling or click the Add Sibling button, , to open a member edit text box in the Outline Editor.

Figure 65: Member Edit Text Box

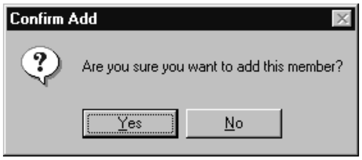


Database: BASIC (Current Alias Table: Default)

➤ To finish adding dimensions, proceed with the following steps:

1. Enter the name of the dimension; for example, Year. See "Rules for Naming Dimensions and Members" on page 179 for restrictions.
2. Press Enter to display a confirming dialog box:

Figure 66: Confirm Add Dialog Box for Dimensions and Members



Confirm Add

Are you sure you want to add this member?

3. Click OK.
4. Enter another dimension name or press Enter to dismiss the member edit text box.

To rearrange dimensions in the outline, see "Positioning Dimensions and Members" on page 184.

Adding Members to Dimensions

Members are organized into dimensions. You can nest members inside of other members. For more information on members, see Chapter 2, “Multidimensional Concepts.”

➤ To add members as children of dimensions:

1. Select the dimension; for example, Year.

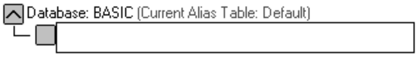
2. Select Edit > Add Child or click the Add Child button, , to open a member edit text box in the Outline Editor.

➤ To add members as siblings of other members:

1. Select the member after which you want to add the new member; for example, Qtr1.

2. Select Edit > Add Sibling or click the Add Sibling button, , to open a member edit text box in the Outline Editor.

Figure 67: Member Edit Text Box



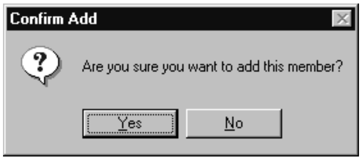
Database: BASIC (Current Alias Table: Default)

➤ To finish adding members, proceed with the following steps:

1. Enter the name of the member; for example, Qtr2. See “Rules for Naming Dimensions and Members” on page 179 for restrictions.

2. Press Enter to display a confirming dialog box:

Figure 68: Confirm Add Dialog Box for Dimensions and Members



Confirm Add

Are you sure you want to add this member?

3. Click OK.

4. Enter another member name or press Enter to dismiss the member edit text box.

To rearrange members in the outline, see "Positioning Dimensions and Members" on page 184.

Positioning Dimensions and Members

You can rearrange dimensions within an outline or members within a dimension. You can select to position dimension and members by:

- "Sorting Dimensions and Members" on page 184
- "Moving Members" on page 186

Note: If you add, delete, or move dimensions or members, Essbase restructures the database, and you must recalculate the data.

If you do not position attribute dimensions at the end of the outline, during outline verification, Application Manager prompts you to move them there.

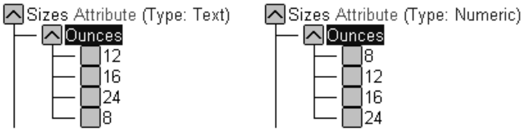
The positions of dimensions in an outline can affect performance. See "Outline Performance Considerations" on page 238.

Sorting Dimensions and Members

You can arrange dimensions within an outline or members within a dimension in alphabetical order (A to Z) or reverse alphabetical order (Z to A).

When you sort level 0 members of numeric attribute dimensions in outlines, the members are sorted by their values. For example, Figure 69 shows text and numeric versions of the Sizes attribute dimension after sorting the members in ascending order. The members of the numeric attribute dimension are sequenced by the numeric values of the members; the member 8 is before the other members. In the text attribute dimension, because the characters are sorted left to right, the member 8 is after the member 24.

Figure 69: Sorting Numeric Versus Text Attribute Dimension in Ascending Order



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

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

You cannot sort Boolean attribute dimensions. For more information about attribute dimension types, see "Attribute Types" on page 230.

CAUTION: Moving dimensions and members can affect the results of your batch calculations. See "Designing an Outline to Optimize Performance" on page 124. Also, sorting members could move a shared member before the actual member in the outline, something that we do not recommend.

➤ To sort all dimensions in an outline:

1. Select the database name in the outline; for example, Basic.
2. Select Edit > Sort Ascending to sort the dimensions in alphabetical order or select Edit > Sort Descending to sort the dimensions in reverse alphabetical order.

➤ To sort all members in the level below the selected dimension or member:

1. Select the dimension or member containing the members to sort; for example, Market.
2. Select Edit > Sort Ascending to sort the members in alphabetical order or select Edit > Sort Descending to sort the members in reverse alphabetical order.

Moving Members

CAUTION: Moving dimensions and members can affect the results of your calculations and retrievals. See “Designing an Outline to Optimize Performance” on page 124. Also, sorting members could move a shared member before the actual member in the outline, something that we do not recommend.

➤ To move one or more members to another part of the outline:

1. Select the member or members you want to move by pressing the left mouse button on a member name. (To select multiple members, hold down the **Ctrl** key while selecting member names.) Continue to hold down the mouse button throughout the move operation.
2. Drag the selection to the desired location in the tree. As you drag the selection, a tree icon ⊞ is displayed.
3. Place the icon over a member at the new location. A border is displayed around the member name.
   - To insert the member as a sibling of the member, place the icon on the □, or to the left of the border. The icon is displayed with a straight line ⊞.
   - To insert the member as a child of the member, place the icon in or to the right of the border. The icon is displayed with a bent line └⊞.
4. Release the mouse button.

Naming Generations and Levels

You can create your own names for generations and levels in an outline. The name is a word or phrase that describes the generation or level. For example, you might create a generation name called Cities for all cities in the outline.

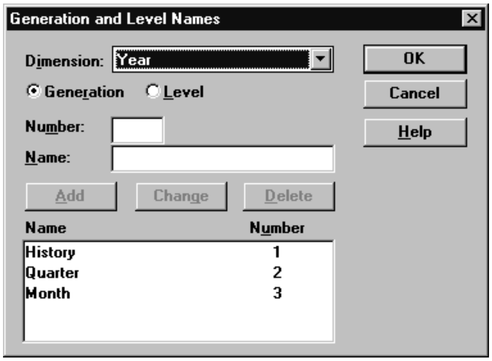
Use generation and level names in calculation scripts or report scripts wherever you need to specify either a list of member names or generation or level numbers. For example, you could limit a calculation in a calculation script to all members in a specific generation. See Chapter 30, “Developing Calculation Scripts” for information about developing calculation scripts.

See “Member Relationships, Generations, and Levels” on page 63 for information about generations and levels.

➤ To create a generation name:

1. Select Outline > Gen/Level Names to open the Generation and Level Names dialog box.

Figure 70: Generation and Level Names Dialog Box



Dimension: Year

● Generation ○ Level

Number:

Name:

Name	Number
History	1
Quarter	2
Month	3

2. Select the appropriate dimension name from the Dimension list box; for example, Year. By default, Essbase displays the current dimension.
3. To name a generation, select Generation. To name a level, select Level. For example, to name the months in the Sample Basic database, select Generation.

4. Enter the generation or level number in the Number text box. For example, to name the months in the Sample Basic database, enter 3.
5. Enter the generation or level name in the Name text box; for example, to name the months in the Sample Basic database, enter Months.

   Note: You can define only one name for each generation or level. When you name the generations and levels, follow the same naming rules as for members. See "Rules for Naming Dimensions and Members" on page 179.

6. Click Add to display the new name in the list box.
7. Click OK.

To view the list of generation or level names, select Outline > Gen/Level Names again or print the outline.

Customizing the Outline Editor

The Outline Editor displays information about the outline and the dimensions and members that it contains. You can customize what Essbase displays in the Outline Editor and which font it uses. This section includes the following topics:

- "Making Members Case-Sensitive" on page 188
- "Customizing the View" on page 189
- "Setting the Outline Font" on page 190
- "Confirming Changes to Dimensions and Members" on page 190

Making Members Case-Sensitive

You can specify whether members in the outline should be case-sensitive. For example, Budget and budget are both unique names if the member names are case-sensitive. By default, member names are not case-sensitive.

To set member names to be case-sensitive, select Case Sensitive Members in the Settings menu of the Outline Editor.

Customizing the View

You can customize the view of the Outline Editor to view or hide the following details by clearing them in the View menu:

- Consolidation Objects—displays consolidation operators (such as +, -, and %) in parentheses to the right of the member. For information on setting consolidation operators, see "Setting Member Consolidation Properties" on page 210.
- Formula Objects—displays formulas to the right of the member. For information on creating formulas, see "Naming Generations and Levels" on page 187.
- Dimension Tags—displays dimension tags to the right of the member, if a dimension tag other than None is selected. For information on adding dimension tags, see "Setting the Dimension Type" on page 194.
- Aliases—displays aliases in parentheses to the right of the member. For more information on creating aliases, see Chapter 10, "Creating and Managing Aliases."
- Member Properties—displays member properties to the right of the member. For more information on assigning properties, see Chapter 8, "Setting Dimension and Member Properties."
- Comments—displays comments to the right of the member. For more information on adding comments to members, see "Setting Comments on Dimensions and Members" on page 222.
- Attribute Associations—displays associated attribute dimension names to the right of the base dimension and associated attribute dimension members to the right of base dimension members. For information on associating attribute dimensions and members to base dimensions, see Chapter 9, "Working with Attributes."
- Attributes Type—displays the attribute dimension type to the right of attribute dimension names. For information on attribute dimension types, see "Attribute Types" on page 230.
- Toolbar—displays the Outline Editor toolbar, which contains shortcuts for working with outlines, dimensions, and members.
- Properties Bar—displays the Outline Editor properties bar, which contains shortcuts for changing or assigning properties to dimensions and members.

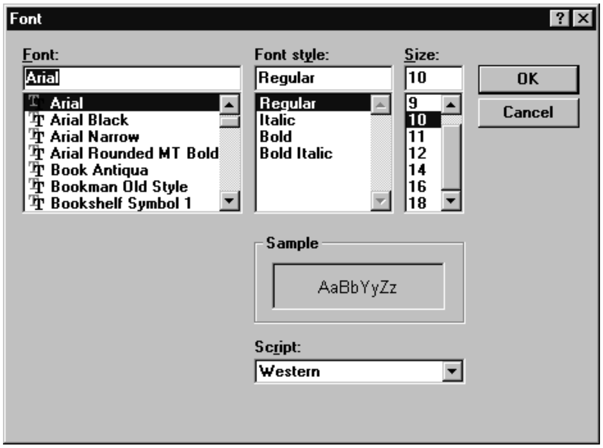
Setting the Outline Font

You can specify the font that Essbase uses to display text in the Outline Editor.

➤ To set a font:

1. Select Options > Font to open the Font dialog box.

Figure 71: Font Dialog Box



Font: Arial

Font style: Regular

Size: 10

Script: Western

2. Select the desired font from the Font list. You can also select the style and size of font. The default display font is Arial, Regular, 10.
3. Click OK.

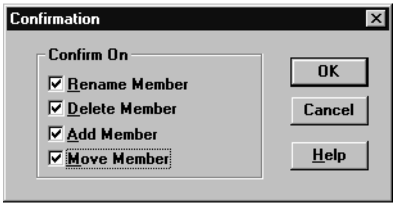
Confirming Changes to Dimensions and Members

You can specify whether Essbase prompts you when you make changes to dimensions or members. By default, Essbase opens a dialog box asking you if you’re sure you want to change the member or dimension. You must click Yes before Essbase makes the change.

➤ To change the default way Essbase prompts users:

1. Select Options > Confirmation to open the Confirmation dialog box:

Figure 72: Confirmation Dialog Box



Confirmation

Confirm On

☑ Rename Member
☑ Delete Member
☑ Add Member
☑ Move Member

2. Clear the options for which Essbase should not prompt.

   By default, all options are selected, which means that Essbase opens a dialog box to prompt you before it makes the changes. When an option is not selected, Essbase does *not* prompt you before making that kind of change.

3. Click OK.

IBM Relational Storage Manager

If you use the relational storage manager in IBM's DB2 OLAP Server, any changes to metadata are reflected in both Essbase and the DB2 OLAP Server. Metadata is data contained in the database outline that describes the values within a database. Dimension and member names are metadata. Data that describes values is stored only on the DB2 OLAP Server.

Chapter

8

Setting Dimension and Member Properties

After you create and organize your Essbase OLAP Server outline, as described in Chapter 7, “Creating and Changing Database Outlines,” you are ready to specify how the dimensions and members in the outline behave. This chapter describes each dimension and member property and how to set them in the Outline Editor in Application Manager.

Note: For information on setting Dynamic Time Series members, see Chapter 29, “Calculating Time Series Data.”

Setting the Dimension Type

When you tag a dimension as a specific type, it can access built-in functionality designed for that type. For example, if you define a dimension as accounts, you can specify accounting measures for members in that dimension. The two primary dimension types are time and accounts. This means that Essbase calculates dimensions tagged as time and accounts before other dimensions in the database. By default, all dimensions are tagged as None.

Note: The time and accounts properties are inherited by all members that are in these dimensions. The Country and Currency properties are not inherited by their members.

The following sections describe how to tag dimensions:

- "Tagging a Time Dimension" on page 194
- "Setting the Time Property" on page 195
- "Tagging an Accounts Dimension" on page 195
- "Tagging a Country Dimension" on page 203
- "Tagging a Currency Partition" on page 204
- "Tagging an Attribute Dimension" on page 205

Tagging a Time Dimension

You can optionally use dimensions tagged as time to describe how often you collect and update data. The time dimension enables several accounts dimension functions, such as first and last time balances. In the Sample Basic database, for example, the Year dimension is tagged as time, as are its descendants—all Qtr members and the months (such as Jan).

Follow these rules when tagging a dimension as time:

- You can only tag one dimension in an outline as time.
- When you tag a dimension as time, all members in that dimension inherit the time property.
- You can add time members to dimensions that are not tagged as time.

Setting the Time Property

➤ To use a button to tag a dimension as time:

1. Select the dimension that you want to tag; for example, Year.
2. Click the Time button, , to display “Time” next to the dimension name.

➤ To tag a dimension as time using the Dimension Properties dialog box:

1. Select the dimension that you want to tag; for example, Year.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.
3. Select Time in the Dimension Type box.
4. Click OK.

Tagging an Accounts Dimension

You can optionally tag a dimension as accounts if it contains items that you want to measure, such as profit or inventory.

Follow these rules when tagging an accounts dimension:

- You can only tag one dimension in an outline as accounts.
- When you tag a dimension as accounts, all members in that dimension inherit the accounts property.
- To calculate members of the accounts dimension on the second pass through the outline, see “Setting Two-Pass Calculation Properties” on page 207.

The following sections describe built-in functionality for accounts dimensions:

- "Setting the Accounts Property" on page 196
- "Introducing Time Balance Properties" on page 196
- "Introducing Skip Properties" on page 199
- "Setting Variance Reporting Properties" on page 201
- "Setting Essbase Currency Conversion Properties" on page 202

To perform the tasks in the following sections, the Measures dimension in the Sample Basic database must be tagged as accounts.

Setting the Accounts Property

➤ To use a button to tag a dimension as accounts:

1. Select the dimension that you want to tag; for example, Measures.
2. Click the Accounts button, Acct, to display "Accounts" next to the dimension name.

➤ To tag a dimension as accounts using the Dimension Properties dialog box:

1. Select the dimension that you want to tag; for example, Year.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.
3. Select Accounts in the Dimension Type box.
4. Click OK.

Introducing Time Balance Properties

When you set a time balance property on a member in an accounts dimension, it affects how Essbase calculates the parent of that member in the time dimension. By default, a parent in the time dimension is calculated based on the consolidation and formulas of its children. For example, the Qtr1 member is the sum of its children (Jan, Feb, and Mar). However, setting a time balance property causes parents, for example Qtr1, to roll up differently.

These sections explain examples of setting Time Balance:

- "Example of Setting the Time Balance as None" on page 197
- "Example of Setting the Time Balance as First" on page 197
- "Example of Setting the Time Balance as Last" on page 198
- "Example of Setting the Time Balance as Average" on page 198

Example of Setting the Time Balance as None

This is the default value. When you set the time balance property as none, Essbase rolls up parents in the time dimension in the usual way—a parent's value is based on the formulas and consolidation properties of its children.

Example of Setting the Time Balance as First

Set the time balance as first when you want the parent value to represent the value of the first member in the branch (often at the beginning of a time period).

For example, let's assume that you have a member named OpeningInventory that represents the inventory at the beginning of the time period. If the time period was Qtr1, then OpeningInventory represents the inventory you had at the beginning of Jan. When you ask for the OpeningInventory for Qtr1, you want it to be the same as the OpeningInventory for Jan. That is, if you had 50 cases of Cola at the beginning of Jan, you also had 50 cases of Cola at the beginning of Qtr1.

To do this, tag OpeningInventory as first. Now Essbase calculates the value of OpeningInventory for Qtr1 as the same as the OpeningInventory for Jan. Figure 73 shows this sample consolidation.

Figure 73: Consolidation of OpeningInventory Tagged as First

```
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
```

Example of Setting the Time Balance as Last

Set the time balance as last when you want the parent value to represent the value of the last member in the branch (often at the end of a time period).

For example, let’s assume that you have a member named EndingInventory that represents the inventory at the end of the time period. If the time period was Qtr1, then EndingInventory represents the inventory you had at the end of Mar. When you ask for the EndingInventory for Qtr1, you want it to be the same as the EndingInventory for Mar. That is, if you had 70 cases of Cola at the end of Mar, you also had 70 cases of Cola at the end of Qtr1.

To do this, tag EndingInventory as last. Now Essbase calculates the value of EndingInventory for Qtr1 as the same as the EndingInventory for Mar. Figure 74 shows this sample consolidation.

Figure 74: Consolidation of EndingInventory Tagged as Last

```
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
```

Example of Setting the Time Balance as Average

Set the time balance as average when you want the parent value to represent the average value of its children.

For example, let’s assume that you have a member named AverageInventory that represents the average of the inventory for the time period. If the time period was Qtr1, then AverageInventory represents the average of the inventory you had during Jan, Feb, and Mar.

To do this, tag AverageInventory as average. Now Essbase calculates the value of AverageInventory for Qtr1 as the average of the values for Jan, Feb, and Mar. Figure 75 shows this sample consolidation.

Figure 75: Consolidation of AverageInventory Tagged as Average

```
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
```

Introducing Skip Properties

If you set the time balance as first, last, or average, you must set the skip property to tell Essbase what to do when it encounters missing values or values of 0.

The following table describes how each setting determines what Essbase does when it encounters a missing or zero value.

Setting	Action Essbase Takes
None	Does not skip data when calculating the parent value.
Missing	Skips #MISSING data when calculating the parent value.
Zeros	Skips data that equals zero when calculating the parent value.
Missing and Zeros	Skips both #MISSING data and data that equals zero when calculating the parent value.

If you mark a member as last with a skip property of missing or missing and zeros, then the parent of that time period matches the last non-missing child. In Figure 76, for example, EndingInventory is based on the value for Feb, because Mar does not have a value.

Figure 76: Example of Skip Property

```
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
```

Setting Time Balance Properties

➤ To use buttons to set a time balance property:

1. Select the member that you want to set the property for; for example, OpeningInventory.
2. Click the button that corresponds to the property that you want to set.
   - To set the time balance property as first, click the First button, JFM. "TB First" displays next to the member's name.
   - To set the time balance property as last, click the Last button, JFM. "TB Last" displays next to the member's name.
   - To set the time balance property as average, click the Average button, J+F+M/3. "TB Average" displays next to the member's name.
   - To set the time balance property as none, click the None button, 🚫.
3. Click the button that corresponds to the skip property that you want to set.
   - To not skip any values, click the None button, 🚫.
   - To skip missing values, click the Missing button, #MI.
   - To skip zero values, click the Zero button, 0.
   - To skip both zero and missing values, click the Missing and Zeros button, #MI, 0, X.

➤ To tag a dimension as accounts using the Dimension Properties dialog box:

1. Select the member that you want to set the property for; for example, OpeningInventory.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.
3. Click the Accounts tab.
4. Select None, First, Last, or Average in the Time Balance box.
5. Select None, Missing, Zeros, or Missing and Zeros in the Skip box.
6. Click OK.

Setting Variance Reporting Properties

Variance reporting properties determine how Essbase calculates the difference between actual and budget data in a member with the @VAR or @VARPER function in its the member formula. Any member that represents an expense to the company requires an expense property.

When you are budgeting *expenses* for a time period, the actual expenses should be lower than the budget. When actual expenses are greater than budget, the variance is negative. The @VAR function calculates Budget - Actual. For example, if budgeted expenses were $100, and you actually spent $110, the variance is -10.

When you are budgeting *non-expense* items, such as sales, the actual sales should be higher than the budget. When actual sales are less than budget, the variance is negative. The @VAR function calculates Actual - Budget. For example, if budgeted sales were $100, and you actually made $110 in sales, the variance is 10.

By default, members are non-expense.

➤ To use a button to set an expense property:

1. Select the member that you want to set the property for; for example, COGS.
2. Click the Expense tag button, -$. "Expense Reporting" displays next to the member name.

➤ To tag an accounts member as expense or non-expense using the Member Properties dialog box:

1. Select the member that you want to set the property for; for example, COGS.
2. Click the Data Dictionary button, 🛡, press the Enter key, or select Edit > Properties to open the Member Properties dialog box.
3. Click the Accounts tab.
4. Select Expense or Non Expense.
5. Click OK.

Setting Essbase Currency Conversion Properties

Currency conversion properties define categories of currency exchange rates. These properties are used only in currency databases. For more information on currency properties, see Chapter 12, "Designing and Building Currency Conversion Applications."

Setting	Action Essbase Takes
None	Determines that the member has no relationship to currency conversion. This is the default.
No Conversion	Does not convert the member because it is not a currency value. It could be a value such as a quantity or percentage.
Category	Converts the member. You can enter the type of conversion required. This could be a value, normally in dollars.

➤ To set currency conversion properties in the Outline Editor:

1. Select the member. The member must be in a currency database and be part of a dimension tagged as accounts.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.
3. Click the Account Info tab.
4. Select None, No Conversion, or Category from the currency conversion box.

Tagging a Country Dimension

Use country dimensions to track business activities in multiple countries. If you track business activity in the United States and Canada, for example, your country dimension should contain states, provinces, and countries. If a dimension is tagged as country, you can set the currency name property. The currency name property defines what type of currency this market region uses.

In a country dimension, you can specify the type of currency used in each member. For example, in the Interntl application and database shipped with Essbase, Canada has three markets: Vancouver, Toronto, and Montreal. They use the same currency, Canadian dollars.

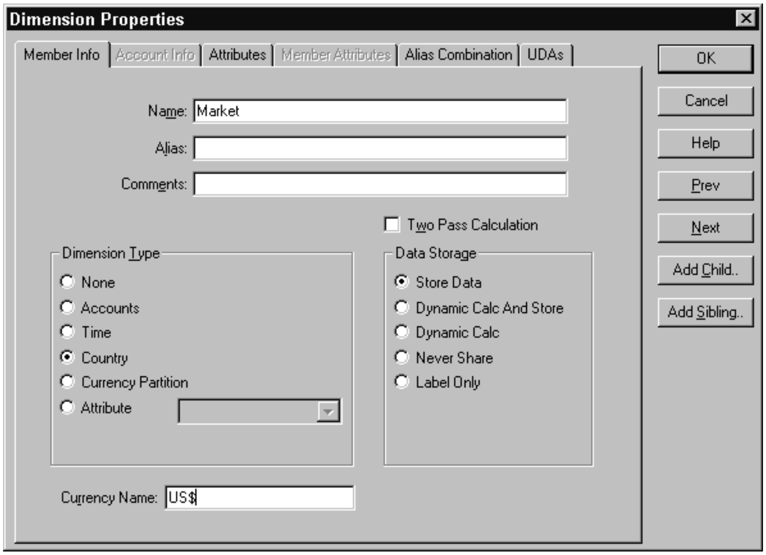
This dimension type is used for currency conversion applications. For more information, see Chapter 12, “Designing and Building Currency Conversion Applications.”

➤ To tag a dimension as country:

1. Select the dimension that you want to tag; for example, Market.
2. Click the Country button, , to display “Country” next to the dimension name.

3. If you want to set the currency name, click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box:

Figure 77: Country Dimension Tagged in Dimension Properties Dialog Box



Name: Market

Alias:

Comments:

☐ Two Pass Calculation

Dimension Type

- ○ None
- ○ Accounts
- ○ Time
- ● Country
- ○ Currency Partition
- ○ Attribute

Data Storage

- ● Store Data
- ○ Dynamic Calc And Store
- ○ Dynamic Calc
- ○ Never Share
- ○ Label Only

Currency Name: US$

4. Enter the currency name, such as US$, in the Currency Name text box.
5. Click OK.

Tagging a Currency Partition

Use currency partition members to separate local currency members from a base currency defined in your application. If your base currency for analysis is US dollars, for example, the local currency members would contain values based on the currency type of the region, such as Canadian dollars.

This dimension type is used for currency conversion applications. For more information, see “Modify the Scenario Dimension” on page 293.

For more information about designing and implementing currency applications, see Chapter 12, “Designing and Building Currency Conversion Applications.”

Tagging an Attribute Dimension

Use attribute dimensions to report and aggregate data based on characteristics of standard dimensions. In the Sample Basic database, for example, the Product dimension is associated with the Ounces attribute dimension. Members of the Ounces attribute dimension categorize products based on their size in ounces. For information about attribute dimensions, see Chapter 9, "Working with Attributes."

Keep in mind the following information about attribute dimensions when you tag a dimension as attribute:

- You can tag only sparse dimensions as attribute.
- Before you can save an outline to the server, each attribute dimension must be associated with a standard, sparse dimension as its base dimension.
- Attribute dimensions must be the last dimensions in the outline.
- Attribute dimensions have a type setting: text, numeric, Boolean, or date. Text is the default setting. For more information, see "Attribute Types" on page 230.
- If you remove the attribute tag from a dimension, Application Manager removes prefixes or suffixes from its member names. Prefixes and suffixes are not visible in the outline. For more information, see "Defining a Prefix or Suffix Format for Member Names of Attribute Dimensions" on page 240.

Setting the Attribute Property

➤ To use a button to tag a dimension as attribute:

1. Select the dimension that you want to tag; for example, Ounces.
2. Click the Attribute button, Attr.

   Outline Editor displays "Attribute" next to the dimension name.

➤ To tag a dimension as attribute using the Dimension Properties dialog box:

1. Select the dimension that you want to tag; for example, Ounces.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.
3. Select Attribute in the Dimension Type box.
4. Click OK.

Setting the Attribute Type

Attributes have a text, Boolean, date or numeric type property. Although assigned at the dimension level, the type applies only to the level 0 members of the dimension. Essbase sets text as the default attribute dimension type. See “Attribute Types” on page 230 for more information.

➤ To use buttons to set the attribute type:

1. Select the attribute dimension; for example, Ounces.
2. Click the button corresponding to the type that you specify:

   - For text, click the T button, forexample, for member names Bottle and Can.
   - For Boolean, click the B button, for example, for attributes describing binary situations such as TRUE and FALSE.
   - For date, click the D button, for example, for 09-15-1999. Date attributes cannot support member names like September 15, 1999.
   - For numeric, click the N button, for example, for attribute names you use in calculations such as 12 or 16.

➤ To set the attribute type in the Dimension Properties dialog box:

1. Select the dimension that you want to tag; for example, Ounces.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.
3. Select the dimension type in the drop-down list box next to Attribute in the Dimension Type box.

   Note: To enable the drop-down list box, select the Attribute option.

4. Click OK.

Setting Two-Pass Calculation Properties

By default, Essbase calculates outlines from the bottom up—first calculating the values for the children and then the values for the parent. Sometimes, however, the values of the children may be based on the values of the parent or the values of other members in the outline. To obtain the correct values for these members, Essbase must first calculate the outline and then re-calculate the members that are dependent on the calculated values of other members. The members that are calculated on the second pass through the outline are called *two-pass calculations*.

For more information on bottom-up calculations, see “Using Bottom-Up Calculation” on page 1399.

For example, to calculate the ratio between Sales and Margin, Essbase needs first to calculate Margin, which is a parent member based on its children, including Sales. To ensure that the ratio is calculated based on a freshly calculated Margin figure, tag the Margin % ratio member as a two-pass calculation. Essbase calculates the database once and then calculates the ratio member again. This produces the correct result.

Note: Even though two-pass calculation is a property that you can give to any non-attribute dimension member, it works only on members of accounts dimensions, Dynamic Calc members, and Dynamic Calc And Store members. If two-pass calculation is assigned to other members, Hyperion Essbase ignores it.

➤ To set a member to be calculated on the second pass in the Outline Editor:

1. Select the member that you want to set the property for; for example, Margin%.
2. Click the Two-Pass tag button, . “Two Pass Calc” displays next to the member name.

➤ To set a member to be calculated on the second pass using the Member Properties dialog box:

1. Select the dimension or member; for example, Margin %.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Member Specification dialog box.
3. Select Two Pass Calculation.
4. Click OK.

Introducing Member Consolidation Properties

Member consolidation properties determine how children roll up into their parents. By default, new members are given the addition (+) operator, meaning that members are added. For example, Jan, Feb, and Mar figures are added and the result stored in their parent, Qtr1.

Note: Essbase does not use consolidation properties with members of attribute dimensions. The Attribute Calculations dimension provides consolidation totals for attribute dimensions. See “Calculating Attribute Data” on page 253.

List of Member Consolidation Operators

Table 12 describes each operator.

Table 12: Consolidation Operators

Operator	Description
+	Adds the member to the result of previous calculations performed on other members. This is the default operator.
-	Multiplies the member by -1 and then adds it to the sum of previous calculations performed on other members.
*	Multiplies the member by the result of previous calculations performed on other members.
/	Divides the member into the result of previous calculations performed on other members.
%	Divides the member into the sum of previous calculations performed on other members. The result is multiplied by 100 to yield a percentage value.
~	Does not use the member in the consolidation to its parent.

Calculating Members with Different Operators

When siblings have different operators, Essbase calculates the data in top-down order. The following section describes how Essbase calculates the members in Figure 78.

Figure 78: Sample Roll Up

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

Essbase calculates Member1 through Member4 in Figure 78 using this precedence:

Figure 79: Sample Roll Up for Members 1 through 4

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

If the result of Figure 79 is X, then Member5 consolidates thus:

Figure 80: Sample Roll Up for Member 5

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

If the result of Figure 80 is Y, then Member6 consolidates thus:

Figure 81: Sample Roll Up for Member 6

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

Because it is set to No Consolidation(~), Essbase ignores Member7 in the consolidation.

Setting Member Consolidation Properties

Note: Consolidation properties do not apply to members of attribute dimensions.

➤ To set the consolidation property for a member using the toolbar in the Outline Editor:

1. Select the member.
2. Click the button corresponding to the consolidation you specify:

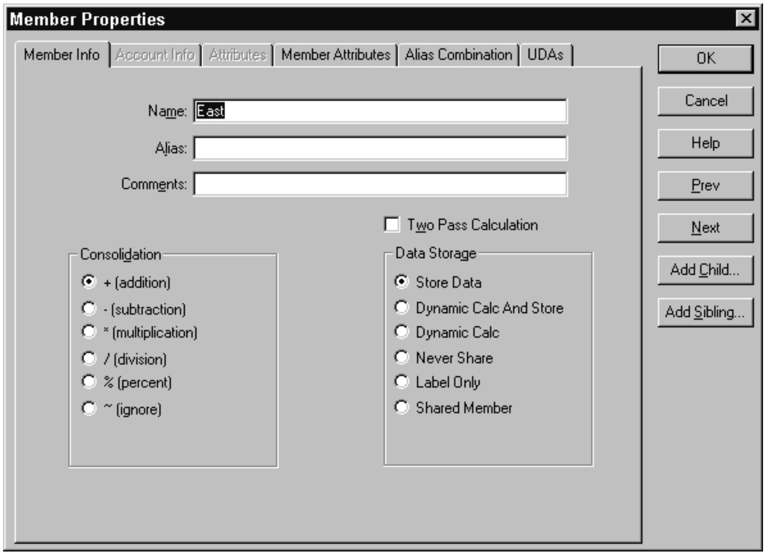
   - For addition, click the ⊞ button.
   - For subtraction, click the ⊟ button.
   - For multiplication, click the ⊠ button.

- For division, click the / button.
- For percents, click the % button.
- To exclude the member from the consolidation, click the ~ button.

➤ To set the consolidation property for a member using the Member Properties dialog box:

1. Select the member.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Member Properties dialog box.

Figure 82: Setting Consolidation Properties



Name: East

Alias:

Comments:

☐ Two Pass Calculation

Consolidation
- ● + (addition)
- ○ - (subtraction)
- ○ * (multiplication)
- ○ / (division)
- ○ % (percent)
- ○ ~ (ignore)

Data Storage
- ● Store Data
- ○ Dynamic Calc And Store
- ○ Dynamic Calc
- ○ Never Share
- ○ Label Only
- ○ Shared Member

3. Select the operator from the Consolidation box on the Member Info page.

Setting Storage Properties

You can determine how and when Essbase stores the data values for a member. For example, you can tell Essbase to only calculate the value for a member when a user requests it and then discard the data value. Table 13 lists each storage property and tells you when to set it and where to go to learn how to set it.

Table 13: Choosing Storage Properties

Storage Property	When to Use	For More Information
Store	Store the data value with the member.	“Storing Data” on page 212
Dynamic Calc And Store	Not calculate the data value until a user requests it, and then store the data value.	“Dynamically Calculating Data” on page 213
Dynamic Calc	Not calculate the data value until a user requests it, and then discard the data value.	“Dynamically Calculating Data” on page 213
Never share	Not allow members to be shared implicitly.	“Implied Sharing” on page 218
Label only	Create members for navigation only, that is, members that contain no data values.	“Creating Label Only Members” on page 214
Shared member	Share values between members. For example, the 100-20 member is stored under the 100 parent and shared under Diet parent.	“Creating Shared Members” on page 215

Storing Data

By default, Essbase stores each data value with the associated member. For example, if 50 cases of Cola were sold in January in Massachusetts, Essbase stores 50 at the intersection of Cola, Jan, Massachusetts.

➤ To use a button to tag a member as stored:

1. Select the member.
2. Click the Store button, .

➤ To tag a member as stored using the Member Properties dialog box:

1. Select the member.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Member Properties dialog box.
3. Select Store Data in the Data Storage box on the Member Info page.
4. Click OK.

Dynamically Calculating Data

When a member is Dynamic Calc, Essbase does not calculate the value for that member until a user requests it. After the user views it, Essbase does not store the value for that member. If you tag a member as Dynamic Calc And Store, Essbase performs the same operation as for a Dynamic Calc member, except that Essbase does store the data value for that member after the user views it.

For more information on Dynamic Calc or Dynamic Calc And Store members, see Chapter 28, “Dynamically Calculating Data Values.”

Note: Essbase automatically tags members of attribute dimensions as Dynamic Calc. You cannot change this setting.

➤ To use buttons to tag a member as Dynamic Calc or Dynamic Calc And Store:

1. Select the member.
2. Click the Dynamic Calc button, , or the Dynamic Calc And Store button, . “Dynamic Calc” or “Dynamic Calc And Store” displays next to the member names.

➤ To tag a member as Dynamic Calc or Dynamic Calc And Store using the Member Properties dialog box:

1. Select the member.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Member Properties dialog box.

3. Select Dynamic Calc or Dynamic Calc And Store in the Data Storage box on the Member Info page.
4. Click OK.

Creating Label Only Members

Label only members have no data associated with them. Use them to group members or to ease navigation and reporting from the Spreadsheet Add-in. Typically, you should give label only members the no consolidation property. For more information on the no consolidation property, see "Setting Member Consolidation Properties" on page 210.

Note: You cannot associate attributes with label only members. If you tag as label only a base dimension member that has attributes associated with it, Essbase removes the attribute associations and displays a warning message.

➤ To use a button to tag a member as label only:

1. Select the member; for example, Inventory.
2. Click the Label Only button, L. "Label Only" displays next to the member name.

➤ To tag a member as label only using the Member Properties dialog box:

1. Select the member.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Member Properties dialog box.
3. Select Label Only in the Data Storage box on the Member Info page.
4. Click OK.

Creating Shared Members

The data values associated with a shared member come from another member with the same name. The shared member stores a pointer to data contained in the other member and the data is only stored once. To define a member as shared, there must be an actual non-shared member of the same name. For example, in the Sample Basic database, the 100-20 member under 100 stores the data for that member. The 100-20 member under Diet points to that value.

Shared members are typically used to calculate the same member across multiple parents. For example, you might want to calculate a Diet Cola member in both the 100 and Diet parents.

Using shared members lets you use members repeatedly throughout a dimension. Essbase stores the data value only once, but it displays in multiple locations. This offers considerable space saving as well as processing efficiency.

Use these sections to understand and create shared members:

- "Rules for Shared Members" on page 215
- "Shared Member Retrieval During Drill-Down" on page 216
- "Implied Sharing" on page 218
- "Setting the Shared Member Property" on page 219

Rules for Shared Members

Follow these rules when creating shared members:

- The shared members must be in the same dimension. For example, both 100-20 members in the Sample Basic database are in the Product dimension.
- Shared members cannot have children.
- You can have an unlimited number of shared members with the same name.
- You cannot assign UDAs or formulas to shared members.
- You cannot associate attributes with shared members.
- You can assign aliases to shared members.
- You should not create an outline where shared members are located before actual members in a dimension.

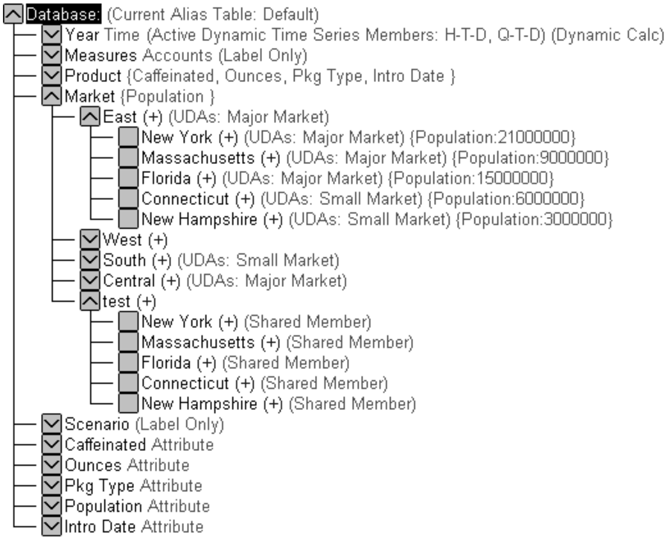
Shared Member Retrieval During Drill-Down

Essbase retrieves shared members during drill-down, depending on their location in the spreadsheet. Essbase follows three rules during this type of retrieval:

- Essbase retrieves stored members (not their shared member counterparts) by default.
- Essbase retrieves from the bottom of a spreadsheet first.
- If a shared member's parent is a sibling of the stored member counterpart of one of the shared members, Essbase retrieves the stored member.

Example: Shared Members from a Single Dimension

If you created a test dimension with all shared members based on the members of the dimension East from Sample, your outline would be similar to this:



```
Database: (Current Alias Table: Default)
  Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
  Measures Accounts (Label Only)
  Product {Caffeinated, Ounces, Pkg Type, Intro Date }
  Market {Population }
    East (+) (UDAs: Major Market)
      New York (+) (UDAs: Major Market) {Population:21000000}
      Massachusetts (+) (UDAs: Major Market) {Population:9000000}
      Florida (+) (UDAs: Major Market) {Population:15000000}
      Connecticut (+) (UDAs: Small Market) {Population:6000000}
      New Hampshire (+) (UDAs: Small Market) {Population:3000000}
    West (+)
    South (+) (UDAs: Small Market)
    Central (+) (UDAs: Major Market)
    test (+)
      New York (+) (Shared Member)
      Massachusetts (+) (Shared Member)
      Florida (+) (Shared Member)
      Connecticut (+) (Shared Member)
      New Hampshire (+) (Shared Member)
  Scenario (Label Only)
  Caffeinated Attribute
  Ounces Attribute
  Pkg Type Attribute
  Population Attribute
  Intro Date Attribute
```

If you retrieved just the children of East, all results would be from stored members because Essbase retrieves stored members by default.

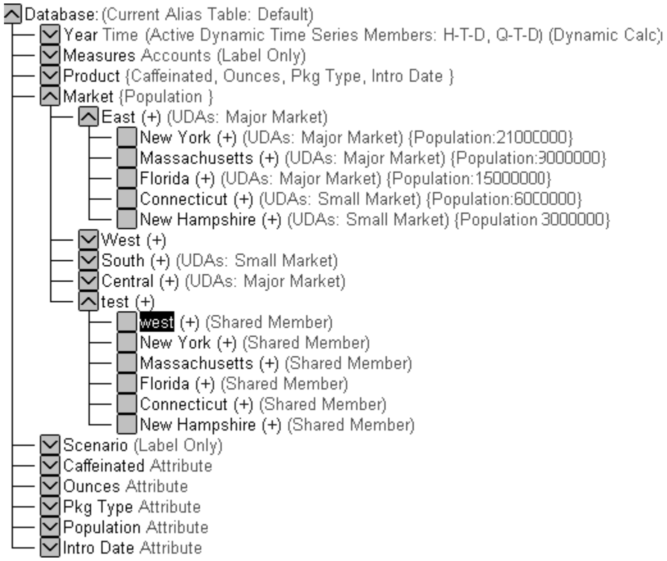
If, however, you retrieved data with test's children above it in the spreadsheet, Essbase would retrieve the shared members:

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

If you moved test above its last two children, Essbase would retrieve the first three children as shared members, but the last two as stored members. Similarly, if you inserted a member in the middle of the list above which was not a sibling of the shared members (for example, California inserted between Florida and Connecticut), then Essbase would retrieve shared members only between the non-sibling and the parent (in this case, between California and test).

Example: Retrieval with Crossed Generation Shared Members

You could modify the Sample Basic outline to create a shared member whose stored member counterpart was a sibling to its own parent:



```
Database: (Current Alias Table: Default)
    Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
    Measures Accounts (Label Only)
    Product {Caffeinated, Ounces, Pkg Type, Intro Date }
    Market {Population }
        East (+) (UDAs: Major Market)
            New York (+) (UDAs: Major Market) {Population:21000000}
            Massachusetts (+) (UDAs: Major Market) {Population:3000000}
            Florida (+) (UDAs: Major Market) {Population:15000000}
            Connecticut (+) (UDAs: Small Market) {Population:6000000}
            New Hampshire (+) (UDAs: Small Market) {Population 3000000}
        West (+)
        South (+) (UDAs: Small Market)
        Central (+) (UDAs: Major Market)
        test (+)
            west (+) (Shared Member)
            New York (+) (Shared Member)
            Massachusetts (+) (Shared Member)
            Florida (+) (Shared Member)
            Connecticut (+) (Shared Member)
            New Hampshire (+) (Shared Member)
    Scenario (Label Only)
    Caffeinated Attribute
    Ounces Attribute
    Pkg Type Attribute
    Population Attribute
    Intro Date Attribute
```

If you created a spreadsheet with shared members in this order, Essbase would retrieve all the shared members, except it would retrieve the stored member West, not the shared member west:

```
west
New York
Massachusetts
Connecticut
New Hampshire
test
```

This happens because test is a parent of west and a sibling of west's stored member counterpart, West.

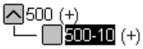
Implied Sharing

The shared member property defines a shared data relationship explicitly. Some members are shared even if you don't explicitly set them as shared. These members are said to be *implied shared members*.

Essbase assumes (or implies) a shared member relationship in the following situations:

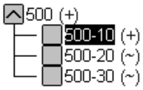
- **A parent has only one child.** In this situation, the parent and the child contain the same data. Essbase ignores the consolidation property on the child and stores the data only once—thus the parent has an implied shared relationship with the child. In Figure 83, for example, the parent 500 has only one child, 500-10, so the parent shares the value of that child.

  Figure 83: Implied Sharing of a Parent with One Child

- **A parent has only one child that consolidates to the parent.** If the parent has four children, but three of them are marked as no consolidation, then the parent and child that consolidates contain the same data. Essbase ignores the consolidation property on the child and stores the data only once—thus the parent has an implied shared relationship with the child. In Figure 84, for example, the parent 500 has only one child, 500-10, that rolls up to it. The other children are marked as No Consolidate(~), so the parent implicitly shares the value of 500-10.

Figure 84: Implied Sharing of a Parent with Multiple Children



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

If you do not want a member to be shared implicitly, mark the parent as Never Share so that the data is duplicated, and is not shared. See “Setting the Shared Member Property” on page 219.

Setting the Shared Member Property

➤ To tag a member as shared in the Outline Editor:

1. Select the shared member.
2. Click the Shared Member button, . “Shared Member” displays next to the member name.
3. If the shared member and the actual member roll up to multiple parents, their values are counted twice in a consolidation of the database. If you want to prevent this, select the shared member and click the No Consolidate button, . The tilde character (~) displays next to the member’s name.

Setting UDAs

You can create your own user-defined attributes for members. A *user-defined attribute (UDA)* is a word or phrase about the member. For example, you might create a UDA called Debit. Use UDAs in:

- Calculation scripts. After you define a UDA, you can query a member for its UDA in a calculation script. For example, you could multiply all members with the UDA Debit by -1 so that they display as either positive or negative (depending on how the data is currently stored). See Chapter 30, "Developing Calculation Scripts."
- Data loading. You can change the sign of the data as it is loaded into the database based on its UDA. See "Flipping Field Signs" on page 639.

If you want to perform a calculation, selectively retrieve data based on attribute values, or provide full crosstab, pivot, and drill-down support in the spreadsheet, create attribute dimensions instead of UDAs. See "Differences Between Attributes and UDAs" on page 234.

Rules for UDAs

Follow these rules when creating UDAs:

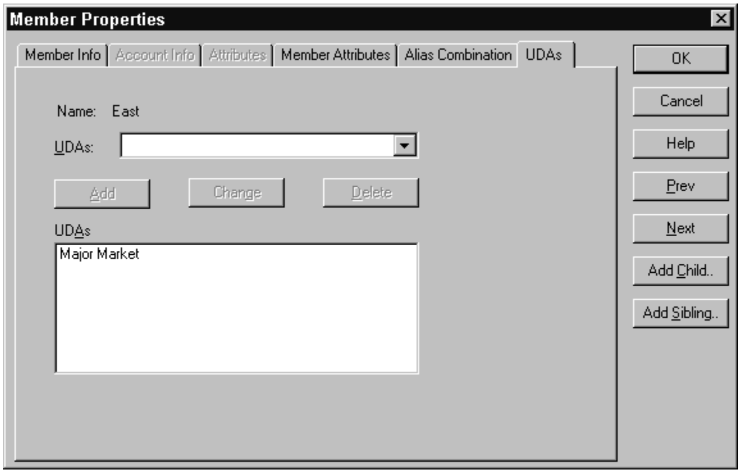
- You can define multiple UDAs per member.
- You cannot set the same UDA twice for one member.
- You can set the same UDA for different members.
- A UDA name can be the same as a member, alias, level, or generation name. When you name UDAs, follow the same naming rules as for members. See "Rules for Naming Dimensions and Members" on page 179.
- You cannot create a UDA on shared members.
- You cannot create a UDA on members of attribute dimensions.
- A UDA applies to the specified member only. Descendants and ancestors of the member don't automatically receive the same UDA.

Creating UDAs

➤ To create a UDA in the Outline Editor:

1. Select the member to create the UDA for; for example, East.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Member Properties dialog box and click the UDAs tab.

Figure 85: UDAs Page



Name: East

UDAs:

UDAs
Major Market

3. Enter the UDA in the UDAs text box or select it from the list box; for example, Major Market.
4. Click the Add button to add it to the UDAs list. The UDAs box lists all UDAs for the selected member.
5. Click OK.

Setting Comments on Dimensions and Members

You can add comments to dimensions and members. The Outline Editor displays these comments to the right of the dimension or member in the following format:

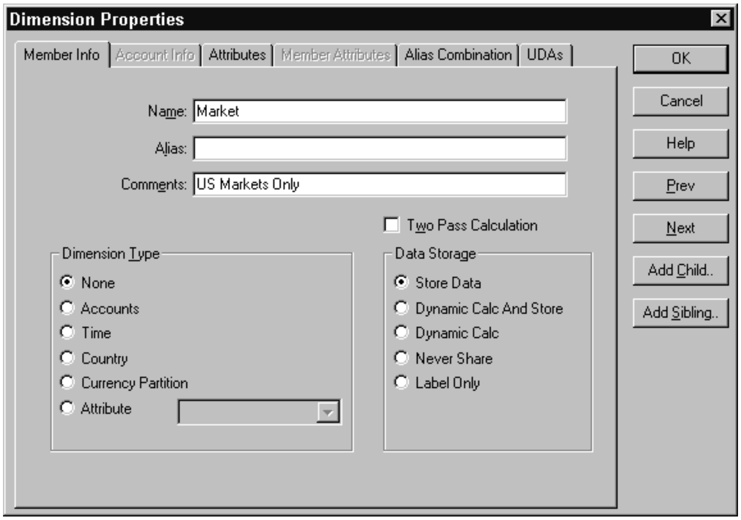
```
/* comment */
```

➤ To add comments to dimensions or members:

1. Select the dimension or member that you'd like to add the comment to; for example, Market.

2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.

Figure 86: Comments in the Dimension Properties Dialog Box



Name: Market

Alias:

Comments: US Markets Only

□ Two Pass Calculation

Dimension Type
● None
○ Accounts
○ Time
○ Country
○ Currency Partition
○ Attribute

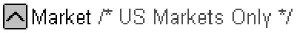
Data Storage
● Store Data
○ Dynamic Calc And Store
○ Dynamic Calc
○ Never Share
○ Label Only

3. Enter the comment in the Comment text box; for example, US Markets Only. A comment can be up to 255 characters long.

4. Click OK.

The comment is listed next to the dimension or member in the Outline Editor:

Figure 87: Sample Dimension Comment



Market /* US Markets Only */

Setting Formulas for Dimensions and Members

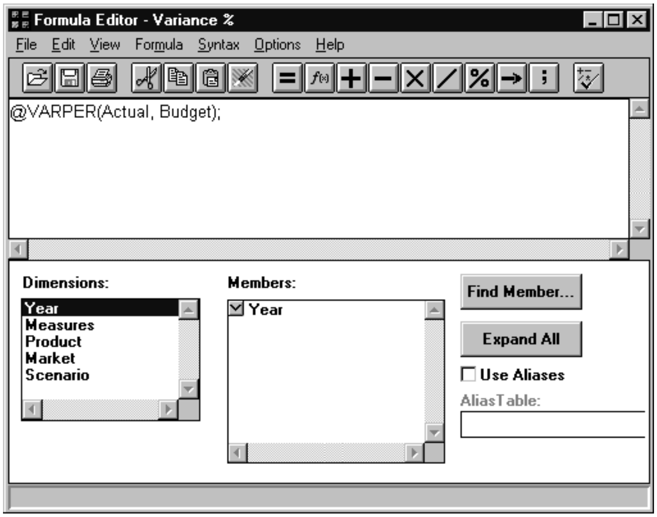
You can apply formulas to standard dimensions and members. You cannot set formulas for attribute dimensions and their members. The formula determines how Essbase calculates the outline data. For more information about formulas, see Chapter 25, “Developing Formulas.”

➤ To add a formula to a dimension or member:

1. Select the dimension or member to which to add the formula; for example, Variance %.

2. Click the Formula button, , or press the = sign on the keyboard to open the Formula Editor.

Figure 88: Formula Editor



```
@VARPER(Actual, Budget);
```

3. Type the formula into the edit field; for example, @VARPER(Actual, Budget).
4. Select File > Close to close the Formula Editor. Essbase checks the formula's syntax. If the syntax is correct, the formula displays next to the member's name. If not, open the Formula Editor and check the syntax.

For more information about using the Formula Editor, see "Example: Creating a Simple Formula" on page 688.

Chapter

9

Working with Attributes

Attributes describe characteristics of data such as the size and color of products. Through attributes you can group and analyze members of dimensions based on their characteristics.

This chapter describes how to create and manage attributes in an OLAP Server outline. It contains the following sections:

- "About Attributes" on page 226
- "Attribute Design Considerations" on page 236
- "Working with Member Names in Attribute Dimensions" on page 239
- "Defining Attributes Manually" on page 249
- "Calculating Attribute Data" on page 253

You can find other information about attributes in relevant sections of this book.

Information Needed	More Information
Defining attributes through dimension build	• Chapter 18, "Introducing Dynamic Dimension Building" • Chapter 19, "Building Dimensions Using a Rules File"
Using attributes in partitions	• Chapter 13, "Designing Partitioned Applications" • Chapter 14, "Building and Maintaining Partitions"
Using attributes in report writer	• Chapter 35, "Developing Report Scripts" • Chapter 36, "Examples of Report Scripts"

About Attributes

You can use the Essbase attribute feature to retrieve and analyze data not only from the perspective of dimensions, but also in terms of characteristics, or attributes, of those dimensions. For example, you can analyze product profitability based on size or packaging, and you can make more effective conclusions by incorporating into your analysis market attributes such as the population size of each market region.

Such an analysis could tell you that decaffeinated drinks sold in cans in small (less than 6,000,000-population) markets are less profitable than you anticipated. For more details, you can filter your analysis by specific attribute criteria, including minimum or maximum sales and profits of different products in similar market segments.

Here are a few ways analysis by attribute provides depth and perspective, supporting better-informed decisions.

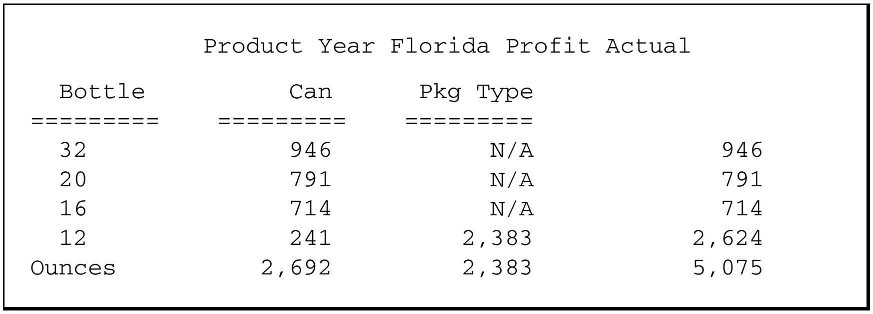
- You can select, aggregate, and report on data based on common features (attributes).
- By defining attributes as having a text, numeric, Boolean, or date type, you can filter (select) data using type-related functions such as AND, OR, and NOT operators and <, >, and = comparisons.
- You can use the numeric attribute type to group statistical values by attribute ranges; for example, population groupings such as <500,000, 500,000–1,000,000, and >1,000,000.
- Through the Attribute Calculations dimension automatically created by Essbase, you can view sums, counts, minimum or maximum values, and average values of attribute data. For example, when you enter Avg and Bottle into a spreadsheet, Essbase retrieves calculated values for average sales in bottles for all the column and row intersections on the sheet.
- You can perform calculations using numeric attribute values in calculation scripts and member formulas; for example, to determine profitability by ounce for products sized by the ounce.

- You can create crosstabs of attribute data for the same dimension, and you can pivot and drill down for detail data in spreadsheets.

  An attribute *crosstab* is a report or spreadsheet showing data consolidations across attributes of the same dimension. For example, the crosstab in Figure 89 displays product packaging as columns and the product size in ounces as rows. At their intersections, you see the profit for each combination of package type and size.

  From this information, you can see which size-packaging combinations were most profitable in the Florida market.

Figure 89: Crosstab Example



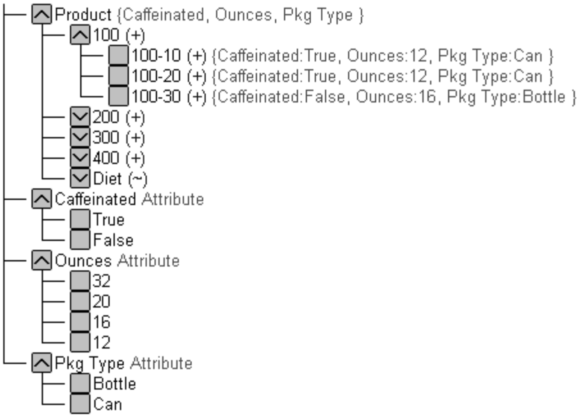
```
             Product Year Florida Profit Actual

  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
```

About Attribute Dimensions

In the Sample Basic database, products have attributes that are characteristics of the products. For example, products have an attribute that describes their packaging. In the outline, you see this as two dimensions, the Products dimension, and the Pkg Type attribute dimension that is associated with it. An *attribute* dimension has the word Attribute next to its name in the outline.

Figure 90 shows part of the Sample Basic outline featuring the Product dimension and three attribute dimensions, Caffeinated, Ounces, and Pkg Type.

Figure 90: Outline Showing Base and Attribute Dimensions



```
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
```

In the outline, to the right of the Product dimension, the terms Caffeinated, Ounces, and Pkg Type, enclosed in braces, show that these attribute dimensions are associated with the Product dimension.

A *standard* dimension is any dimension that is not an attribute dimension. When an attribute dimension is associated with a standard dimension, the standard dimension is the *base* dimension for that attribute dimension. In the outline in Figure 90, the Product dimension is the base dimension for the Caffeinated, Ounces, and Pkg Type attribute dimensions.

Note: Attribute dimensions and members are Dynamic Calc. This means Essbase calculates attribute information at retrieval time. Attribute data is not stored in the database.

Members of Attribute Dimensions

Members of an attribute dimension are potential attributes of the members of the associated base dimension. After you associate a base dimension with an attribute dimension, you associate members of the base dimension with members of the associated attribute dimension. The Market dimension member Connecticut is associated with the 6000000 member of the Population attribute dimension. That makes 6000000 an attribute of Connecticut.

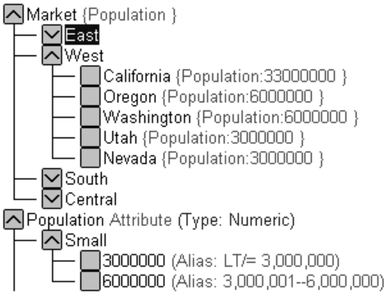
In the outline, the information enclosed in braces next to a base dimension member shows the attributes of that member. In Figure 90, the entry next to product 100-10, {Caffeinated:True, Ounces:12, Pkg Type:Can}, shows that product 100-10 has three attributes: product 100-10 has caffeine, it is sold in 12-ounce containers, and the containers are cans.

There are several important rules regarding members of attribute dimensions and their base dimensions.

- All base dimension members associated with members of a particular attribute dimension must be at the same level.

  For example, in Figure 91, all Market dimension members that have Population attributes are at level 0. You cannot associate East, which is a level 1 member, with a Population attribute since the other members of the Market dimension that have Population attributes are level 0 members.

  Figure 91: Association of Attributes with the Same Level Members of the Market Dimension



```
Market {Population }
  East
  West
    California {Population:33000000 }
    Oregon {Population:6000000 }
    Washington {Population:6000000 }
    Utah {Population:3000000 }
    Nevada {Population:3000000 }
  South
  Central
Population Attribute (Type: Numeric)
  Small
    3000000 (Alias: LT/= 3,000,000)
    6000000 (Alias: 3,000,001--6,000,000)
```

- The level 0 members of attribute dimensions are the only members that you can associate with base dimension members.

  For example, in the Population attribute dimension, you can associate only level 0 members such as 3000000, 6000000, and 9000000, with members of the Market dimension. You cannot associate a level 1 member such as Small.

  The name of the level 0 member of an attribute dimension is the attribute value. The only members of attribute dimensions that have attribute values are level 0 members.

  You can use the higher-level members of attribute dimensions to select and group data. For example, you can use Small, the level 1 member of the Population attribute dimension, to retrieve sales in both the 3000000 and 6000000 population categories.

- A base dimension member can have many attributes, but only one attribute from each particular attribute dimension.

  For example, product 100-10 can have size and packaging attributes, but only one size and only one type of packaging.

- Essbase does not support attributes for Hybrid Analysis-enabled members.

You can use attribute values in calculations in the following comparisons:

```
>  (greater than)
>= (greater than or equal to)
<  (less than)
<= (less than or equal to)
== (evaluates to the same value as)
<> (not equal to)
!= (not equal to)
IN
```

Attribute Types

Attribute dimensions have a text, numeric, Boolean, or date type that enables different functions for grouping, selecting, or calculating data. Although assigned at the dimension level, the attribute type applies only to level 0 members of the attribute dimension.

- The default attribute type is text. Text attributes enable the basic attribute member selection and attribute comparisons in calculations. When you perform such comparisons, Essbase compares characters. For example, the package type Bottle is less than the package type Can because B precedes C in the alphabet. In Sample Basic, Pkg Type is an example of a text attribute dimension.

- The names of level 0 members of *numeric* attribute dimensions are numeric values. You can include the names (values) of numeric attribute dimension members in calculations. For example, you can use the number of ounces specified in the Ounces attribute to calculate profit per ounce for each product.

  You can also associate numeric attributes with ranges of base dimension values; for example, to analyze product sales by market population groupings—states with 3,000,000 population or less in one group, states with a population between 3,000,001 and 6 million in another group, and so on. See "Assigning Member Names to Ranges of Values" on page 247.

- All Boolean attribute dimensions in a database contain only two members. The member names must match the settings for the database; for example, True and False. See "Changing the Member Names for Boolean Attribute Dimensions" on page 244.
- You can use date attributes to specify the date format—month-day-year or day-month-year—and to sequence information accordingly. See "Setting the Member Name Format of Date Attribute Dimensions" on page 246. You can use date attributes in calculations. For example, you can compare dates in a calculation that selects product sales from markets established since 10-12-1999.

  Essbase supports date attributes from January 1, 1970 through January 1, 2038.

Differences Between Attribute and Standard Dimensions

In general, attribute dimensions and their members are similar to standard dimensions and members. You can provide aliases and member comments for attributes. Attribute dimensions can include hierarchies and you can name generations and levels. You can perform the same spreadsheet operations on attribute dimensions and members as you can on standard dimensions and members; for example, to analyze data from different perspectives, you can retrieve, pivot, and drill down in the spreadsheet.

Table 14 describes major differences between attribute and standard dimensions and their members.

Table 14: Differences Between Attribute and Standard Dimensions

	Attribute Dimensions	**Standard Dimensions**
Storage	Must be sparse. Their base dimensions must also be sparse.	Can be dense or sparse
Storage property	Dynamic Calc only, therefore not stored in the database. The outline does not display this property.	Can be Store Data, Dynamic Calc And Store, Dynamic Calc, Never Share, or Label Only
Position in outline	Must be the last dimensions in the outline	Must be ahead of all attribute dimensions in the outline
Partitions	Cannot be defined along attribute dimensions, but you can use attributes to define a partition on a base dimension.	Can be defined along standard dimensions.
Formulas (on members)	Cannot be associated	Can be associated
Shared members	Not allowed	Allowed
Alias combinations	Cannot include attribute dimensions or members	Can include standard dimensions and members
Two-pass calculation member property	Not available	Available
Two-pass calculation with run-time formula	If member formula must be executed at run time and is tagged two-pass, calculation skips the member and issues warning message. Run-time dependent functions include: @CURRMBR, @PARENT, @PARENTVAL, @MDPARENTVAL, @ANCEST, @ANCESTVAL, and @MDANCESTVAL.	Calculation is performed on standard members with run-time formulas and tagged two-pass.

Table 14: Differences Between Attribute and Standard Dimensions (Continued)

	Attribute Dimensions	**Standard Dimensions**
Two-pass, multiple dimensions: Calculation order	Order of calculation of members tagged two-pass depends on order in outline: last dimension calculated last.	Calculation result is not dependent on outline order for members tagged two-pass in more than one dimension.
Two-pass calculation with no member formula	Calculation skipped, warning message issued. Thus member intersection of two-pass tagged members and upper level members may return different results from calculation on standard dimensions.	Available
Dense dynamic calc members in non-existing stored blocks	Calculation skips dense dimensions if they are on any non-existing stored block. To identify non-existing stored blocks, export the database or run query to find out whether block has any data.	Available
UDAs on members	Not allowed	Allowed
Consolidations	For all members, calculated through the Attribute Calculations dimension members: Sum, Count, Min, Max, and Avg.	Consolidation operation indicated by assigning the desired consolidation symbol to each member
Member selection facilitated by Level 0 member typing	Available types include: text, numeric, Boolean, and date.	All members treated as text.
Associations	Must be associated with a base dimension	N/A
Spreadsheet drill-downs	List the base dimension data associated with the selected attribute. For example, drilling down on the attribute Glass displays sales for each product packaged in glass, where Product is the base dimension for the Pkg Type attribute dimension.	List lower or sibling levels of detail in the standard dimensions. For example, drilling down on QTR1 displays a list of products and their sales for that quarter.

Differences Between Attributes and UDAs

Attributes and UDAs both enable analysis based on characteristics of the data. Attributes provide much more capability than UDAs. Table 15 compares them. Checkmarks indicate the feature supports the corresponding capability.

Table 15: Comparing Attributes and UDAs

Capability	Attributes Feature	UDAs Feature
Data Storage		
You can associate with sparse dimensions.	✓	✓
You can associate with dense dimensions.		✓
Data Retrieval		
You can group and retrieve consolidated totals by attribute or UDA value. For example, associate the value High Focus Item to various members of the Product dimension and use that term to retrieve totals and details for just those members.	✓ Simple	✓ More difficult to implement, requiring additional calculation scripts or commands
You can categorize attributes in a hierarchy and retrieve consolidated totals by higher levels in the attribute hierarchy; for example, if each product has a specific size attribute such as 8, 12, 16, or 32, and the sizes are categorized as small, medium, and large. You can view the total sales of small products.	✓	✓ More difficult to implement
You can create crosstab views displaying aggregate totals of attributes associated with the same base dimension.	✓ You can show a crosstab of all values of each attribute dimension.	✓ You can only retrieve totals based on specific UDA values.

Table 15: Comparing Attributes and UDAs (Continued)

Capability	Attributes Feature	UDAs Feature
You can use Boolean operators AND, OR, and NOT with attribute and UDA values to further refine a query. For example, you can select decaffeinated drinks from the 100 product group.	✓	✓
Because attributes have a text, Boolean, date, or numeric type, you can use appropriate operators and functions to work with and display attribute data. For example, you can view sales totals of all products introduced after a specific date.	✓	
You can group numeric attributes into ranges of values and let the dimension building process automatically associate the base member with the appropriate range. For example, you can group sales in various regions based on ranges of their populations: less than 3 million, between 3 and 6 million, and so on.	✓	
Through the Attribute Calculations dimension, you can view aggregations of attribute values as sums, counts, minimums, maximums, and averages.	✓	
You can use an attribute in a calculation that defines a member. For example, you can use the weight of a product in ounces to define the profit per ounce member of the Measures dimension.	✓	
You can retrieve specific base members using attribute-related information.	✓ Powerful conditional and value-based selections	✓ Limited to text string matches only
Data Conversion		
Based on the value of a UDA, you can change the sign of the data as it is loaded into the database. For example, you can reverse the sign of all members with the UDA Debit.		✓

Table 15: Comparing Attributes and UDAs (Continued)

Capability	Attributes Feature	UDAs Feature
Calculation Scripts		
You can perform calculations on a member if its attribute or UDA value matches a specific value. For example, you can increase the price by 10% of all products with the attribute or UDA of Bottle.	✓	✓
You can perform calculations on base members whose attribute value satisfies conditions that you specify. For example, you can calculate the Profit per Ounce of each base member.	✓	

Introducing the Attribute Calculations Dimension

When you create the first attribute dimension in the outline, Essbase also creates the Attribute Calculations dimension comprising five members with the default names Sum, Count, Min (minimum), Max (maximum), and Avg (average). You can use the members of the Attribute Calculations dimension to retrieve calculated information; for example, the average sales of cola within different market population ranges.

The Attribute Calculations dimension is not visible in the outline. You can see it wherever you select dimension members, such as in the Member Selection dialog box and in the Query Designer in the Spreadsheet Add-in.

For more information about the Attribute Calculations dimension, see “Attribute Calculations Dimension” on page 254 and “Changing Member Names of the Attribute Calculations Dimension” on page 243.

Attribute Design Considerations

Essbase provides more than one way to design attribute information into a database. Most often, defining characteristics of the data through attribute dimensions and their members is the best approach.

Using Attribute Dimensions

For the most flexibility and functionality, use attribute dimensions to define attribute data. Using attribute dimensions provides the following features:

- Sophisticated, flexible data retrieval

  You can view attribute data only when you want to, you can create meaningful summaries through crosstabs, and using type-based comparisons, you can selectively view just the data you want to see.

- Additional calculation functionality

  Not only can you perform calculations on the names of members of attribute dimensions to define members of standard dimensions, you can also access five different types of consolidations of attribute data: sums, counts, averages, minimums, and maximums.

- Economy and simplicity

  Because attribute dimensions are sparse, Dynamic Calc, they are not stored as data. Compared to using shared members, outlines using attribute dimensions contain fewer members and are easier to read.

For more about attribute features, see "About Attributes" on page 226.

Using Alternative Design Approaches

In some situations, you should consider one of the following approaches:

- UDAs—Although UDAs provide less flexibility than attributes, you can use them to group and retrieve data based on its characteristics. See "Differences Between Attributes and UDAs" on page 234.
- Shared members—For example, to include a seasonal analysis in the Year dimension, repeat the months as shared members under the appropriate season; Winter: Jan (shared member), Feb (shared member), and so on. A major disadvantage of using shared members is that the outline becomes very large if the categories repeat a lot of members.
- Standard dimensions and members—Additional standard dimensions provide flexibility, but they add storage requirements and complexity to a database. See "Analyzing Database Design" on page 110.

Table 16 describes situations where you might consider one of these alternative approaches for managing attribute data in a database.

Table 16: Considering Alternatives to Attribute Dimensions

Situation	Alternative to Consider
Analyze attributes of dense dimensions	UDAs or shared members.
Perform batch calculation of data	Shared members or members of separate, standard dimensions.
Define the name of a member of an attribute dimension as a value as that results from a formula	Shared members or members of separate, standard dimensions
Define attributes that vary over time	Members of separate, standard dimensions. For example, to track product maintenance costs over a period of time, the age of the product at the time of maintenance is important. However, using the attribute feature you could associate only one age with the product. You need multiple members in a separate dimension for each time period that you want to track.
Minimize retrieval time with large numbers of base-dimension members	Batch calculation with shared members or members of separate, standard dimensions.

Outline Performance Considerations

Outline layout and content can affect attribute calculation and query performance. For general outline design guidelines, see "Designing an Outline to Optimize Performance" on page 124. To optimize attribute query performance, consider the following design tips:

- Ensure that attribute dimensions are the only sparse Dynamic Calc dimensions in the outline.
- Locate sparse dimensions after dense dimensions in the outline. Place the most-queried dimensions at the beginning of the sparse dimensions and attribute dimensions at the end of the outline. In most situations, the base dimensions are the most queried dimensions.

For information on optimizing calculation of outlines containing attributes, see "Calculation and Retrieval Performance Considerations" on page 259.

Working with Member Names in Attribute Dimensions

All member names in an outline must be unique. When you use the attribute feature, Essbase establishes some default member names in some places. These default names might duplicate names that already exist in the outline. Use the Settings > Attribute Member Names command to change these system-defined names for the database. You can also use this command to establish other settings for members of attribute dimensions in the database.

Define the member name settings before you define or build the attribute dimensions. Changing the settings after the attribute dimensions and members are defined could result in invalid member names.

The following sections describe how to work with the names of members of attribute dimensions:

- "Defining a Prefix or Suffix Format for Member Names of Attribute Dimensions" on page 240
- "Changing Member Names of the Attribute Calculations Dimension" on page 243
- "Changing the Member Names for Boolean Attribute Dimensions" on page 244
- "Setting the Member Name Format of Date Attribute Dimensions" on page 246
- "Assigning Member Names to Ranges of Values" on page 247

Note: If you partition on outlines containing attribute dimensions, the name format settings of members described in this section must be identical in source and target outlines.

Tip: You can use the GETATTRIBUTESPECS command in ESSCMD to view the attribute member name format specifications for the database. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Defining a Prefix or Suffix Format for Member Names of Attribute Dimensions

The names of members of Boolean, date, and numeric attribute dimensions are values. It is possible to encounter duplicate attribute values in different attribute dimensions.

- Boolean example—If you have more than one Boolean attribute dimension in an outline, the two members of each of those dimensions have the same names, by default, True and False.
- Date example—If you have more than one date attribute dimension, some member names in both dimensions could be the same. For example, the date that a store opens in a certain market could be the same as the date a product was introduced.
- Numeric example—12 can be the attribute value for the size of a product and 12 could also be the value for the number of packing units for a product. This would result in two members with the same name: 12.

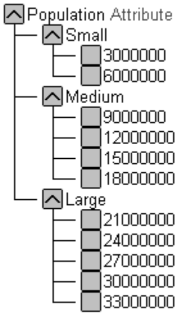
Because Essbase does not allow duplicate member names, Application Manager helps you to define unique names by attaching a prefix or suffix to member names in Boolean, date, and numeric attribute dimensions in the outline. For example, by setting member names of attribute dimensions to include the dimension name as the suffix, attached by an underscore, the member value 12 in the Ounces attribute dimension assumes the unique, full attribute member name, 12_Ounces.

By default, Essbase assumes that no prefix or suffix is attached to the names of members of attribute dimensions.

Note: The convention that you select applies to the level 0 member names of all numeric, Boolean, and date attribute dimensions in the outline.

For the examples in the following instructions, consider the Population attribute dimension:

Figure 92: The Population Attribute Dimension and Members



➤ To define prefixes or suffixes for the names of members of Boolean, date, and numeric attribute dimensions:

1. From Application Manager, open the outline.
2. Select the Settings > Attribute Member Names menu command.

   The Sample box shows the format of member names in numeric, Boolean, and date attribute dimensions.
3. In the Value option group, select the source of the value that Essbase attaches to the attribute member name in the outline.
   - **None** uses the member name in the outline as the full name. Nothing is attached; for example, 3000000.
   - **Parent** attaches the immediate parent; for example, Small_3000000.
   - **Grandparent** attaches the parent's parent; for example, Population_3000000.
   - **Ancestor** attaches all generations; for example, Population_Small_3000000.
   - **Dimension** attaches the attribute dimension name; for example, Population_3000000.

   If you select a value other than None, Essbase assumes the other format options are Prefix and "_(underscore)."

4. In the Separator option group, select the separator character to insert between the attribute member name in the outline and the attached value.

   - **_(underscore)** inserts an underscore between each value in the prefix or suffix; for example, Population_Small_3000000.
   - **| (pipe)** inserts a pipe between each value in the prefix or suffix; for example, Population|Small|3000000.
   - **^ (caret)** inserts a caret between each value in the prefix or suffix; for example, Population^Small^3000000.

5. In the Prefix/Suffix option group, select whether to attach a prefix or a suffix to the attribute member name in the outline.

   - **Prefix** attaches the value selected in the Value option group before the attribute member name; for example, Population_3000000.
   - **Suffix** attaches the value selected in the Value option group after the attribute member name; for example, 3000000_Population.

6. Click OK.

Note: The outline does not show the full attribute names. You can see and use the full attribute names anywhere you select members, such as when you define partitions or select information to be retrieved.

Tip: You can use the GETATTRINFO command in ESSCMD to view the dimension, attribute value and attribute type of a specific attribute member. For example, to view attribute name information for the Caffeinated_True attribute:

```
GETATTRINFO "Caffeinated_True"
```

See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Changing Member Names of the Attribute Calculations Dimension

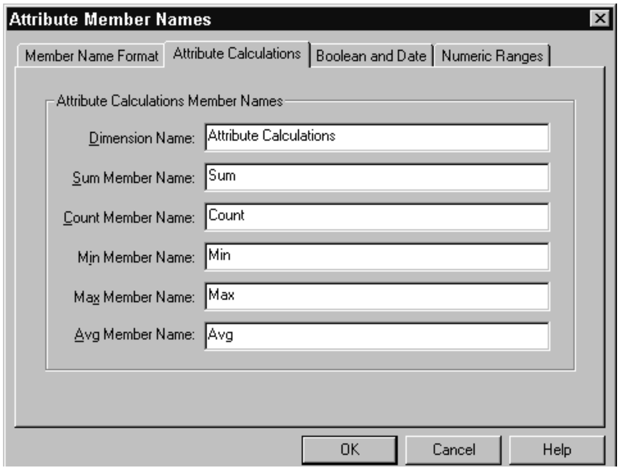
To avoid duplicating names in an outline, you may need to change the name of the Attribute Calculations dimension or its members. For more information about the Attribute Calculations dimension, see “Attribute Calculations Dimension” on page 254.

Regardless of the name that you use for a member, its function remains the same. For example, the second (Count) member always counts, no matter what you name it.

➤ To change the names of the Attribute Calculations dimension and its members:

1. From Application Manager, open the outline.
2. Select the Settings > Attribute Member Names menu command.
3. Select the Attribute Calculations tab.

Figure 93: Attribute Calculations Page in the Attribute Member Names Dialog Box



Attribute Calculations Member Names

Dimension Name: Attribute Calculations

Sum Member Name: Sum

Count Member Name: Count

Min Member Name: Min

Max Member Name: Max

Avg Member Name: Avg

4. In the Attribute Calculations Member Names option group, type the new member name for each member name that you want to change. Follow the member naming rules described in “Rules for Naming Dimensions and Members” on page 179.

   The default names for Text Box Labels:

Text Box Label	Default Name
Dimension Name	Attribute Calculations
Sum Member Name	Sum
Count Member Name	Count
Min Member Name	Min
Max Member Name	Max
Avg Member Nam	Avg

Note: The Sum member totals members based on their consolidation property or formula. For example, the Sum member uses the following formula to consolidate the profit percentages of 12-ounce products:

$$\text{Sum of Profit\% of 12-ounce products} = \frac{\text{Sum of Profit of base members with the Ounces attribute 12}}{\text{Sum of Sales of base members with the Ounces attribute 12}} * 100$$

This calculation is not the sum of all percentages for all base-dimension members with the Ounces attribute 12. For more information about each member, see “Default Attribute Calculations Members” on page 256.

5. Click OK.

Changing the Member Names for Boolean Attribute Dimensions

All Boolean attribute dimensions in a single database have two, level 0 members with the same names, by default, True and False.

Essbase enables you to change the names of members of Boolean attribute dimensions; for example, to Yes and No.

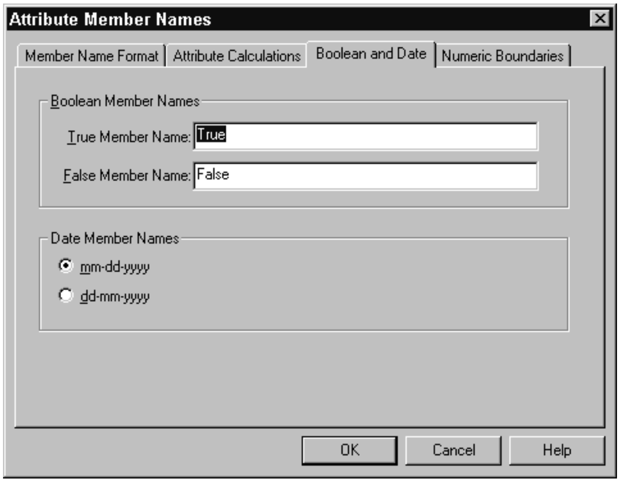
Note: Changing the Boolean member-name setting does not retroactively change the names of members of existing Boolean attribute dimensions. You must manually change the names of existing Boolean members to the names specified in the Boolean member names setting.

When you set an attribute dimension type as Boolean, Essbase automatically creates two members with the names specified in the setting. If other members exist in the Boolean attribute dimension, you must remove them.

➤ To change the member names setting of Boolean attribute dimensions in a database:

1. From Application Manager, open the outline.
2. Select the Settings > Attribute Member Names menu command.
3. Select the Boolean and Date tab.

Figure 94: Boolean and Date Page in the Attribute Member Names Dialog Box



Attribute Member Names

Boolean Member Names

True Member Name: True

False Member Name: False

Date Member Names

● mm-dd-yyyy

○ dd-mm-yyyy

4. In the Boolean Member Names option group, type the new member name for each member name that you want to change. Follow the member naming rules described in "Rules for Naming Dimensions and Members" on page 179.

   The default names:

Text Box Label	Default Name
`True Member Name`	`True`
`False Member Name`	`False`

5. Click OK.

If you have more than one Boolean attribute dimension, you must specify a prefix or suffix member name format to ensure unique member names; for example, Caffeinated_True and Caffeinated_False. See "Defining a Prefix or Suffix Format for Member Names of Attribute Dimensions" on page 240.

Setting the Member Name Format of Date Attribute Dimensions

➤ To change the format of members of date attribute dimensions at the database level:

1. From Application Manager, open the outline.
2. Select the Settings > Attribute Member Names menu command.
3. Click the Boolean and Date tab.
4. In the Date Member Names group, select a format:
   - mm-dd-yyyy displays the month before the day; for example, October 18, 1999 is displayed as 10-18-1999.
   - dd-mm-yyyy displays the day before the month; for example, October 18, 1999 is displayed as 18-10-1999.
5. Click OK.

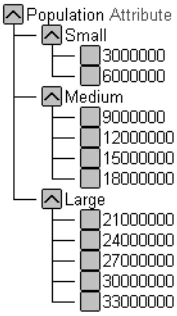
If you change the date member name format, the names of existing members of date attribute dimensions may be invalid. For example, if the 10-18-1999 member exists and you change the format to dd-mm-yyyy, outline verification will find this member invalid. If you change the date format, you must rebuild the date attribute dimensions.

Assigning Member Names to Ranges of Values

Members of numeric attribute dimensions can represent single numeric values or ranges of values

- Single value example: the member 12 in the Ounces attribute dimension represents the single numeric value 12; you associate this attribute with all 12-ounce products. The outline includes a separate member for each size; for example, 16, 20, and 32.
- Range of values example: the Population attribute dimension:

Figure 95: Population Attribute Dimension and Members



In this outline, the members of the Population attribute dimension represent ranges of population values in the associated Market dimension. The 3000000 member represents populations from zero through 3,000,000; the 6000000 member represents populations from 3,000,001 through 6,000,000; and so on. Each range includes values greater than the name of the preceding member up to and including the member value itself. A setting for the outline establishes that each numeric member represents the top of its range.

You can also define this outline setting so that members of numeric attribute dimensions are the bottoms of the ranges that they represent. For example, if numeric members are set to define the bottoms of the ranges, the 3000000 member represents populations from 3,000,000 through 5,999,999 and the 6000000 member represents populations from 6,000,000 through 8,999,999.

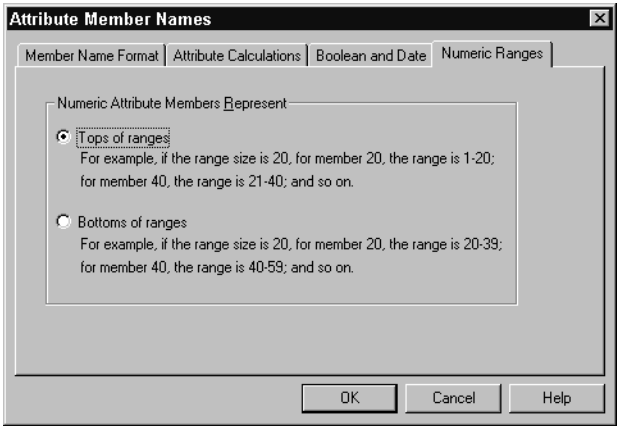
When you build the base dimension, Essbase automatically associates members of the base dimension with the appropriate attribute range. For example, if numeric members represent the tops of ranges, Essbase automatically associates the Connecticut market, with a population of 3,269,858, with the 6000000 member of the Population attribute dimension.

In the dimension build rules file, specify the size of the range for each member of the numeric attribute dimension. In the above example, each attribute represents a range of 3,000,000. See "Working With Numeric Ranges" on page 510.

➤ To define the rule for assigning numeric attribute member names to ranges of values:

1. From Application Manager, open the outline.
2. Select the Settings > Attribute Member Names menu command.
3. Select the Numeric Ranges tab.

Figure 96: Numeric Ranges Page in the Attribute Member Names Dialog Box



Attribute Member Names

Numeric Attribute Members Represent

● Tops of ranges
For example, if the range size is 20, for member 20, the range is 1-20; for member 40, the range is 21-40; and so on.

○ Bottoms of ranges
For example, if the range size is 20, for member 20, the range is 20-39; for member 40, the range is 40-59; and so on.

4. Select the option that sets whether numeric attribute values define the tops or bottoms of the ranges that they represent.

   - Tops of ranges. For example, Essbase associates the 6000000 attribute with all markets in a population area greater than the value of the next lower member of the attribute dimension and less than or equal to 6,000,000. This is the default setting.
   - Bottoms of ranges. For example, Essbase associates the 6000000 attribute with all markets in a population area equal to or greater than 6000000 and less than the value of the next higher member of the attribute dimension.

5. Click OK.

Defining Attributes Manually

When manually working with attributes, use the Outline Editor in Application Manager to perform the following dimension and member-related tasks:

- Create the attribute dimensions. See "Adding Dimensions to Outlines" on page 181. In the outline, position the attribute dimensions after all standard dimensions.
- Tag the dimensions as attribute dimensions. See "Tagging an Attribute Dimension" on page 205.
- Set each attribute dimension type as text, numeric, Boolean, or date. See "Setting the Attribute Type" on page 206.
- Add members to the attribute dimensions. See "Adding Members to Dimensions" on page 183.
- Associate base dimensions with the attribute dimensions. See "Associating Base Dimensions with Attribute Dimensions" on page 250.
- Associate base dimension members with members of the attribute dimensions. See "Associating Base Dimension Members with Attributes" on page 251.

9

Associating Base Dimensions with Attribute Dimensions

When you associate an attribute dimension with a standard dimension, the standard dimension is known as the *base* dimension for that attribute dimension.

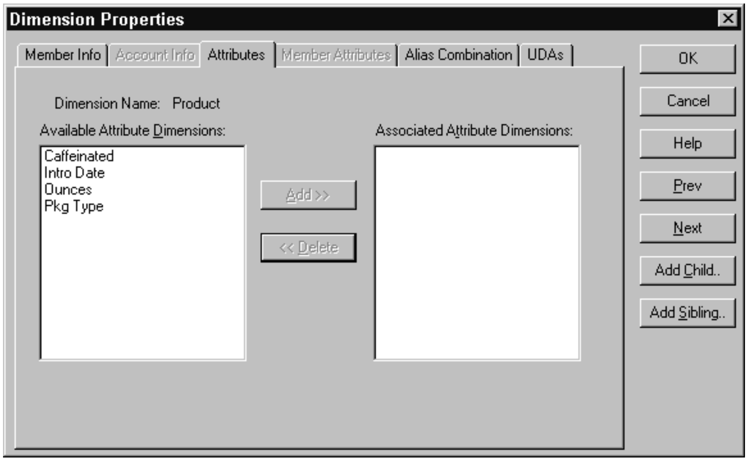
- You can only associate attribute dimensions with sparse standard dimensions.
- A standard dimension can be a base dimension for more than one attribute dimension.
- An attribute dimension can be associated with only one base dimension.

  For example, you might have a Size attribute dimension with members Small, Medium, and Large. If you associate the Size attribute dimension with the Product dimension, you cannot also associate the Size attribute dimension with the Market dimension. If you also want to track size-related information for the Market dimension, you must create another attribute dimension with a different name, for example, MarketSize, and associate the MarketSize attribute dimension with the Market dimension.

➤ To associate attribute dimensions with a standard dimension:

1. From Application Manager, open the outline.
2. Select the base dimension, for example Product.
3. Click the Data Dictionary button, ▣, press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.

Figure 97: Attributes Page in the Dimension Properties Dialog Box



Dimension Properties

Dimension Name: Product

Available Attribute Dimensions:

Caffeinated
Intro Date
Ounces
Pkg Type

Associated Attribute Dimensions:

The Attributes page displays the name of the selected base dimension and contains the following items:

- Available Attribute Dimensions list box, which displays the names of attribute dimensions that are not associated with a base dimension
- Associated Attribute Dimensions list box, which displays the names of the attribute dimensions associated with the selected base dimension
- Add button, which moves attribute dimensions from the Available Attribute Dimensions list box to the Associated Attribute Dimensions list box
- Delete button, which removes attribute dimensions from the Associated Attribute Dimensions list box, putting them back in the Available Attribute Dimensions list box

4. Use the Add and Delete buttons to move selected attribute dimension names in and out of the Associated Attribute Dimensions list box.
5. When the Associated Attribute Dimensions list box contains the names of all attribute dimensions to be associated with the base dimension, click OK.

Associating Base Dimension Members with Attributes

Attribute associations must follow these rules:

- You cannot associate multiple members from the same attribute dimension with the same base dimension member. For example, the same product cannot be associated with both bottle and can package types.
- You can associate only level-0 members of attribute dimensions.
- The base dimension members to which you associate members from a specific attribute dimension must all be at the same level.

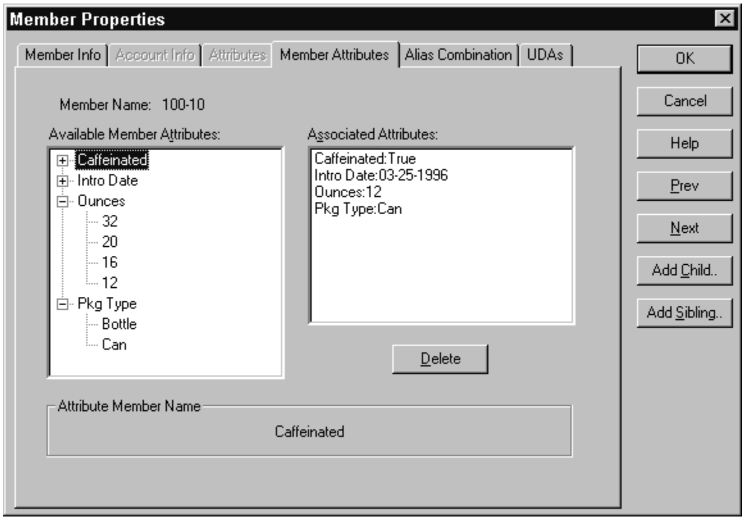
You can associate members from different attribute dimensions with the same member of a base dimension. For example, a decaffeinated cola product (100-30) sold in 16 ounce bottles has three attributes: Caffeinated:False; Ounces:16; and Pkg Type:Bottle.

After attributes are associated with base dimension members, if you cut or copy and paste base dimension members to another location in the outline, the attribute associations are lost.

➤ To associate a member of a base dimension with members of attribute dimensions:

1. From Application Manager, open the outline.
2. Select the base dimension member with which you want to associate the attributes, for example 100-10.
3. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Member Properties dialog box.
4. Select the Member Attributes tab.

Figure 98: Member Attributes Page in the Member Properties Dialog Box



Member Properties

Member Name: 100-10

Available Member Attributes:

- Caffeinated
- Intro Date
- Ounces
  - 32
  - 20
  - 16
  - 12
- Pkg Type
  - Bottle
  - Can

Associated Attributes:

Caffeinated:True
Intro Date:03-25-1996
Ounces:12
Pkg Type:Can

Attribute Member Name

Caffeinated

The Member Attributes page displays the name of the selected member of the base dimension and contains the following items:

- Available Member Attributes list box, which lists attribute dimensions associated with the base dimension to which the selected member belongs
- Associated Attributes list box, which lists attributes associated with the selected member of the base dimension
- Delete button, which deletes attribute associations from the Associated Attributes list box
- Attribute Member Name box, which displays the full member name of the selected attribute dimension member

5. Click the plus symbol, ⊞, by the name of an attribute dimension or member to display its children.

   Repeat until member names display without plus or minus symbols, ⊟, next to them. Attribute values are the level 0 members in an attribute dimension.

6. Double-click the attribute value that you want to associate with the selected base-dimension member.

   The Associated Attributes list box displays the name of the selected attribute dimension and the associated attribute value.

7. Repeat for other attribute dimensions, until the Associated Attributes list box lists the correct attribute values to be associated with the base dimension member.

   Note: Essbase does not require that each member of a base dimension be associated with a member of an attribute dimension.

8. Click **OK**.

Calculating Attribute Data

Essbase calculates attribute data dynamically at retrieval time, using members from a system-defined dimension created specifically by Essbase. Using this dimension, you can apply different calculation functions, such as a sum or an average, to the same attribute. You can also perform specific calculations on members of attribute dimensions; for example, to determine profitability by ounce for products sized by the ounce.

The following information assumes that you understand the concepts of attribute dimensions and Essbase calculations, including dynamic calculations. For more information, see Part VI, “Calculating Data.”

This section includes the following:

- “Attribute Calculations Dimension” on page 254
- “Default Attribute Calculations Members” on page 256
- “An Attribute Calculation Example” on page 257
- “Accessing Attribute Calculations Members Using the Spreadsheet” on page 258

- “Calculation and Retrieval Performance Considerations” on page 259
- “Using Attributes in Calculation Formulas” on page 259

Attribute Calculations Dimension

The Attribute Calculations dimension contains five members. You can use these members in spreadsheets or in reports to dynamically calculate and report on attribute data, such as the average yearly sales of 12-ounce bottles of cola in the West.

The attribute calculation dimension has the following properties:

- System-defined. When you create the first attribute dimension in an application, Essbase creates the Attribute Calculations dimension and its members (Sum, Count, Min, Max, and Avg). Each member represents a type of calculation to be performed for attributes. For more information, see “Default Attribute Calculations Members” on page 256.
- Label only. Like all label only dimensions, the Attribute Calculations dimension shares the value of its first child, Sum. For more information on the label only dimension property, see “Defining Dimension and Member Properties” on page 121.
- Dynamic Calc. The data in the Attribute Calculations dimension is calculated when a user requests it and is then discarded. You cannot store calculated attribute data in a database. For more information on dynamic calculations, see Chapter 28, “Dynamically Calculating Data Values.”
- Not displayed in Outline Editor. The Attribute Calculations dimension is not displayed in the Outline Editor. Members from this dimension can be viewed in spreadsheets and in reports.

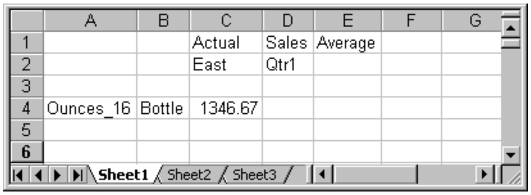
There is no consolidation along attribute dimensions. You cannot tag members from attribute dimensions with consolidation symbols (for example, + or -) or with member formulas in order to calculate attribute data. As Dynamic Calc members, attribute calculations do not affect the batch calculation in terms of time or calculation order. To calculate attribute data at retrieval time, Essbase:

1. Finds the base-dimension members that are associated with the specified attribute-dimension members present in the current query
2. Dynamically calculates the sum, count, minimum, maximum, or average for the attribute-member combination for the current query
3. Displays the results in the spreadsheet or report
4. Discards the calculated values—that is, the values are not stored in the database

Note: Essbase excludes `#MISSING` values when calculating attribute data.

For example, as shown in Figure 99, a spreadsheet user specifies two members of attribute dimensions (Ounces_16 and Bottle) and an Attribute Calculations member (Avg) in a spreadsheet report. Upon retrieval, Essbase dynamically calculates the average sales values of all products associated with these attributes for the current member combination (Actual -> Sales -> East -> Qtr1):

Figure 99: Retrieving an Attribute Calculations Member



	A	B	C	D	E	F	G
1			Actual	Sales	Average		
2			East	Qtr1			
3							
4	Ounces_16	Bottle	1346.67				
5							
6							

For more information on accessing calculated attribute data, see “Accessing Attribute Calculations Members Using the Spreadsheet” on page 258.

Default Attribute Calculations Members

The Attribute Calculations dimension contains five members used to calculate and report attribute data. These members are:

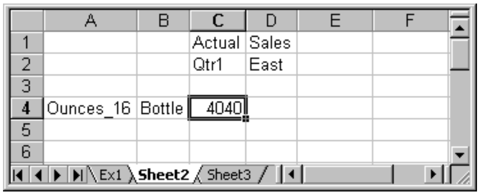
- Sum calculates a sum, or total, of the values for a member with an attribute or combination of attributes. Supports non-additive consolidations such as multiplication, and two-pass measures.
- Count calculates the number of members with the specified attribute or combination of attributes, for which a data value exists. Count includes only those members that have data blocks in existence. To calculate a count of all members with certain attributes, regardless of whether or not they have data values, use the @COUNT function in combination with the @ATTRIBUTE function. For more information, see the *Technical Reference* in the `docs` directory.
- Avg calculates a mathematical mean, or average, of the non-missing values for an specified attribute or combination of attributes (Sum divided by Count).
- Min calculates the minimum data value for a specified attribute or combination of attributes.
- Max calculates the maximum data value for a specified attribute or combination of attributes.

Note: Each of these calculations excludes `#MISSING` values.

You can change these default member names using Application Manager, subject to the same naming conventions as standard members. For more information on changing Attribute Calculations member names, see "Changing Member Names of the Attribute Calculations Dimension" on page 243.

The default calculation for attributes is Sum. If a spreadsheet user specifies a member of an attribute dimension in a spreadsheet but does not specify a member of the Attribute Calculations dimension, Essbase retrieves the sum for the specified attribute or combination of attributes. For example, in the spreadsheet view shown in Figure 100, the value in cell C4 represents the sum of sales values for the attributes Ounces_16 and Bottle for the current member combination (Actual -> Sales -> East -> Qtr1), even though the Sum member is not displayed in the sheet.

Figure 100: Retrieving the Default Attribute Calculations Member



	A	B	C	D	E	F
1			Actual	Sales		
2			Qtr1	East		
3						
4	Ounces_16	Bottle	4040			
5						
6						

An Attribute Calculation Example

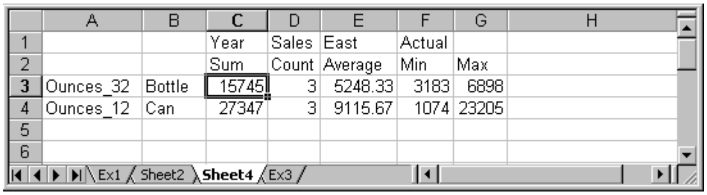
As an example of how Essbase calculates attribute data, consider the following yearly sales data for the East:

Table 17: Sample Attribute Data

Base-Dimension Member	Associated Attributes	Sales Value for Attribute-Member Combination
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

A spreadsheet report showing calculated attribute data might look like this:

Figure 101: Sample Spreadsheet with Attribute Data



	A	B	C	D	E	F	G	H
1			Year	Sales	East	Actual		
2			Sum	Count	Average	Min	Max	
3	Ounces_32	Bottle	15745	3	5248.33	3183	6898	
4	Ounces_12	Can	27347	3	9115.67	1074	23205	
5								
6								

As shown in the figure above, you can retrieve multiple Attribute Calculations members for attributes. For example, you can calculate Sum, Count, Avg, Min, and Max for 32-ounce bottles and cans.

Accessing Attribute Calculations Members Using the Spreadsheet

You can access members from the Attribute Calculations dimension in Spreadsheet Add-in. From the spreadsheet, users can view Attribute Calculations dimension members by:

- Entering members directly into a sheet
- Selecting members from the Query Designer
- Selecting members from the Member Selection dialog box
- Entering members as an EssCell parameter

For more information on accessing calculated attribute data from the spreadsheet, see the *Essbase Spreadsheet Add-in User’s Guide*.

Calculation and Retrieval Performance Considerations

Keep in mind the following calculation and retrieval performance considerations:

- The calculation order for attribute calculations is the same as the order for dynamic calculations. For more information, see “Calculation Order for Dynamic Calculations” on page 780.
- Since Essbase calculates attribute data dynamically at retrieval time, attribute calculations do not affect the performance of the overall (batch) database calculation.
- Tagging base-dimension members as Dynamic Calc may increase retrieval time.
- When a query includes the Sum member and an attribute-dimension member whose associated base-member is tagged as two-pass, retrieval time may be slow.
- To maximize attribute retrieval performance, use any of these techniques:
  - Configure the outline as described in “Outline Performance Considerations” on page 238.
  - Drill down to the lowest level of base dimensions before retrieving data. For example, in Spreadsheet Add-in, turn on the Navigate Without Data feature, drill down to the lowest level of the base dimensions included in the report, and then retrieve data.
  - When the members of a base dimension are associated with several attribute dimensions, consider grouping the members of the base dimension according to their attributes. For example, in the Sample Basic database, you could group all 8-ounce products. Grouping members by attribute may decrease retrieval time.

Using Attributes in Calculation Formulas

In addition to using the Attribute Calculations dimension to calculate attribute data, you can also use calculation formulas on members of standard or base dimensions to perform specific calculations on members of attribute dimensions; for example, to determine profitability by ounce for products sized by the ounce.

Note: You cannot associate formulas with members of attribute dimensions.

You can use these functions to perform specific calculations on attributes:

Type of Calculation	Function to Use
Generate a list of all base members with a specific attribute. For example, you can generate a list of members that have the Bottle attribute, and then increase the price for those members.	@ATTRIBUTE
Return the value of the level 0 attribute member from a numeric or date attribute dimension (@ATTRIBUTEVAL), from a boolean attribute dimension (@ATTRIBUTEBVAL), or from a text attribute dimension (@ATTRIBUTESVAL) that is associated with the base member being calculated. For example, you can return the numeric value of a size attribute (for example, 12 for the member 12 under Ounces) for the base member being calculated (for example, Cola).	@ATTRIBUTEVAL @ATTRIBUTEBVAL @ATTRIBUTESVAL
Convert a date string to numbers for a calculation. For example, you can use @TODATE in combination with the @ATTRIBUTEVAL function to increase overhead costs for stores opened after a certain date.	@TODATE
Generate a list of all base dimension members associated with attributes that satisfy the conditions that you specify. For example, you can generate a list of products that are greater than or equal to 20 ounces, and then increase the price for those products.	@WITHATTR

For syntax information and examples for these functions, see the *Technical Reference* in the `docs` directory. For an additional example using @ATTRIBUTEVAL in a formula, see “Calculating an Attribute Formula” on page 742.

Chapter 10 Creating and Managing Aliases

An alias is an alternate name for a member or shared member. This chapter describes how to create and manage alias tables in an OLAP Server outline. It contains the following sections:

Introducing Aliases and Alias Tables

You can assign one or more alternate names, or *aliases*, to a member or a shared member. Aliases can improve the readability of an outline or a report. For example, members in the Sample Basic database's Product dimension are identified both by product codes, such as 100, and by more descriptive aliases, such as Cola.

Rules for Aliases

Use the following rules when creating aliases:

- The naming conventions for aliases are the same as those for member names. See “Rules for Naming Dimensions and Members” on page 179.
- You can select different aliases based on the combination of other dimension members. An alias based on a combination of members is called an *alias combination*. This list shows members with a different alias for each market in which they are sold:

Product	Market Names	Market
100-10	The Best Cola	All, unless different for specific market
100-10	Kola	New York
100-10	Cool	Cola California

  For information on creating alias combinations, see “Creating Aliases for Member Combinations” on page 265.

- You can set more than one alias for a member using alias tables. For example, you could use different aliases for different kinds of reports—users may be familiar with 100-10 as Cola, but advertisers and executives may be familiar with it as The Best Cola. This list shows some products in the Sample Basic database that have two descriptive alias names:

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

- Each alias name in a dimension must be unique.
- To efficiently load aliases into an outline, you can provide alias names in a text file which you can use in an import alias operation. See “Importing an Alias Table” on page 271. Use the following rules when setting up the text import file:
  - The first line of the file must say, $ALT_NAME. Optionally, for your own reference, you can add one or more spaces followed by the name of the alias table.
  - The last line of the file must say, $END.

  - In the lines in between, place two values on every line. The first value must be the name of an existing member; the second value is the alias. Separate these values by one of more spaces or tabs.
  - Enclose in double quotation marks any member or alias name that contains a blank.
- Essbase does not support aliases for Hybrid Analysis-enabled members.

  For information on creating multiple aliases, see "Creating New Alias Tables" on page 267.

Introducing Alias Tables

Aliases are stored in one or more tables as part of a database outline. When you create a database outline, Essbase creates an empty alias table called Default. If you don't create any other alias tables, all of the aliases that you create are stored in the Default alias table.

If you want to create more than one alias for a member or set of members, create a new alias table and set it as your current alias table. Then create one or more new aliases. These aliases are stored in the new alias table. The last bullet in "Rules for Aliases" on page 262, for example, shows two alias tables. The first alias table is named `Default` and the second alias table is named `Long Names`. When users retrieve data, they see the aliases in whichever table that they pick. For more information, see "Creating New Alias Tables" on page 267 and "Setting Alias Tables" on page 267.

Note: You can create up to 10 alias tables for each outline.

Format of an Alias Table

An alias table contains a column listing all of the members followed by a column listing all of the aliases that correspond to those members. Alias tables must use the following format:

- $ALT_NAME must be at the top of the table.
- Database member names must be in the first field of the file. If a member name contains embedded blanks, enclose it in double quotes.
- Alias names must be in the second field of the file. If an alias name contains embedded blanks, enclose it in double quotes.

This is a sample alias table:

```
$ALT_NAME
100                Cola
"member name 2"   "Alias Name 2"
$END
```

Tip: Use the DISPLAYALIAS command in ESSCMD to view the all aliases in an alias table. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, “Automating the Production Environment” for information about ESSCMD.

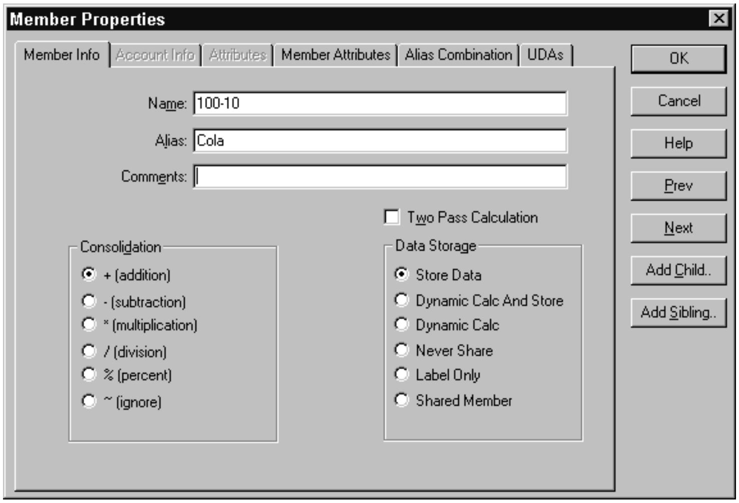
Creating Aliases for Members

You can create aliases for members. For more information on aliases, see “Introducing Aliases and Alias Tables” on page 261.

➤ To create an alias in the Outline Editor:

1. Select the member for which the alias is to be defined; for example, 100-10.
2. Click the Data Dictionary button, , press the Enter key, or select Edit > Properties to open the Member Properties dialog box.

Figure 102: Member Properties Dialog Box



Name: 100-10

Alias: Cola

Comments:

☐ Two Pass Calculation

Consolidation
- ● + (addition)
- ○ - (subtraction)
- ○ * (multiplication)
- ○ / (division)
- ○ % (percent)
- ○ ~ (ignore)

Data Storage
- ● Store Data
- ○ Dynamic Calc And Store
- ○ Dynamic Calc
- ○ Never Share
- ○ Label Only
- ○ Shared Member

3. Enter the alias name (for example, Colas) into the Alias text box on the Member Info page.

Note: Member names and alias name limits are specified in Appendix A, "Limits."

Creating Aliases for Member Combinations

You can create aliases for non-attribute members that are based on member combinations, called alias combinations. For more information on aliases, see "Introducing Aliases and Alias Tables" on page 261.

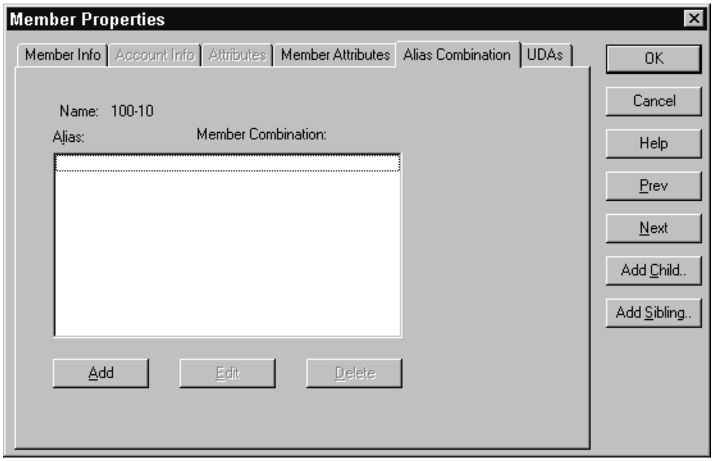
➤ To set an alias based on a member combination:

1. In Outline Editor, select the member for which the alias is to be defined; for example, 100-10.

2. Click the Data Dictionary button, ⛊, press the Enter key or select Edit > Properties to open the Member Properties dialog box.

3. Click the Alias Combinations tab.

Figure 103: Alias Combinations Page



Member Properties

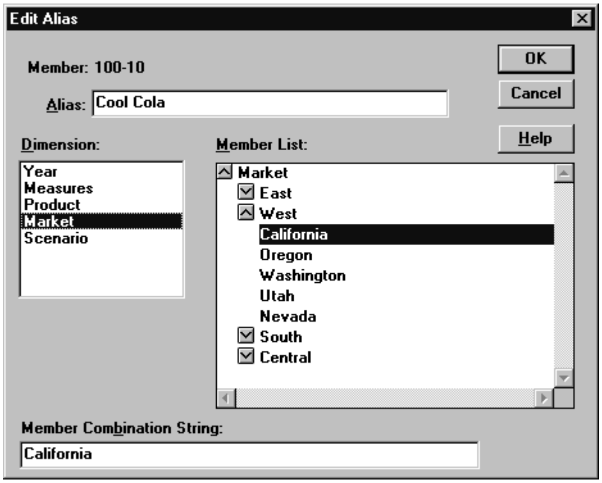
Name: 100-10

Alias:

Member Combination:

4. Click the Add button to open the Edit Alias dialog box to add a new alias to the list. Member names and alias names can be up to 79 characters long..

Figure 104: Edit Alias Dialog Box



Member: 100-10

Alias: Cool Cola

Member Combination String: California

5. Enter the alias in the Alias text box; for example, Cool Cola.

6. Either enter the member combination string in the Member Combination String text box or select the members from the Dimension and Member list boxes. To expand or contract the Member List, double-click the box in front of the member name. To add the member to the combination list, select the member.

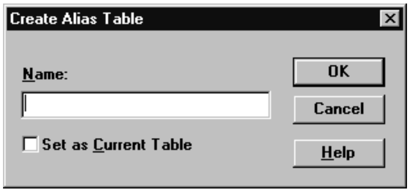
7. Click OK.

Creating New Alias Tables

➤ To create a new alias table for the selected outline:

1. Select Outline > Aliases > Create Table to open the Create Alias Table dialog box.

Figure 105: Create Alias Table Dialog Box



2. Enter the name of the new alias table in the Name text box.

   Note: You can create up to 10 alias tables for an outline.

3. If desired, set the new alias table as the current table by selecting Set as Current Table.
4. Click OK.

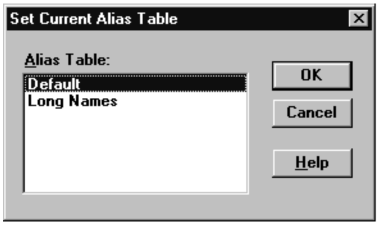
Setting Alias Tables

Essbase uses the current alias table as the source for the aliases displayed in the outline. For more information on alias tables, see “Introducing Alias Tables” on page 263.

➤ To set an alias table to be the current alias table:

1. Select Outline > Aliases > Set Table to open the Set Current Alias Table dialog box.

Figure 106: Set Current Alias Table Dialog Box



2. Select the alias table from the Alias Table list box. If the alias table that you want to use is not in the list box, you must create it. See "Creating New Alias Tables" on page 267.

3. Click OK.

Tip: You can view and set the current alias table without Application Manager:

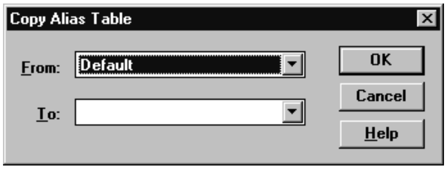
Tool	**Instructions**	**For More Information**
Administration Services	Set Alias Table dialog box	*Essbase Administration Services Online Help*
ESSCMD	`LISTALIASES` `SETALIAS`	*Technical Reference* in the `docs` directory

Copying Existing Alias Tables

➤ To copy an alias table to a different name in the same outline:

1. Select Outline > Aliases > Copy Table to open the Copy Alias Table dialog box.

   Figure 107: Copy Alias Table Dialog Box

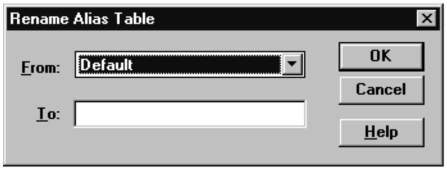
2. Select the alias table to copy from the From list box.
3. Enter the new name of the alias table to copy it to in the To list box.
4. Click OK.

Renaming Alias Tables

➤ To change the names of existing alias tables:

1. Select Outline > Aliases > Rename Table to open the Rename Alias Table dialog box.

   Figure 108: Rename Alias Table Dialog Box

2. Select the alias table to rename from the From list box.
3. Enter the new name of the alias table in the To list box.
4. Click OK.

Deleting and Clearing Alias Tables

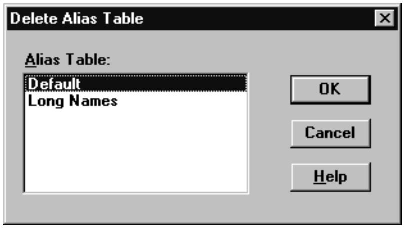
You can delete an alias table from the outline or you can clear all of the aliases from an alias table without deleting the alias table itself.

Deleting Alias Tables

➤ To delete an alias table from the outline:

1. Select Outline > Aliases > Delete Table to open the Delete Alias Table dialog box.

   Figure 109: Delete Alias Table Dialog Box

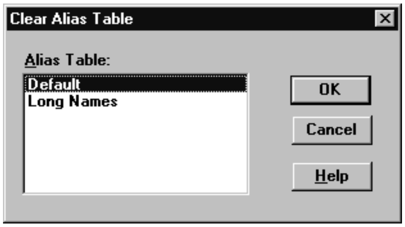
2. Select the alias table to delete from the Alias Table list box.
3. Click OK.

Clearing Alias Tables

➤ To clear the contents of an alias table, that is, remove all of the entries in the table without deleting the table:

1. Select Outline > Aliases > Clear Table to open the Clear Alias Table dialog box.

   Figure 110: Clear Alias Table Dialog Box

2. Select the alias table to clear from the Alias Table list box.
3. Click OK.

Importing and Exporting Alias Tables

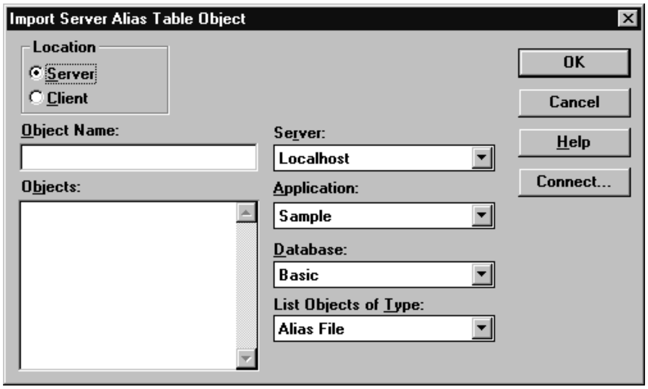
You can import or export alias tables into an outline. For more information on alias tables, see "Introducing Aliases and Alias Tables" on page 261.

Importing an Alias Table

➤ To import an alias table:

1. Select File > Import > Alias Table to open the Import Server Alias Table Object dialog box.

Figure 111: Import Alias Table Object Dialog Box



Import Server Alias Table Object

Location: ● Server ○ Client

Object Name:

Objects:

Server: Localhost

Application: Sample

Database: Basic

List Objects of Type: Alias File

2. Select the name of the alias table import file. Application Manager expects these names to end in `.alt`.
3. Specify the location of the alias table import file by clicking either the Server or Client button.

   If you select Server, the import file must reside in the database directory under `\ESSBASE\APP\`*`application_name`\`database_name`*, where *application_name* and *database_name* represent the name of your application and database. Type the name of the alias table import file in the Object Name text box or select it from the Objects list box.

If you select Client, the alias table can reside in either the application or database directory under `\ESSBASE\CLIENT` or on the drives accessible from the client file system. Click the File System button to select a file from a standard Open Client Data Files dialog box.

Select the alias table import file to load.

Note: The `\ESSBASE\APP` and `\ESSBASE\CLIENT` are the default directories specified during installation. You may have set these directories differently.

4. Click OK.

You can also use the LOADALIAS command in ESSCMD to perform this task. To unload an alias table, use the UNLOADALIAS command in ESSCMD. See the *Technical Reference* in the `docs` directory for information about these commands. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Exporting an Alias Table

To export an alias table:

1. Select File > Export Alias Table to open the Export Server Alias Table Object dialog box.
2. Save the alias table to the desired location.

Tip: Use the LISTLINKEDOBJECTS and PURGELINKEDOBJECTS commands in ESSCMD to perform these tasks. See the *Technical Reference* in the `docs` directory for information about these commands. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Chapter 11

Linking Objects to Essbase Data

This chapter described how you can link various kinds of data with any cell in an Essbase database, using a linked reporting object (LRO). This ability is similar to the file attachment features in an e-mail software package.

An LRO provides improved support for planning and reporting applications and can enhance your data analysis capabilities.

This chapter contains the following sections:

- "Understanding LROs" on page 273
- "Understanding LRO Types and Data Cells" on page 274
- "Managing LROs" on page 275

Note: To limit the size of an LRO to improve performance, see "Limiting LRO File Sizes for Storage Conservation" on page 1452.

Understanding LROs

LROs are objects that you associate with specific data cells in an Essbase database:

Table 18: Types of Linked Objects

Object Type	Description
Cell note	A text annotation of up to 599 characters.
File	An external file, such as a Microsoft Word document, an Excel spreadsheet, a scanned image, an audio clip, or an HTML file (for example, `mypage.htm`).

Table 18: Types of Linked Objects (Continued)

Object Type	Description

URL	An acronym for Uniform Resource Locator. A string that identifies the location of a resource on the World Wide Web, such as a document, image, downloadable file, service, electronic mailbox, or other resource.
	For example:
	`http://www.hyperion.com`
	`ftp://ftp.hyperion.com`
	`file:///D|/ESSBASE/docs/index.htm`

Linked partition	A set of data cells that you can link to in another Essbase database. For more information on linked partitions, see Chapter 13, "Designing Partitioned Applications."

For example, a sales manager may attach cell notes to recently updated budget items. A finance manager might link a spreadsheet containing supporting data for this quarter's results. A product manager might link bitmap images of new products. A sales manager may link the URL of a company's Web site to quickly access the information on the Web site.

Understanding LRO Types and Data Cells

LROs are linked to data cells—not to the data contained in the cells. The link is based on a specific member combination in the database. Adding or removing links to a cell does not affect the cell contents.

When a user links an object to a cell, Essbase creates an entry for the object in the linked object catalog for the database. The catalog, an internal data structure stored with the database's index, contains information describing the object, such as the object handle, object type, the name of the last user to modify the object, and the date the object was modified. Developers use the object handle in Essbase API functions to refer to the object.

The storage strategy depends on the LRO type:

- If the object is a cell note, the text is stored as part of the object description in the catalog entry.
- If the object is a file, the Essbase Kernel stores the contents of the file in the database's directory on the server, giving it a `.LRO` extension. Essbase imposes no restrictions on the data formats of linked files and performs no file-type checking. It is up to the user's client machine to render the file after retrieving it from the OLAP Server.
- If the object is a URL, Essbase stores the URL string as part of the object description in the catalog entry. Essbase does not check the syntax of the URL until the user tries to view it. At that time, Essbase does a preliminary syntax check; then the default Web browser checks for the existence of the URL.
- If the object is a linked partition, it is available through the Essbase Partitioning feature. For more information on linked partitions, see Chapter 13, "Designing Partitioned Applications."

Managing LROs

Users create linked objects through the Spreadsheet Add-in interface by selecting a data cell and choosing a menu item. There is no limit to the number of objects you can link to a cell. The objects are stored on the OLAP Server where they are available to any user with the appropriate access privileges. Users retrieve and edit the objects through the Linked Objects Browser, which displays all objects linked to the selected cell.

The next section describes in more detail how Essbase manages LROs. For more information on how end users work with these objects, see the *Essbase Spreadsheet Add-in User's Guide*.

Before you perform any tasks related to LROs, be aware of these facts:

- Essbase uses a database's index to locate and retrieve linked objects. If you remove data values from a database by choosing Database > Clear Data > All in the Application Manager, the index is deleted and so are the links to linked objects. If you restructure a database, the index is preserved and so are the links to linked objects.
- Shared members share data values but do not share LROs. This is because LROs are linked to specific member combinations and shared members do not have identical member combinations. If you want a given object to be linked to shared members, you must link it to each shared member individually.

This section discusses storing and maintaining LROs on the server:

- "Setting Up Permissions for LROs" on page 276
- "Viewing and Deleting LROs" on page 277
- "Exporting and Importing LROs" on page 280

Setting Up Permissions for LROs

Users who add, edit, and delete LROs through client interfaces need to have the appropriate security permissions in the active database. If the object is a linked partition, the user must also have the required privileges in the database containing the linked partition.

This table lists the permissions required for several different tasks.

Table 19: Permissions Required for LRO Tasks

Task	Permission
Add a linked object to a database	Read/Write
View an existing linked object	Read
Edit an existing linked object	Read/Write
Delete a linked object	Read/Write
Export the LRO catalog to a file	Read
Import the LROs from the LRO-catalog file	Read/Write

Sometimes you might want to prevent users from linking files to data cells without changing their access to other data in a database. You can accomplish this by setting the maximum file size for linked files to 1. Users can then create cell notes, link to a URL, or view linked partitions but can only attach very small files (under 1 kilobyte).

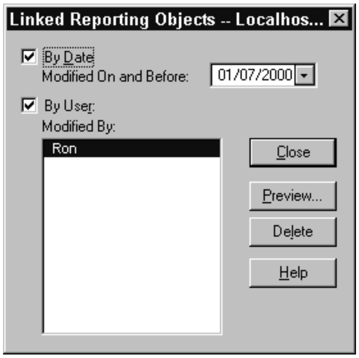
Viewing and Deleting LROs

Users work with LROs through the Spreadsheet Add-in on a cell-by-cell basis. That is, they select a cell and open the Linked Object Browser, which displays the objects linked to the selected cell. Through Application Manager, you can view and delete all LROs for the entire database. You can also view LROs based on selection criteria such as user name and last modification date. For example, you might want to purge all objects that are older than a certain date, or remove the objects belonging to a user who has left the company.

➤ To view or delete the linked objects for a database, follow these steps:

1. From the Application Desktop window, select the application and database name.
2. Select Database > Linked Reporting Objects to display this dialog box:

Figure 112: LROs Dialog Box



Linked Reporting Objects -- Localhos...

☑ By Date
Modified On and Before: 01/07/2000

☑ By User:
Modified By:
Ron

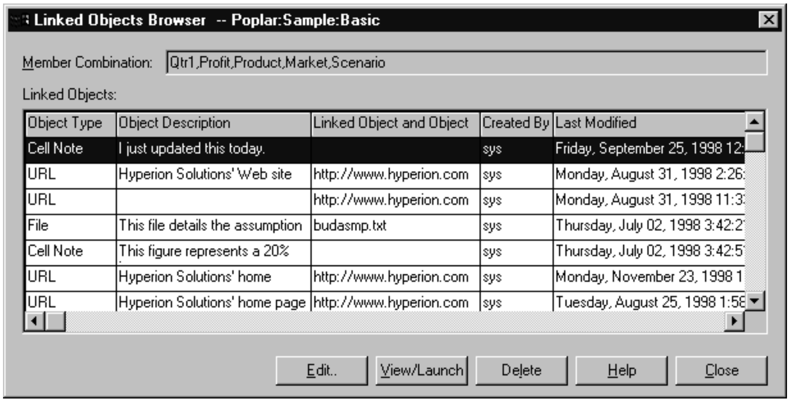
3. Specify your selection criteria as follows:

   - To select by modification date, select By Date and enter a modification date. Objects modified on or before the date are selected.
   - To select by user name, select By User and select a user name. Objects last modified by the user are selected.

     If you select *both* options, objects matching both criteria are selected. If you select neither By Date nor By User, all objects in the database are selected.

4. To delete all objects that meet your criteria, click Delete. To see a list of the objects that meet the criteria, click Preview. If you click Preview, Essbase displays the Linked Objects Browser:

Figure 113: Linked Objects Browser



Member Combination: Qtr1,Profit,Product,Market,Scenario

Linked Objects:

Object Type	Object Description	Linked Object and Object	Created By	Last Modified
Cell Note	I just updated this today.		sys	Friday, September 25, 1998 12:
URL	Hyperion Solutions' Web site	http://www.hyperion.com	sys	Monday, August 31, 1998 2:26:
URL		http://www.hyperion.com	sys	Monday, August 31, 1998 11:3
File	This file details the assumption	budasmp.txt	sys	Thursday, July 02, 1998 3:42:2
Cell Note	This figure represents a 20%		sys	Thursday, July 02, 1998 3:42:5
URL	Hyperion Solutions' home	http://www.hyperion.com	sys	Monday, November 23, 1998 1
URL	Hyperion Solutions' home page	http://www.hyperion.com	sys	Tuesday, August 25, 1998 1:5

In the Linked Objects Browser, the Linked Objects list box displays all the objects that match the criteria you entered. To view, edit, or delete an object, first select it and then click the appropriate button. After you select an object, the member combination it is linked to displays for your reference. The actions you can perform vary depending on the type of object you select.

If you select a linked file, you can:

- Click View/Launch to view the contents of the file. Essbase retrieves the file from the server, copies it to your local client operating system's temporary directory, and saves it under a temporary file name. It then sends a command to the client operating system to open the temporary file. If you make changes to the file and wish to save your changes on the OLAP Server, first save the file on your client machine (making note of the temporary file name), then return to the Linked Objects Browser and click Edit to re-attach the new file.
- Click Edit to re-attach the linked file. You can use this to update the contents of the linked file with a newer version, or to replace the file with a different one.
- Click Delete to delete the link. This destroys the link in the index file and removes the file from the server. It does not affect copies of the file stored locally on client machines.

If you select a cell note, you can:

- Click View/Launch to read the cell note.
- Click Edit to edit its contents.
- Click Delete to remove it from the database.

If you select a URL, you can:

- Click View/Launch to open your default Web browser to view the URL.
- Click Edit to edit the URL string.
- Click Delete to delete the link.

You cannot make changes to linked partitions from the Linked Objects Browser. To create or change a linked partition, open the Partition Manager by choosing Database > Partition Manager. For more information on linked partitions, see Chapter 13, "Designing Partitioned Applications."

You cannot change the member combination associated with any linked object with the Application Manager. To move an object to another member combination, first delete it, then use the Spreadsheet Add-in to re-link the object to the desired member combination.

You can also perform using the ESSCMD commands LISTLINKEDOBJECTS PURGELINKEDOBJECTS and instead of Application Manager. See the *Technical Reference* in the `docs` directory for more information. See also Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Exporting and Importing LROs

To improve backup and data-migration capabilities, you can export and re-import LROs from data intersections in a database.

In preparation for backing up databases, clearing and re-loading data, or migrating data from one database to another, you can export the LROs from a database using the MaxL language interface. For information about exporting and importing LROs, see the MaxL section of the *Technical Reference* in the `docs` directory.

Chapter 12

Designing and Building Currency Conversion Applications

Organizations with offices in different countries generally do business in the currency of the host country (known as the *local currency*). World and regional headquarters of such organizations must convert data entered in local currencies to a *common currency* for consolidation and analysis. The currency conversion option is designed to meet the needs of this complex business problem. This option can be licensed as an "add-on" to OLAP Server.

The currency conversion option includes a sample currency application that contains two databases: INTERNTL and XCHGRATE. This chapter provides background information, step-by-step instructions for building the sample currency application, and procedures for calculating and reporting currency conversions after you have built your own currency application.

This chapter contains the following topics:

- "Overview of the Business Problem" on page 282
- "Overview of the Currency Application Structure" on page 282
- "Steps for Creating a Currency Conversion Application" on page 286
- "Creating the Main Database Outline" on page 287
- "Creating the Currency Database Outline" on page 296
- "Linking the Main and Currency Databases" on page 301
- "Calculating Currency Conversions" on page 303

Note: Your Essbase installation includes the option to install the Sample databases, INTERNTL and XCHGRATE. If you don't have access to these databases, contact your Essbase administrator. For information about installing sample applications, see the *Essbase Installation Guide*.

Overview of the Business Problem

This chapter focuses on solving the business problems that arise as the sample company, The Beverage Company (TBC), expands its business outside the United States. This chapter builds on the business scenario introduced in Chapter 5, "Case Study: Designing a Single-Server Application," as TBC adds the following markets:

- Three locations in Canada: Toronto, Vancouver, and Montreal
- Four locations in Europe: the UK, Germany, France, and Spain

In addition, TBC adds a new member, US, which is a consolidation of data from the United States regions: East, West, South, and Central.

Data for each TBC market location is captured in local currency. Dollar values are derived by applying exchange rates to local values. These values must then be converted to a common currency (in this case, US$).

TBC needs to analyze actual data in two ways:

- Actuals are converted at actual exchange rates.
- Actuals are converted at budget exchange rates to analyze variances due to exchange rates.

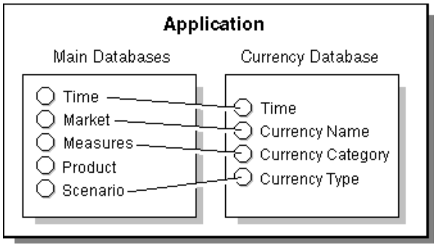
When all actuals have been processed, budget data is converted with budget exchange rates.

Overview of the Currency Application Structure

Currency conversion applications are defined by linking a currency database to a main application database, as illustrated in Figure 114. The *main database* contains the full database outline and associated data values; the *currency database* contains only exchange rates and other currency-related information.

In the example provided in this chapter, TBC's main database is INTERNTL and the currency database is XCHGRATE. On your server, these databases are in the Sample application.

Figure 114: Currency Application Databases



Contents of the Main Database

The main database can be from 3 to *n* dimensions in size. At a minimum, the database must contain the following dimensions:

- A dimension tagged as Time—In the TBC example this dimension is Year.
- A dimension tagged as Accounts—Within an Accounts dimension, individual members may be tagged with different categories of exchange rates. This allows you to apply different exchange rates to various accounts or measures.

  For example, P&L accounts may use exchange rates that differ from those used with balance sheet accounts. In addition, some accounts may not require conversion. For example, members such as Units, Headcount, and Margin% require no conversion.
- A market-related dimension tagged as Country—In a market dimension, each member has an associated currency name. In the TBC example, the currency name US$ is defined for the Market dimension tagged Country, because all local currencies for the new markets must be converted to US$, the company's common currency.

  Because many members can have the same currency name, the number of currency names is typically less than the total number of members in the dimension. As shown in Table 20, TBC uses only six currency names for its 15 Market dimension members. The children of the member Europe all use a different currency and, therefore, must be assigned individual currency names. However, in the case of the Country dimension and the US member and its four cities, all locations use the same currency. The same is true of the Canada member and its three cities. When the children of a given member share a single currency, you only need to define a currency name for the parent member.

Table 20: TBC Currency Names

Dimensions and Members	Currency Name
Market - Country US East West South Central	US$
Canada Toronto Vancouver Montreal	CN$
Europe	
UK	Pound
Germany	Mark
France	Franc
Spain	Peseta

A typical main database also contains an *optional* fourth dimension: “A dimension defined with a currency partition.” Databases are usually partitioned in a Scenario dimension. Creating a currency partition allows Essbase to internally track currency relationships and previously-converted values of the main and exchange rates databases.

Note: A currency conversion partition applies only to the currency conversion option. It is not related to the Partitioning option that enables data to be shared between databases using a replicated, linked, or transparent partition.

The *Essbase Spreadsheet Add-in User's Guide* provides examples of ad hoc currency reporting capabilities. Report scripts also let you define reports that convert data when the report is displayed, as discussed under "Converting Data to a Different Currency in Reports" on page 306.

Contents of the Currency Database

The currency database requires the following three dimensions:

- A Currency Time dimension—Exchange rates are usually defined for each time period. Each member of the time dimension in the main database must be defined in the currency database. Values by time period in the main database are usually converted to the exchange rates of their respective time period from the currency database (although you can convert data values against any period's exchange rate).
- Currency Name dimension—This dimension contains the names of currencies relevant to the markets (or countries) defined in the main database. Each currency name defined in the main database must also exist in the currency database. The currency names define the country-to-exchange rate mapping when conversion occurs.
- Currency Category dimension—This dimension contains all the account categories of the main database. The category names define the exchange rate mapping when a conversion occurs. In the TBC example, the account categories included are P&L (Profit & Loss) and B/S (Balance Sheet).

A currency database typically includes an *optional* fourth dimension: "A Currency Type dimension" which contains members that identify various currency scenario types**.** Typically an application has different exchange rates for scenarios, such as actual, budget, and forecast. Members of the currency-type dimension are not directly mapped to members of the main database. Therefore, member names in this dimension are not required to match member names of the main database.

Conversion Methodologies

Different currency applications have different conversion requirements. Essbase supports two currency database types, each with a different conversion method:

- **A database containing only converted data values.** Some applications require only converted values to be stored in the main database. Local values are entered and the conversion operation overwrites the local values with a common currency. This method assumes that there is no requirement for reporting or analyzing local currencies.

  Because this operation overwrites data, you must load local values and recalculate the data each time you perform a conversion. This method is useful only when you want to perform a single (not ongoing) conversion.

- **A database containing both local and converted data.** Most applications require data to be stored in both local and master values. This method permits reporting and analyzing local data. In addition, data modifications and recalculations are easier to control. To use this method, you must define a currency partition (see "Modify the Scenario Dimension" on page 293). The Beverage Company (TBC) uses this method for defining their currency application.

Either of these two methods may also require a currency conversion to be applied at report time. This allows you to analyze various exchange rate scenarios without actually storing data in the database. The currency conversion module allows you to perform ad hoc conversions with the Spreadsheet Add-in, which is discussed in the *Essbase Spreadsheet Add-in User's Guide*, or with a report script, as discussed under "Converting Data to a Different Currency in Reports" on page 306.

Steps for Creating a Currency Conversion Application

To accommodate the new markets TBC has added and to provide Essbase with required currency-related information, TBC has modified their existing Essbase database outline and created a currency conversion application. You can use the TBC model to create your own currency conversion application, revising the steps as needed to fit your specific requirements.

The TBC currency conversion application was created using the following steps:

1. "Creating the Main Database Outline" on page 287
    - "Open the Existing Database Outline" on page 287
    - "Modify the Measures Dimension" on page 288
    - "Modify the Market Dimension" on page 290
    - "Modify the Scenario Dimension" on page 293
    - "Save the Main Database Outline Changes" on page 295
2. "Creating the Currency Database Outline" on page 296
    - "Generate the Currency Database Outline" on page 296
    - "Review the Contents of the Currency Database Outline" on page 299
    - "Add New Members to the CurType Dimension" on page 299
    - "Save the Currency Database Outline Changes" on page 300
3. "Linking the Main and Currency Databases" on page 301

Creating the Main Database Outline

To create the main database outline (INTERNTL), you need to open your existing Essbase database outline, make changes to its contents, and then save the outline for use in your currency conversion application.

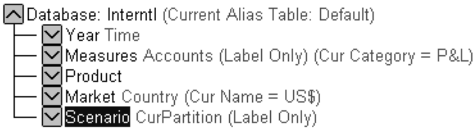
Open the Existing Database Outline

Begin creating the main database outline by opening TBC's existing outline, as follows:

1. Start Application Manager and select Server > Connect.
2. Choose the OLAP Server that contains the Sample application.

3. Select Sample from the Applications list box and open the database outline for the database INTERNTL, as shown in Figure 115.

Figure 115: Sample Database INTERNTL Essbase Outline



```
Database: Interntl (Current Alias Table: Default)
├── Year Time
├── Measures Accounts (Label Only) (Cur Category = P&L)
├── Product
├── Market Country (Cur Name = US$)
└── Scenario CurPartition (Label Only)
```

TBC has modified the Measures, Market, and Scenario dimensions. The Year and Product dimensions require no changes because they have no information specifically related to currencies.

Note: The Year dimension *must* be tagged as the Time dimension.

Modify the Measures Dimension

To create their currency conversion main database, TBC modified the Measures dimension using the following steps. You can follow along and revise the procedures as necessary to create your own currency application.

Note: Each descendant of a member inherits the currency category tag of its ancestor. A member or sub-branch of members can also have its own category defined.

1. Tag Measures as an Accounts dimension.

   You must have a dimension tagged as Accounts in your main database. To meet varying conversion and exchange rate requirements, individual members within the Accounts dimension can then be tagged with different categories of exchange rates.

   All descendants of the Profit member, for example, use a special currency rate that applies to Profit and Loss accounts. Therefore, they are defined with a currency category of P&L.

2. Define the Profit member with a currency category name:

   a. Select the Profit member and click the Data Dictionary button, 📖, on the toolbar. The Member Specification dialog box is displayed.

   b. Select the Category option button from the currency conversion group, and enter the category name of P&L (for Profit and Loss).

3. Define the Measures dimension with a currency category name:

   a. Select Prev to move to the previous member in the dimension and select Category from the currency conversion group. The name P&L is displayed from the previous entry. Because the Measures dimension shows data for Profit, it must also have the P&L currency category name applied.

   b. Click OK to close the dialog box.

4. Define the Inventory member with a currency category name, as follows:

   a. Select the Inventory member and click the Data Dictionary button, , on the toolbar. The Member Specification dialog box is displayed.

   b. Select Category from the currency conversion group, and enter the category name of B/S, because all descendants of the Inventory member use a special rate that applies to balance sheet accounts, B/S (for Balance Sheet).

   c. Click OK to close the dialog box.

5. Define the Ratio member as not requiring a conversion:

   a. Select the Ratios member, and press the Data Dictionary button, , on the toolbar. The Member Specification dialog box is displayed.

   b. Select No Conversion from the currency conversion group. Ratio members do not require a currency conversion, because the local and converted values yield the same result.

Note: The children of a member defined with No Conversion do not inherit the No Conversion definition. Each member must be defined individually.

6. Define the children of the Ratio member (Margin % and Profit %) as not requiring a conversion:

   a. Select Next to move down the member list to Margin %.

   b. Define Margin % with the No Conversion option.

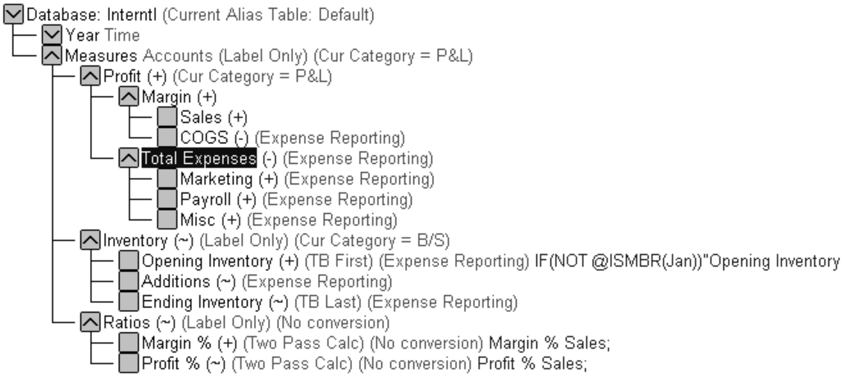
   c. Select Next to move down the member list to Profit %.

   d. Define Profit % with the No Conversion option.

   e. Click OK to close the dialog box.

Figure 116 shows the resulting Measures dimension outline after making the modifications necessary for the TBC currency conversion application.

Figure 116: TBC Measures Main Database Outline



```
Database: Interntl (Current Alias Table: Default)
  Year Time
  Measures Accounts (Label Only) (Cur Category = P&L)
    Profit (+) (Cur Category = P&L)
      Margin (+)
        Sales (+)
        COGS (-) (Expense Reporting)
      Total Expenses (-) (Expense Reporting)
        Marketing (+) (Expense Reporting)
        Payroll (+) (Expense Reporting)
        Misc (+) (Expense Reporting)
    Inventory (~) (Label Only) (Cur Category = B/S)
      Opening Inventory (+) (TB First) (Expense Reporting) IF(NOT @ISMBR(Jan))"Opening Inventory
      Additions (~) (Expense Reporting)
      Ending Inventory (~) (TB Last) (Expense Reporting)
    Ratios (~) (Label Only) (No conversion)
      Margin % (+) (Two Pass Calc) (No conversion) Margin % Sales;
      Profit % (~) (Two Pass Calc) (No conversion) Profit % Sales;
```

Modify the Market Dimension

TBC needed to make the following modifications to the Market dimension to accommodate their newly added markets and create their currency conversion application main database. Procedures for making these changes follow the bullet list.

- Tag the Market dimension as Country. Remember, having a Country tag enables you to define the common currency to be used for converting local currency values. In the TBC example, local currency values for their new markets are converted to US$, the currency name defined for the Country dimension.
- Add the US member, which consolidates data from existing United States region members East, West, South, and Central.
- Add the Canada member and its children, Toronto, Vancouver, and Montreal.
- Add the Europe member and its children, UK, Germany, France, and Spain.
- Define currency names for each of the newly added dimensions/members, as necessary. Remember, when the children of a given member share a single currency, you only need to define a currency name for the parent member.

Modify the Market dimension as follows:

1. Tag Market as a Country dimension by selecting the Market dimension and clicking Country, , on the toolbar.

   The existing members (East, West, South, and Central) belong in TBC's newly added US market. To make this outline change, TBC added the US member with the existing members as descendants.

2. Add the new US member:

   a. Select the Market member and click the Add Child button, .

   b. Type the name US in the Member Edit box and press Enter twice.

3. Make existing members East, West, South, and Central, children of US by clicking the Add Child button for each member.

4. Add Canada and its respective cities to the Market dimension:

   a. Select the US member and click the Add Sibling button, .

   b. Type the name Canada in the edit box and press Enter twice.

   c. Select the Canada member and click the Add Child button.

   d. Add the Toronto, Vancouver, and Montreal members and press Enter twice.

5. Define currency names for each appropriate member. Because TBC required the database to be converted to US dollars, they defined the Market dimension with the US$ currency name:

   a. Select the Canada member and press the Add Sibling button.

   b. Type the name Europe in the edit box and press Enter twice. Select the Europe member and click the Add Child button.

   c. Add the UK, Germany, France, and Spain members and press Enter twice.

6. Define currency names for the Market dimension and its members, as appropriate. All descendants of Canada also use the same currency (CN$):

   a. Select the Market dimension and click the Data Dictionary button, . The Member Specification dialog box displays.

   b. Enter US$ in the Currency Name type in box.

   c. Click OK to close the dialog box.

   All descendants of the Market dimension inherit the US$ currency name (unless a child branch has already been defined with another currency). Because members in the US branch use US$ as their local currency, there is no need to define a currency name for these members.

7. Define a currency name for the Canada member:

   a. Select Canada and click the Data Dictionary button, . The Member Specification dialog box is displayed.

   b. Type CN$ in the Currency Name box.

   c. Click OK to close the dialog box.

   The children of Europe all use different local currencies. Unlike the Market dimension and the Canada member, you must define each Europe member with an individual currency name.

8. Define individual currency names for children of the Europe member:

   a. Select the UK member name and click the Data Dictionary button, . The Member Specification dialog box is displayed.

   b. Type Pound in the Currency Name box.

   c. Click Next to define the currency name for next country (Germany).

   d. Type Mark in the Currency Name box.

   e. Click Next to define the currency name for next country (France)

   f. Type Franc in the Currency Name box.

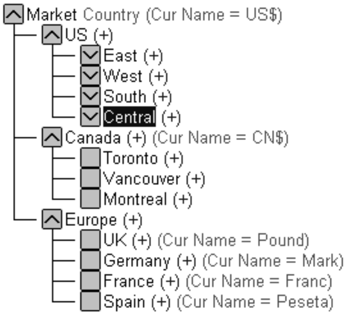
   g. Click Next to define the currency name for next country (Spain).

   h. Type Peseta in the Currency Name box.

   i. Click OK to close the dialog box.

Figure 117 shows the resulting Market dimension outline after making the modifications necessary for the TBC currency conversion application.

Figure 117: TBC Market Dimension Database Outline



```
Market Country (Cur Name = US$)
    US (+)
        East (+)
        West (+)
        South (+)
        Central (+)
    Canada (+) (Cur Name = CN$)
        Toronto (+)
        Vancouver (+)
        Montreal (+)
    Europe (+)
        UK (+) (Cur Name = Pound)
        Germany (+) (Cur Name = Mark)
        France (+) (Cur Name = Franc)
        Spain (+) (Cur Name = Peseta)
```

Modify the Scenario Dimension

In the TBC main database (INTERNTL), the Scenario dimension contains members for both local and converted values. TBC needed to make the following modifications to the Scenario dimension to create their currency conversion application main database:

- Create a currency partition. This enabled TBC to store both converted and non-converted values in their main database (INTERNTL).
- Define the converted members under a Dollars parent. This enabled TBC to track actuals converted at actual exchange rates.
- Add a new member called Actual @ Bud XChg. This enabled TBC to track actuals converted at separate budget exchange rates to analyze variances due to exchange rates of their newly added markets.
- Define members that can store data in local currency values (Actual and Budget).
- Apply the Exclude member from consolidation tag to the Local, Bud member, because it doesn't store data.
- Tag the Local member as a label, because it doesn't store data.
- Tag the Scenario dimension as a label, because it also does not store data.

Modify the Scenario dimension as follows:

1. Set a Currency Partition tag on the Scenario dimension:
   - a. Select the Scenario dimension name and click the CurPartition button, Cur-Part.
   - b. Define the converted members under a parent called Dollars.
   - c. Add the member Dollars as a child of Scenario. Select the dimension Scenario and press the Add Child button.
   - d. Type Dollars in the member edit box and press Enter twice.
   - e. Move the Actual, Budget, Variance, and Variance % members as children of Dollars.
2. The Dollars member does not store data and must, therefore, be tagged as a label by selecting it and clicking the Label button, L.
3. TBC also wants to track actuals using the budget exchange rate. The Scenario dimension requires a new member to handle this data. Add a new member, Actual @ Bud XChg, to the Scenario dimension Actual member:
   - a. Select the Actual member and click the Add Sibling button.
   - b. Enter the member Actual @ Bud XChg and press Enter twice.
4. The newly added Actual @ Bud XChg member does not consolidate into the Dollars member. Apply the “Exclude member from consolidation” tag to the Actual @ Bud XChg member by pressing the No Consolidation button, ∿.
5. Define members that can store the data in local currency values:
   - a. Add the member Local as a sibling of Dollars. Select the member name Dollars and press the Add Child button.
   - b. Enter the name Local in the Member Edit box and press Enter twice.
6. Data only needs to be captured in local currency for Actual and Budget. These are the only children to define for Local. Define Local member children to capture local data:
   - a. Select the Local member and click the Add Child button.
   - b. Enter the member names Act and Bud and press Enter twice.

7. Apply the “Exclude member from consolidation” tag to the Bud member by pressing the No Consolidation button, .

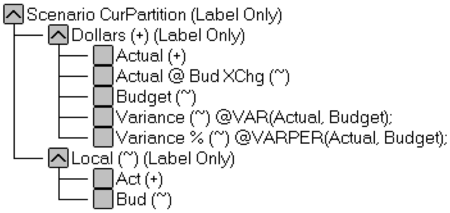
   Values for the Act member consolidate to Local, but values for the Bud member do not.

8. Tag the Local member (which does not store data) as a Label by selecting it and clicking the Label button.

9. Tag the Scenario dimension (which does not contain any consolidated results) as a Label by selecting it and clicking the Label button.

   Figure 118 shows the resulting Scenario dimension outline after making the modifications necessary for the TBC currency conversion application.

Figure 118: Scenario Dimension Database Outline

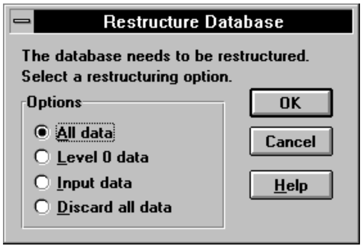


```
Scenario CurPartition (Label Only)
    Dollars (+) (Label Only)
        Actual (+)
        Actual @ Bud XChg (~)
        Budget (~)
        Variance (~) @VAR(Actual, Budget);
        Variance % (~) @VARPER(Actual, Budget);
    Local (~) (Label Only)
        Act (+)
        Bud (~)
```

Save the Main Database Outline Changes

Save the outline changes by clicking the Save button. If your database contains data values, Essbase restructures the database to reflect changes to the outline. When you click Save, the Restructure Database dialog box shown in Figure 119 is displayed.

Figure 119: Restructure Database Dialog Box



Restructure Database

The database needs to be restructured.
Select a restructuring option.

Options

● All data
○ Level 0 data
○ Input data
○ Discard all data

When an existing outline is updated, you have several choices that let you restructure data values. Because no data has been loaded into the model, click OK to continue the outline update.

Creating the Currency Database Outline

Once you have verified and saved the main database outline, you can generate the currency outline. The currency outline contains dimensions, members, currency names, and currency categories previously defined in your main database outline. It is basically structured and ready to use after being generated but may require additions to make it complete.

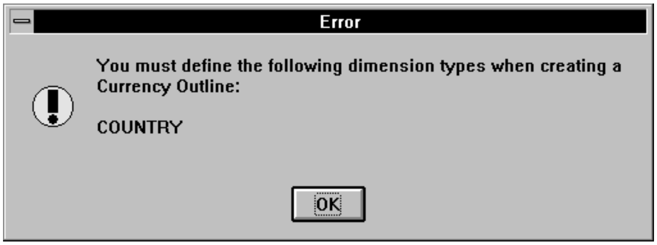
Generate the Currency Database Outline

Follow these steps to generate the currency outline:

1. Press the Generate Currency Outline button, $.

   If the main database outline is missing any required dimension tags (such as Time, Accounts, or Country), Essbase displays an error box like the one shown in Figure 120.

   Figure 120: Error Dialog Box Displayed When No Country Dimension Is Defined

   You must define the following dimension types when creating a Currency Outline:

   COUNTRY

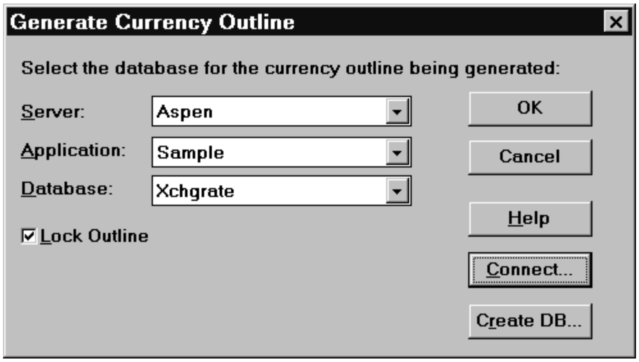
2. If the Error dialog box is displayed after you attempt to generate the currency outline, do the following:

   a. Note the errors and click OK.

   b. Tag the dimensions as required.

c. Attempt to generate the currency outline again. If all tags are correctly defined, the screen displays the dialog box shown in Figure 121.

Figure 121: Generate Currency Outline Dialog Box



Generate Currency Outline

Select the database for the currency outline being generated:

Server: Aspen

Application: Sample

Database: Xchgrate

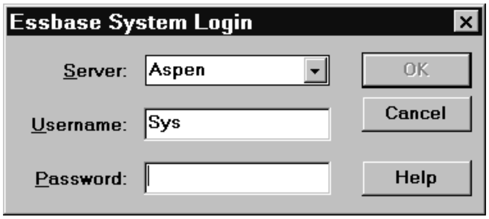
☑ Lock Outline

The Generate Currency Outline dialog box lets you define the database for which the outline is generated. The dialog box contains the following controls:

- The Server list box, which lets you select a OLAP Server
- The Application list box, which lets you select an application
- The Database list box, which lets you select a database (only currency databases is displayed in the list)
- The Connect button, which lets you log in to a different server if necessary
- The Create DB button, which lets you create a new currency database

If you need to log in to a different server, click the Connect button. The dialog box shown in Figure 122 is displayed. If you do not need to log in to a different server, go to Step 5.

Figure 122: Essbase System Login Dialog Box



Essbase System Login

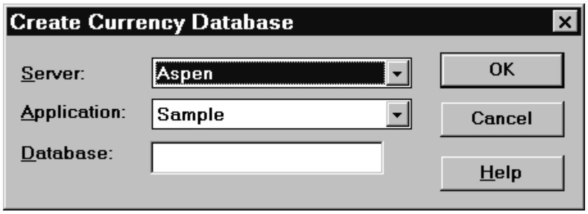
Server: Aspen

Username: Sys

Password:

3. Select the correct server, enter your user name and password, and click OK to close the dialog box and return to the Generate Currency Outline dialog box.
4. Click the Create DB button. The dialog box shown in Figure 123 is displayed.

Figure 123: Create Currency Database Dialog Box



Server: Aspen

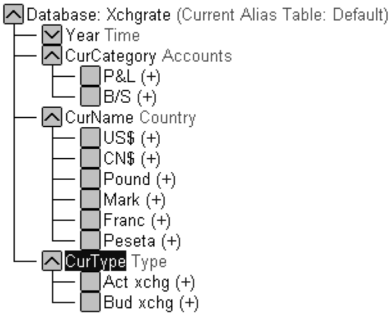
Application: Sample

Database:

5. Type the database name in the Database text box. In the TBC example, we use the database name XCHGRATE.
6. Click OK to create the database.
7. When the Generate Currency Outline dialog box is displayed again, click OK to generate the outline.

   The outline for the currency database XCHGRATE is displayed in the Outline Editor as shown in Figure 124.

Figure 124: Currency Database Outline



```
Database: Xchgrate (Current Alias Table: Default)
    Year Time
    CurCategory Accounts
        P&L (+)
        B/S (+)
    CurName Country
        US$ (+)
        CN$ (+)
        Pound (+)
        Mark (+)
        Franc (+)
        Peseta (+)
    CurType Type
        Act xchg (+)
        Bud xchg (+)
```

Review the Contents of the Currency Database Outline

After generating your currency database outline, review its contents and note the following items that are automatically created based on the contents of the main database that you defined previously:

- The Year dimension is replicated from the main database. This allows you to define exchange rates for each time period.
- The currency category names that were defined on some members of the Measures dimension in the main database become members of the CurCategory dimension.
- The currency names from the Market dimension become members of the CurName dimension.
- The CurType dimension, which contains no members, is also created. This dimension is not directly mapped to the main database.

Add New Members to the CurType Dimension

After you have generated the currency database, you can add members to any dimension. Because the TBC currency database (XCHGRATE) contained different exchange rates for actual and budget, they added two new members to the CurType dimension to apply to different scenarios in the main database:

1. Add the Act xchg member to the CurType dimension.
2. Add the Bud xchg member to the CurType dimension.

Exchange rates from these two members are applied to different scenarios from the main database, as follows:

- Actual (from the main database) is converted with a CurType member called Act xchg.
- Actual @ Bud Xchg (from the main database) is converted with a CurType member called Bud xchg.
- Budget (from the main database) is converted with a CurType member called Bud xchg.

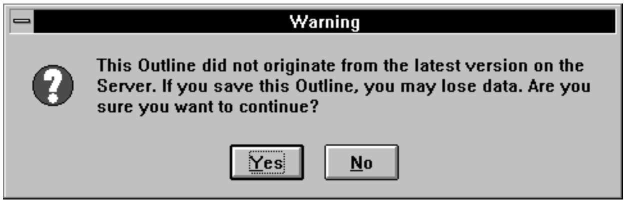
Note: For details about the currency conversion calculation process, see "Calculating Currency Conversions" on page 303.

Save the Currency Database Outline Changes

When you have reviewed the newly generated currency database outline and have made any necessary additions, you need to save your changes.

1. Save changes made to the currency database outline by pressing the Save button. The dialog box shown in Figure 125 is displayed.

Figure 125: Using the Latest Version on the Server Warning



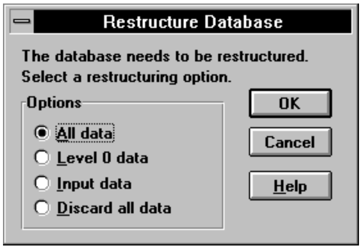
Warning

This Outline did not originate from the latest version on the Server. If you save this Outline, you may lose data. Are you sure you want to continue?

Because the newly-generated outline contains all new members, you are given a choice of continuing or canceling the operation.

2. Click Yes to continue. The Restructure Database dialog box shown in Figure 126 is displayed.

Figure 126: Restructure Database Dialog Box



Restructure Database

The database needs to be restructured.
Select a restructuring option.

Options

● All data
○ Level 0 data
○ Input data
○ Discard all data

When an existing outline is updated, you have several choices that let you restructure data values.

3. Because no data has been loaded into the model, click OK to continue the outline update.
4. Close the Outline Editor for the main and currency databases.

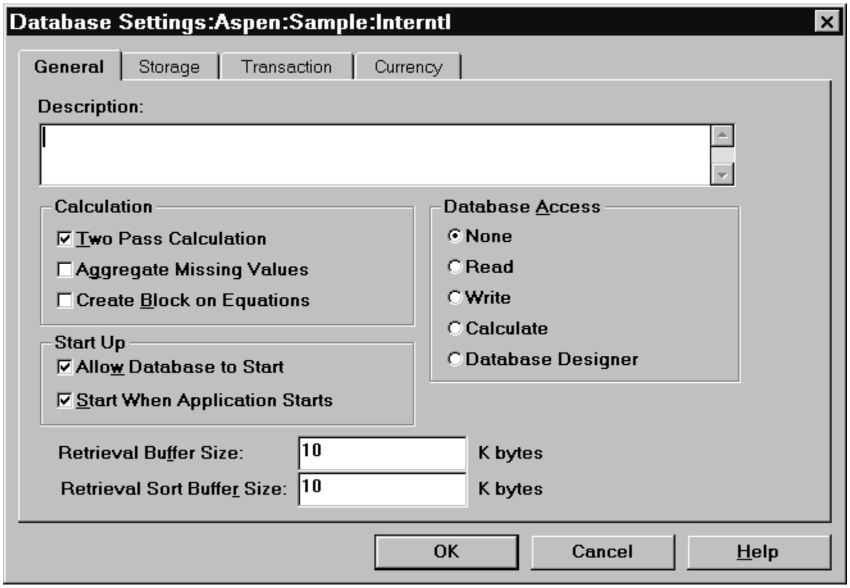
Linking the Main and Currency Databases

To perform a currency conversion calculation, Essbase must recognize a link between the main and currency databases. Generating a currency outline does not automatically link a main database with a currency database.

➤ To link a currency database to the main database in Application Manager:

1. Select the main database from the application's database list.
2. Choose Database > Settings. The General page of the Database Settings dialog box shown in Figure 127 is displayed.

Figure 127: Database Settings Dialog Box General Page



Database Settings:Aspen:Sample:Interntl

Description:

Calculation
☑ Two Pass Calculation
☐ Aggregate Missing Values
☐ Create Block on Equations

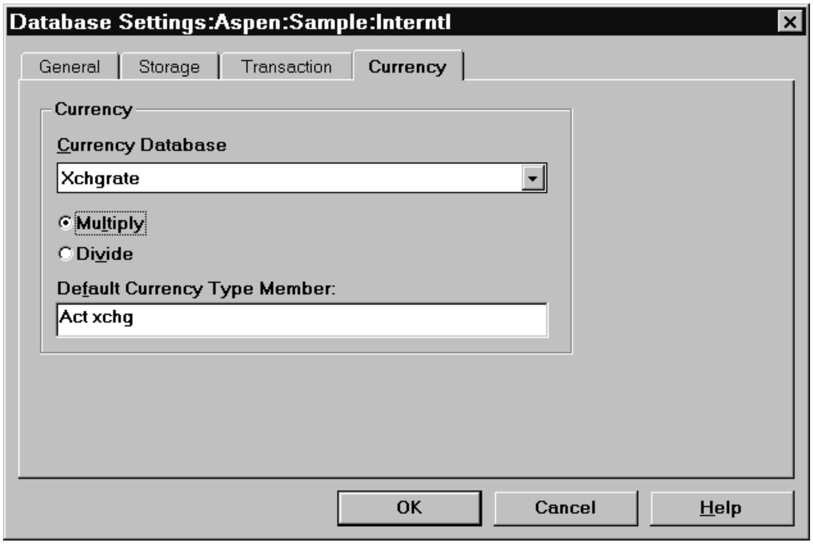
Start Up
☑ Allow Database to Start
☑ Start When Application Starts

Database Access
● None
○ Read
○ Write
○ Calculate
○ Database Designer

Retrieval Buffer Size: 10 K bytes
Retrieval Sort Buffer Size: 10 K bytes

3. Select the Currency page. The Currency page shown in Figure 128 is displayed.

   Figure 128: Database Settings Dialog Box Currency Page

   Database Settings:Aspen:Sample:Interntl

   Currency Database: Xchgrate

   ● Multiply
   ○ Divide

   Default Currency Type Member: Act xchg

4. Select a database from the currency database list box.

   The Multiply and Divide options are enabled. Use these option buttons to define the conversion calculation method to be used. The Multiply option button multiplies each local data value by the exchange rate. The Divide option button divides each local data value by the exchange rate.

5. Select the Multiply option.

6. The Default Currency Type Member text box lets you define a currency type member to use as a default for currency conversion calculations. Type **Act xchg** to assign this as the default.

7. Click OK twice to close the dialog box and save these settings.

After the databases are defined and linked, the next step is to load data into the main database and exchange rates into the currency database. After values have been entered, a currency conversion is calculated. This is accomplished by running a calculation script (see "Calculating Currency Conversions" on page 303) or by using the Database Properties window in Administration Services.

Calculating Currency Conversions

You convert data values from a local currency to a common, converted currency using the CCONV *currExchMbr* command in a calculation script. For example, you might convert data from a variety of European currencies into US$.

Note: You cannot use the CCONV command to convert data in a transparent partition.

You convert the data values back to the original, local currencies using the CCONV TOLOCALRATE *CurType* command.

You can convert all or part of your main database using the rates defined in your currency database. You can overwrite the local values with the converted values, or you can keep both the local and converted values in your main database, depending on your tracking and reporting needs.

Overwriting Local Values with Converted Values

If you want to overwrite the local values, you do *not* need to create a CURPARTITION dimension in your main database. Use the CCONV command in a calculation script to convert all the data in your database.

The following calculation script converts the values in the database to US$.

```
CCONV US$;
CALC ALL;
```

If required, you can specify a currency name that contains the required exchange rate. The following calculation script converts the values in the database to US$, using the exchange rate for Jan as defined in the currency database.

```
CCONV Jan->US$;
CALC ALL;
```

The CALC ALL command is required in the examples shown, because the CCONV command only converts currencies. It does not consolidate or calculate members in your database.

The following calculation script converts the values back to their original values in their local currencies using the `"Act xchg"` rate:

```
CCONV TOLOCALRATE "Act xchg";
CALC ALL;
```

For more information on the CCONV command, see the *Technical Reference* in the `docs` directory.

Note: You cannot use the FIX command unless you are using a CURPARTITION dimension and the `CCTRACK` setting is `TRUE` in the `ESSBASE.CFG` file.

Keeping Local and Converted Values

You can keep both local and converted values in your database. In your main database you need to define the members that store the local and converted values. You do this by creating a CurPartition dimension (see "Modify the Scenario Dimension" on page 293). The CurPartition dimension has two partitions, one for local values and one for converted values.

➤ To create a calculation script that copies local data to the converted partition and calculates it, use this procedure:

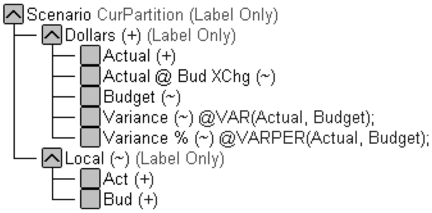
1. Use the DATACOPY command to copy data from the local to the converted partition.
2. Use the FIX command to calculate only the converted partition and use the CCONV command to convert the data.

   Note: When using a CurPartition dimension, you must FIX on a member of this dimension to use the CCONV command.

3. Use the CALC command to recalculate your database.

The following example is based on the Sample INTERNTL database and corresponding Sample XCHGRATE currency database. Figure 129 shows the Currency Partition from the Sample INTERNTL database.

Figure 129: Calculating Local and Converted Currency Conversions



```
Scenario CurPartition (Label Only)
├── Dollars (+) (Label Only)
│   ├── Actual (+)
│   ├── Actual @ Bud XChg (~)
│   ├── Budget (~)
│   ├── Variance (~) @VAR(Actual, Budget);
│   └── Variance % (~) @VARPER(Actual, Budget);
└── Local (~) (Label Only)
    ├── Act (+)
    └── Bud (+)
```

The following calculation script performs three currency conversions for Actual, Budget and Actual @ Bud Xchg data values.

```
/* 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;
```

The following calculation script converts the Actual and Budget values back to their original values in their local currencies:

```
FIX(Actual)
CCONV TOLOCALRATE "Act xchg";
ENDFIX
FIX(Budget)
CCONV TOLOCALRATE "Bud xchg";
ENDFIX
CALC ALL;
```

Note: When you convert currencies using the CCONV command, the resulting data blocks are marked as *dirty* for the purposes of Intelligent Calculation. This means that Essbase recalculates all the converted blocks when you recalculate your database. For more information on Intelligent Calculation, see Chapter 52, “Optimizing with Intelligent Calculation.”

Converting Data to a Different Currency in Reports

You can calculate currency conversions in report scripts, using the CURRENCY *target currency* command to set the output currency and currency type. For the syntax and definitions of Report Writer commands, see the *Technical Reference* in the `docs` directory.

Note: Essbase cannot perform “on the fly” currency conversions across transparent databases. If you have two transparent partition databases that are calculated using different conversions, you cannot calculate currency conversions in reports.

The following Sample report contains first quarter Budget Sales for colas, using the January exchange rate for the Peseta currency.

```
                       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
```

Use the following script to create the Sample currency conversion report:

```
<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
   !
```

Effects of CCTRACK Parameter on Conversion Calculations

The CCTRACK setting in your ESSBASE.CFG file controls whether exchange rates are tracked while Essbase calculates currency conversions.

When CCTRACK is True, Essbase tracks exchange rates that are applied to data as conversions are calculated, allowing conversion to occur at report time through the Spreadsheet Add-in or the Report Writer.

Setting CCTRACK to False turns off the tracking system and has the following results:

- A CCONV command assumes the data to be converted is always in local currency.
- Currency report options assume that the data to be converted is always in local currency. This means that even if the data has already been converted in the database, it is reconverted at report time and can yield inaccurate results.
- The restrictions on operations such as FIX or DATACOPY are removed for conversion operations. For example, when you have a currency partition and CCTRACK is set to True, you can’t use the CCONV command outside of a FIX. Conversely, when you do not have a currency partition and CCTRACK is set to TRUE, Essbase does not allow you to calculate without a FIX. Setting CCTRACK to False eliminates these restrictions.

For more information, refer to *Technical Reference* in the docs directory.

Part

III

Designing, Building, and Maintaining Partitions

This part describes how to design, build, and maintain partitions:

- Chapter 13, “Designing Partitioned Applications” provides the information you need to decide when, or if, to use partitions, how to determine which data to partition, choosing the correct type of partition, security issues related to partitions, and scenarios of partition use.
- Chapter 14, “Building and Maintaining Partitions” describes how to create and manage application partitions, including how to synchronize the outlines of two or more partitioned databases and how to update data in a replicated partition.

Chapter 13 Designing Partitioned Applications

An Essbase partitioned application can span multiple servers, processors, or computers. Partitioning your applications can help you:

- Improve the scalability, reliability, availability, and performance of your databases
- Reduce the size of your databases
- Use your resources more efficiently

This chapter contains the following sections:

CAUTION: You must design partitions carefully. We strongly recommend that you read this chapter before creating partitions.

Introducing Essbase Partitioning

Essbase Partitioning is a collection of features that makes it easy to design and administer databases that span Hyperion Essbase applications or servers. You must license Partitioning separately from Essbase. You must license the Partitioning option for every server that contains a database partition.

Partitioning can help you:

- Synchronize the data in multiple partitioned databases. Essbase tracks changes made to data values in a partition and provides tools for updating the data values in related partitions.
- Synchronize the outlines of multiple partitioned databases. Essbase tracks changes made to the outlines of partitioned databases and provides tools for updating related outlines.
- Allow users to navigate between databases with differing dimensionality. When users drill across to the new database, they can drill down to more detailed data.

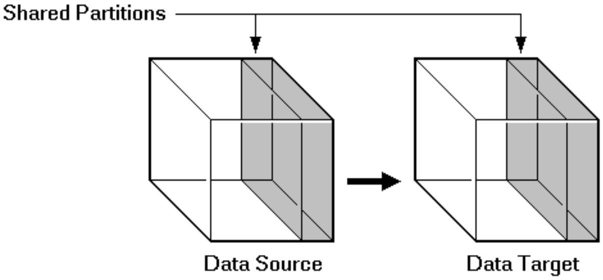
Based on user requirements, you can decide to:

- Partition applications from the top down. *Top-down partitioning* allows you to split a database onto multiple processors, servers, or computers. Top-down partitioning can improve the scalability, reliability, and performance of your databases. To achieve the best results with top-down partitioning, create a separate application for each partitioned database.
- Partition applications from the bottom up. *Bottom-up partitioning* allows you to manage the flow of data between multiple related databases. Bottom-up partitioning can improve the quality and accessibility of the data in your databases.
- Partition database according to attribute values associated with base dimensions (a base dimension is a standard dimension associated with one or more attribute dimensions). Partitioning a base dimension according to its attributes enables the user to extract data based on the characteristics of a dimension such as flavor or size. For more information on attributes, see Chapter 9, “Working with Attributes.”

Data Sources and Data Targets

Partitioned databases contain at least one *data source*, the primary site of the data, and at least one *data target*, the secondary site of the data. A single database can serve as both the data source for one partition and the data target for another. When you define a partition, you map cells in the data source to their counterparts in the data target:

Figure 130: Data Source and Data Target



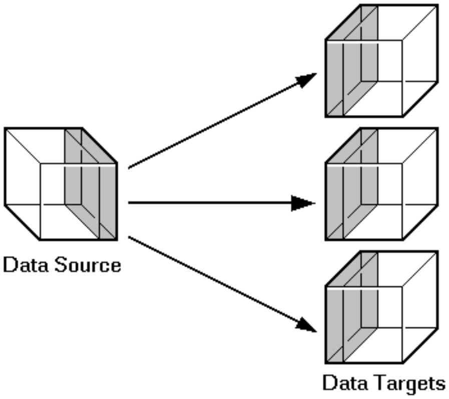
An Essbase database can contain many different partitions as well as data that is not shared with any other Essbase database. You can define partitions between:

- Different databases in different applications, as long as each database uses the same language (for example, German).
- Different databases in different applications on different processors or computers, as long as each database uses the same language (for example, German).
- Different databases in one application. This is not recommended, because you cannot reap the full benefits of partitioning your databases unless each database is in a separate application.

You can only define one partition of each type between the same two databases. For example, you can only create one replicated partition between the Sampeast East database and the Samppart Company database. The East or Company databases can, however, contain many replicated partitions that connect to other databases.

A single database can serve as the data source or data target for multiple partitions. To share data among many databases, create multiple partitions, each with the same data source and a different data target:

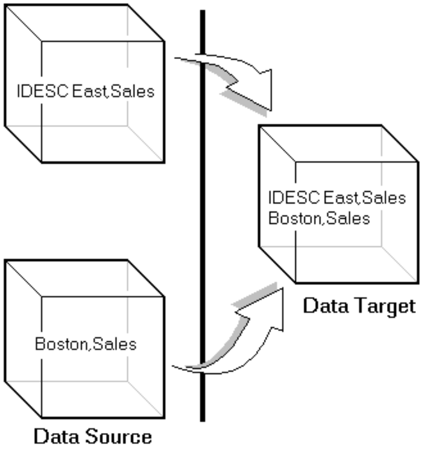
Figure 131: Data Shared at Multiple Targets



Overlapping Partitions

An overlapping partition occurs when similar data from two or more databases serve as the data source for a single data target in a partition. For example, `IDESC East, Sales` from database 1 and `Boston, Sales` from database 2 are mapped to `IDESC East, Sales` and `Boston, Sales` in database 3. Because `Boston` is a member of the dimension `East`, the data for `Boston` mapped to database 3 from database 1 and database 2, overlap. This data overlap results in an overlapping partition:

Figure 132: Overlapping Partitions



An overlapping partition is allowed in linked partitions, but is invalid in replicated and transparent partitions and will generate an error message during validation.

What Is a Partition?

A *partition* is the piece of a database that is shared with another database. Partitions contain the following parts, as illustrated in Figure 133.

- Data source information

  The server, application, and database name of the data source.

- Data target information

  The server, application, and database name of the data target.

- Login and password

  The login and password information for the data source and the data target. This information is used for internal requests between the two databases to execute administrative and user operations.

- Type of partition

  A flag indicating whether the partition is replicated, transparent, or linked.

- Shared areas

  A definition of one or more areas shared between the data source and the data target. An area is a subcube of the database that is shared between databases. To share more than one non-contiguous portion of a database, define multiple areas in a single partition. This information determines which parts of the data source and data target are shared so that Essbase can put the proper data into the data target and keep the outlines for the shared areas synchronized.

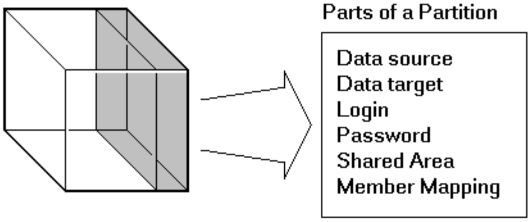
- Member mapping information

  A description of how the members in the data source map to members in the data target. Essbase uses this information to determine how to put data into the data target if the data target and the data source use different names for some members and dimensions.

- State of the partition

  Information about whether the partition is up-to-date and when the partition was last updated.

Figure 133: Parts of a Partition



Workflow for Partitioning a Database

Here is the suggested workflow for partitioning a database.

1. Design the partitioned databases. See "Determining Which Data to Partition" on page 318 and "Deciding Which Type of Partition to Use" on page 319.
2. Edit existing applications that use the individual databases.
   - Edit the outlines and existing calculation scripts.
   - If necessary, edit the data sources, rules files, and report scripts.
   - Determine whether you need to define additional security filters to control what users can view in a partitioned database. See "Setting Up Security for Partitioned Databases" on page 339.

3. Create the partitions. See Chapter 14, “Building and Maintaining Partitions.”
4. Load data into the partitions. Load the data into the new databases. See Chapter 20, “Introducing Data Loading.”
5. If necessary, calculate the new databases. See Chapter 24, “Introduction to Database Calculations.”
6. Test and maintain the partitions. Include tests of updates, data loads, calculation scripts, report scripts, and database restructures. For more information, see Chapter 14, “Building and Maintaining Partitions.”

When to Partition a Database

Review the following list of questions. If you find yourself answering yes to many of them, or answering yes to some that are very important to you, partitioning your databases may solve your problems.

- Should the data be closer to the people who are using it? Is the network being stressed because users are accessing data that is far away?
- Would a single failure be catastrophic? If everyone is using a single database for mission-critical purposes, what happens if the database goes down?
- Does it take too long to perform calculations after new data is loaded? Can you improve performance by spreading the calculations across multiple processors or computers?
- Do users want to see the data in different application contexts? Would you like to control how they navigate between databases?
- Do you have separate, disconnected databases storing related information? Does the related information come from different sources? Are you having trouble synchronizing it?
- Will you be adding many new organizational units? Would they benefit from having their own databases? Partitioned databases help you grow incrementally.
- Are users having to wait as other users access the database?
- Do you want to save disk space by giving users access to data stored in a remote location?
- Should you reduce network traffic by replicating data in several locations?
- Do you need to control database outlines from a central location?

When Not to Partition a Database

Sometimes, it does not make sense to partition a centralized database. Partitioning a database can require additional disk space, network bandwidth, and administrative overhead. Review the following list of questions. If you find yourself answering yes to many of them, or answering yes to some that are very important to you, you may not want to partition your database.

- Do you have resource concerns? For example, are you unable to purchase more disk space or allow more network traffic?
- Do you perform complex allocations where unit level values are derived from total values?
- Are you required to keep all databases online at all times? This could be a problem if you have databases in several time zones, because peak user load may differ between time zones. Using linked and transparent partitions would exacerbate this problem, but using replicated partitions could help.
- Are the databases in different languages? Essbase can only partition databases if both databases use the same language, such as German.

Determining Which Data to Partition

When designing a partitioned database, you need to determine:

- Which database should be the data source and which the data target? The database that “owns” the data should be the data source. Owning the data means that this is the database where the data is updated and where most of the detail data is stored.
- Are some parts of the database accessed more frequently than others?
- What data can you share among multiple sites?
- How granular does the data need to be at each location?
- How frequently is the data accessed, updated, or calculated?
- What are your resources? How much disk space do you have? CPUs? Network resources?
- How much data needs to be transferred over the network? How long does that take?
- Where is the data stored? Is it in one location or in more than one location?

- Where is the data accessed? Is it in one location or in more than one location?
- Is there information in separate databases that should be accessed from a central location? How closely are groups of data related?

The answers to these questions determine which data to include in each partition. For examples, see “Scenarios for Designing Partitioned Databases” on page 342.

Note: You cannot partition attribute dimensions.

Deciding Which Type of Partition to Use

You can create the following types of partitions:

- A *replicated partition* is a copy of a portion of the data source that is stored in the data target. For more information, see “Replicated Partitions” on page 319.
- A *transparent partition* allow users to access data from the data source as though it were stored in the data target. The data is, however, stored at the data source, which can be in another application, in another Essbase database, or on another OLAP Server. For more information, see “Transparent Partitions” on page 325.
- A *linked partition* sends users from a cell in one database to a cell in another database. This gives users a different perspective on the data. For more information, see “Linked Partitions” on page 333.

Replicated Partitions

A replicated partition is a copy of a portion of the data source that is stored in the data target. Some users can then access the data in the data source while others access it in the data target.

In the Samppart and Sampeast applications shipped with Essbase, for example, the database administrator at The Beverage Company (TBC) created a replicated partition between the East database and the Company database containing Actual, Budget, Variance, and Variance%. Users in the eastern region now store their budget data locally. Because they don’t have to retrieve this data live from the corporate headquarters, their response times are faster and they have more control over the down times and administration of the local data. For a more complete description of the sample partitioned databases provided with Essbase, see “Scenario 1: Partitioning an Existing Database” on page 342.

Changes to the data in a replicated partition flow from the data source to the data target. Changes made to replicated data in the data target do not flow back to the data source. If users change the data at the data target, Essbase overwrites their changes when the database administrator updates the replicated partition.

The database administrator can prevent the data in the replicated portion of the data target from being updated. This setting (choose Settings from the Connect page of the Partition Wizard) takes precedence over access provided by security filters and is also honored by batch operations such as dataload and calculation. By default, replicated partitions are not updateable. For information on how to set a partition as updateable or not, see "Specifying the Partition Type and Connection Information" on page 353.

Rules for Replicated Partitions

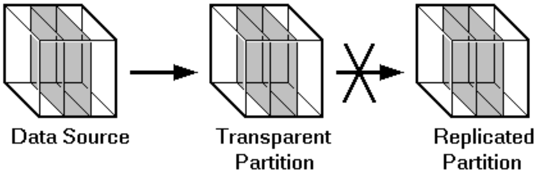
Replicated partitions must follow these rules:

- You must be able to map the shared replicated areas of the data source and data target outlines even though the shared areas do not have to be identical. This means that you must tell Essbase how each dimension and member in the data source maps to each dimension and member in the data target.

  The data source and data target outlines for the non-shared areas do not have to be mappable.

- You cannot create a replicated partition on top of a transparent partition. In other words, none of the areas that you use as a replicated partition source can come from a transparent partition target:

  Figure 134: Invalid Replicated Partition

- The cells in the data target of a replicated partition cannot come from two different data sources; the cells in one partition must come from just one database. If you want to replicate cells from more than one database, create a different partition for each data source.

The cells in a data target can be the data source for a different replicated partition. For example, if the Samppart Company database contains a replicated partition from the Sampeast East database, you could replicate the cells in the Sampeast East database into a third database, such as the Sampwest West database.

- You cannot use attribute dimension members to define a replicated partition. For example, associated with the Market dimension, the Market Type attribute dimension members are Urban, Suburban, and Rural. You cannot define a partition on Urban, Suburban, or Rural because a replicated partition contains dynamic data, not stored data. Hence, an attempt to map attributes in replicated partitions results in an error message. However, you can use the WITHATTR command to replicate attribute data in the Areas tab of the Partition Wizard.

For information on using Dynamic Time Series members in replicated partitions, see "Using Dynamic Time Series Members in Partitions" on page 814.

Advantages of Replicated Partitions

Replicated partitions can solve many database problems. Following are the advantages of using a replicated partition:

- Replicated partitions can decrease network activity, because the data is now stored closer to the end users, in the data target. Decreased network activity results in improved retrieval times for the users.
- The data is more easily accessible to all users. Some users access the data at the data source, others at the data target.
- Failures are not as catastrophic. Because the data is in more than one place, if a single database fails, only the users connected to that database are unable to access the information. It is still available at and can be retrieved from the other sites.
- Local database administrators can control the down time of their local databases. For example, because users in the eastern region are accessing their own replicated data instead of the Company database, the database administrator can bring down the Company database without affecting the users in the eastern region.

- Because only the relevant data is kept at each site, databases can be smaller. For example, users in the eastern region can replicate just the eastern budget information, instead of accessing a larger company database containing budget information for all regions.

Disadvantages of Replicated Partitions

Replicated partitions are not always the ideal partition type. Following are the disadvantages of using a replicated partition:

- You need more disk space, because you are storing the data in two or more locations.
- The data must be refreshed regularly by the database administrator, so it is not up-to-the-minute.

If these disadvantages are too serious, consider using transparent or linked partitions instead.

Performance Considerations for Replicated Partitions

To improve the performance of replicated partitions, consider the following when replicating data.

- Not replicating members that are dynamically calculated in the data source can greatly reduce replication time, because Essbase must probe the outline to find dynamically calculated members and their children to determine how to perform the calculation. For more information on dynamically calculated members, see Chapter 28, "Dynamically Calculating Data Values."
- Not replicating derived data from the data source can greatly reduce replication time. Instead, try to replicate the lowest practical level of each dimension and perform the calculations on the data target after you complete the replication.

For example, to replicate along the Market dimension:

- Define the shared area as the lowest level members of the Market dimension that you care about, for example, East, West, South, and Central and the level 0 members of the other dimensions.
- After you complete the replication, calculate the values for Market and the upper level values in the other dimensions at the data target.

Sometimes you cannot calculate derived data at the data target. In that case, you must replicate it from the data source. For example, you cannot calculate derived data at the data source if the data:

- Requires data outside the replicated area to be calculated.
- Requires calculation scripts from which you cannot extract just the portion to be calculated at the data target.
- Is being replicated onto a computer with little processing power, such as a laptop.

- Partitioning along a dense dimension takes more time than partitioning along a sparse dimension. When Essbase replicates data partitioned along a dense dimension, it must access every block in the data source and then create each block in the data target during the replication operation. For example, if the Market dimension were dense and you replicated the data in the East member, Essbase would have to access every block in the database and then create each block at the data target during the replication operation.
- You cannot replicate data into a member that is dynamically calculated at the data target. Dynamic Calc and Dynamic Calc And Store members do not contain any data until a user requests the data at run time. Essbase does not load or replicate into Dynamic Calc and Dynamic Calc And Store members. Essbase avoids sending replicated data for both dynamic dense and dynamic sparse members on the replication target, since this data is not stored on the data target. For more information on Dynamic Calc and Dynamic Calc And Store members, see Chapter 28, “Dynamically Calculating Data Values.”
- Use the Application Manager or ESSCMD to replicate only the data values that have changed instead of the entire partition.

When to Use a Replicated Partition

Use a replicated partition when you want to:

- Decrease network activity.
- Decrease query response times.
- Decrease calculation times.
- Make it easier to recover from system failures.

Replicated Partitions and Port Usage

One port is used for every unique user and machine combination. If a user defines several replicated partitions on one server using the same user name, then only one port is occupied.

In a replicated partition, when a user (user1) drills into an area in the target that accesses source data, user1 is using the user name declared in the partition definition (partition user) to access the data from the source database. This causes the use of an additional port because different users (user1 and partition user) are connecting to the application.

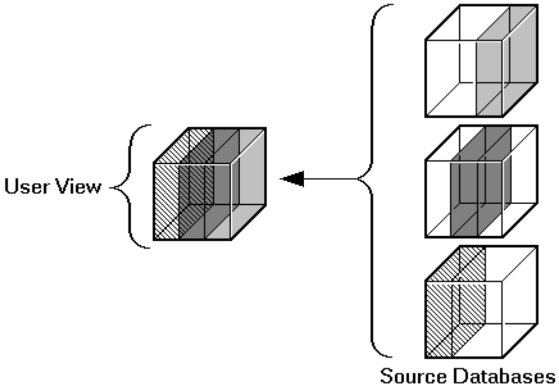
If a second user (user2) connects to the target database and drills down to access source data, user2 also uses the user name declared in the Partition Wizard (partition user) to access the source database. Because the partition user is already connected to the source database, an additional port is not needed for the partition user, as long as user2 is accessing the same source database.

Note: Because of the short-term nature of replication, replicated partitions and ports are rarely a problem.

Transparent Partitions

A transparent partition allows users to manipulate data that is stored remotely as if it were part of the local database. The remote data is retrieved from the data source each time that users at the data target request it. Users do not need to know where the data is stored, because they see it as part of their local database.

Figure 135: Transparent Partitions



Because the data is retrieved directly from the data source, users see the latest version of the data. When they update the data, their updates are written back to the data source. This means that other users at both the data source and the data target have immediate access to those updates.

If you create a transparent partition, users at the data source may notice slower performance as more users access the source data and users at the data target may notice slower performance as more users access the source data.

For example, the database administrator at TBC could use a transparent partition to calculate each member of the Scenario dimension on a separate CPU. This reduces the elapsed time for the calculation, while still providing users with the same view of the data. For a more complete description of the partitioned Sample Basic database, see “Scenario 1: Partitioning an Existing Database” on page 342.

These sections help you assess the value of transparent partitions for your site:

- "Rules for Transparent Partitions" on page 326
- "Advantages of Transparent Partitions" on page 328
- "Disadvantages of Transparent Partitions" on page 328
- "Performance Considerations for Transparent Partitions" on page 330
- "When to Use a Transparent Partition" on page 330
- "Calculating Transparent Partitions" on page 331
- "Guidelines for Transparent Calculations" on page 331
- "Transparent Partitions and Member Formulas" on page 332
- "Transparent Partitions and Port Usage" on page 333

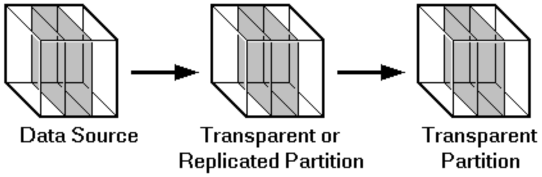
Rules for Transparent Partitions

Transparent partitions must follow these rules:

- The shared transparent areas of the data source and data target outlines do not have to be identical, but you must be able to map the dimensions in them. This means that you must tell Essbase how each dimension and member in the data source maps to each dimension and member in the data target.
- The data source and data target outlines for the non-shared areas do not have to be mappable. *Exception:* attribute associations should be identical. Otherwise, users can get incorrect results for some retrievals. For example, if product 100-10-1010 is associated with the Grape Flavor attribute on the source, but product 100-10-1010 is not associated with Grape on the target, the total of sales for all Grape flavors in New York would be incorrect.
- You cannot use attribute dimensions or members to define a transparent partition. For example, associated with the Market dimension, the Market Type attribute dimension has members Urban, Suburban, and Rural. You cannot define a partition on Urban, Suburban, or Rural.

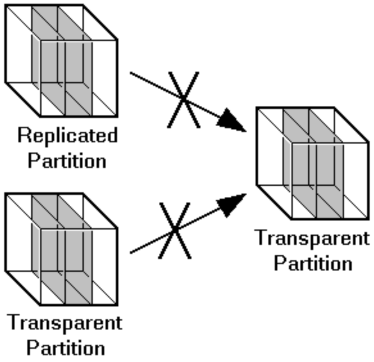
- You can create a transparent partition on top of a replicated partition. In other words, you can create a transparent partition target using a replicated partition source:

Figure 136: Valid Transparent Partition



- As illustrated in Figure 137, you cannot create a transparent partition on top of more than one other partition. In other words, you cannot create a transparent partition target from multiple sources. This is because each cell in a database must be retrieved from only one location—either the local disk or a remote disk.

Figure 137: Invalid Transparent Partition



- Carefully consider any formulas you assign to members in the data source and data target. For more information, see “Transparent Partitions and Member Formulas” on page 332.

For more information on using Dynamic Time Series members in transparent partitions, see “Using Dynamic Time Series Members in Partitions” on page 814.

Advantages of Transparent Partitions

Transparent partitions can solve many database problems. Following are the advantages of using a transparent partition:

- You need less disk space, because you are storing the data in one database.
- The data accessed from the data target is always the latest version.
- When the user updates the data at the data source, Essbase makes those changes at the data target.
- Individual databases are smaller, so they can be calculated more quickly.
- The distribution of the data is invisible to the end user and the end user's tools.
- You can load the data from either the data source or the data target.

Disadvantages of Transparent Partitions

Transparent partitions are not always the ideal partition type. Following are the disadvantages of using a transparent partition:

- Transparent partitions increase network activity, because Essbase transfers the data at the data source across the network to the data target. Increased network activity results in slower retrieval times for users.
- Because more users are accessing the data source, retrieval time may be slower.
- If the data source fails, users at both the data source and the data target are affected. This means that the network and data source must be available whenever users at the data source or the data target need them.
- You can perform some administrative operations only on local data. For example, if you archive the data target, Essbase archives just the data target and does *not* archive the data source. The following administrative operations work only on local data:
    - CLEARDATA calculation command
    - DATACOPY calculation command
    - EXPORT command

  - VALIDATE command
  - BEGINARCHIVE and ENDARCHIVE commands
  - Restructure operations in the Application Manager

- When you perform a calculation on a transparent partition, Essbase performs the calculation using the current values of the local data and transparent dependents. Essbase does not recalculate the values of transparent dependents. To calculate all partitions, issue a CALC ALL command for each individual partition, and then perform a CALC ALL command at the top level using the new values for each partition.

  Essbase does not recalculate the values of transparent dependents because the outlines for the data source and the data target may be so different that such a calculation would not be accurate.

  For example, suppose that the data target outline contained a Market dimension with East, West, South, and Central members and the data source outline contained an East dimension with New York and New Jersey members. If you tried to calculate the data target outline, you would assume that East was a level 0 member. In the data source, however, East is derived by adding New York and New Jersey. Any calculations at the data target, however, would not know this and could not reflect any changes made to New York and New Jersey in the data source. To perform an accurate calculation, therefore, you must first calculate East in the data source and then calculate the data target.

  For more information on transparent calculations, see “Calculating Transparent Partitions” on page 331.

- Formulas assigned to members in the data source may produce calculated results that are inconsistent with formulas or consolidations defined in the data target, and vice versa. For more information, see “Transparent Partitions and Member Formulas” on page 332.

If these disadvantages are too serious, consider using replicated or linked partitions instead.

Performance Considerations for Transparent Partitions

To improve the performance of transparent partitions, consider the following facts when creating the partition:

- Partitioning along dense dimensions in a transparent partition can greatly slow performance. This is because dense dimensions are used to determine the structure and contents of data blocks. If a database is partitioned only along a dense dimension at the target, Essbase must compose data blocks by performing network calls for the remote data in the transparent partition in addition to the disk I/O for the local portion of the block. To improve performance, consider including one or more sparse dimensions in the area definition so that the number of blocks required is limited to combinations with the sparse members.
- Basing transparent partitions on the attribute values of a dimension could increase retrieval time, because attributes are associated with sparse dimensions. In such cases, partitioning at a level higher than the level that is associated with attributes improves retrieval time. For example, in the Product dimension of the Sample Basic database, if children 100-10, 200-10, and 300-10 (level 0) are associated with attributes, then partition their parents 100, 200, and 300 (level 1) for better retrieval performance.
- Loading data into the data source from the data target can greatly slow performance. If possible, load data into the data source locally.
- Retrieval time is slower because users access the data over the network.
- Partitioning base dimensions can greatly slow performance.
- For calculation-related performance considerations, see “Guidelines for Transparent Calculations” on page 331.

When to Use a Transparent Partition

Use a transparent partition when you want to:

- Show users the latest version of the data.
- Allow users at the data target to update data.
- Decrease disk space.

Calculating Transparent Partitions

When you perform a calculation on a transparent partition, Essbase performs the calculation using the current values of the local data and transparent dependents. When calculating local data that depends on remote data, Hyperion Essbase performs a bottom-up calculation. The bottom-up calculation can be done only when the calculator cache on the target database is used properly. For complete information on bottom-up calculations, see "Using Bottom-Up Calculation" on page 1399. For information on the calculator cache, see "Sizing the Calculator Cache" on page 1327.

Increasing the amount of memory assigned to the calculator cache greatly improves calculation performance with transparent partitions. When a calculation is started, a message in the application log indicates whether or not the calculator cache is enabled or disabled on the target database. Using the calculator cache on the target database reduces the number of blocks that are requested from the data source during calculation. This, in turn, reduces the amount of network traffic that is generated by transferring blocks across the network. For information on estimating the size of the calculator cache, see "Sizing the Calculator Cache" on page 1327.

Guidelines for Transparent Calculations

Calculating data on the data target can greatly slow performance when itmust retrieve each dependent data block across the network, and then perform the calculation.

Performance with transparent calculations may also slow if Essbase must perform a top-down calculation on any portion of the data target that contains top-down member formulas. When the data target contains no top-down member formulas, it can perform a bottom-up calculation on the data target, which is much faster.

When Essbase performs the calculation on the data source, it can always perform a bottom-up calculation. For more information on top-down and bottom-up calculations, see "Using Bottom-Up Calculation" on page 1399.

Consider using these alternatives:

- Dynamic Calc or Dynamic Calc And Store members as parents of the transparent data so that the data is calculated on the fly when it's retrieved. This reduces the batch processing time for batch calculation. Essbase performs the calculation only when users request it.
- A replicated layer between the low-level transparent data and high-level local data.

Other performance strategies including:

- Keep the partition fully within the calculator cache area (see "Sizing the Calculator Cache" on page 1327). Keeping a partition fully within the calculator cache means that any sparse members in the partition definition must be contained within the calculator cache. For example, in the Sample Basic database, if a partition definition includes @IDESC(East), all descendants of East must be within the calculator cache.
- Enable the calculator cache, and assign a sufficient amount of memory to it. For more information, see "Sizing the Calculator Cache" on page 1327.
- Do not use complex formulas on any members that define the partition. For example, in Sample Basic, assigning a complex formula to New York or New Jersey (both children of East) forces Hyperion Essbase to use the top-down calculation method. For more information, see "Understanding Bottom-Up and Top-Down Calculation" on page 1400.

Transparent Partitions and Member Formulas

If the data target and data source outlines are identical except for different member formulas, make sure that your partition definition will produce the desired calculation results.

For example, suppose that the data source and data target outlines both contain a Market dimension with North and South members, and children of North and South. On the data target, Market is calculated from the data for the North and South members (and their children) on the data source. If any of these members on the data source contain member formulas, these formulas are calculated, thus affecting the calculated value of Market on the data target. These results may be different from how the Market member would be calculated from the North and South members on the data target, where these formulas may not exist.

Make sure that any formulas you assign to members in the data source and data target will produce the desired results.

Transparent Partitions and Port Usage

One port is used for every unique user and machine combination. If a user defines several transparent partitions on one server, using the same user name, then only one port is occupied.

In a transparent partition, when a user (user1) drills into an area in the target that accesses source data, user1 is using the user name declared in the partition definition (partition user) to access the data from the source database. This causes the use of an additional port because different users (user1 and partition user) are connecting to the application.

If a second user (user2) connects to the target database and drills down to access source data, user2 also uses the user name declared in the Partition Wizard (partition user) to access the source database. Because the partition user is already connected to the source database, an additional port is not needed for the partition user, as long as user2 is accessing the same source database.

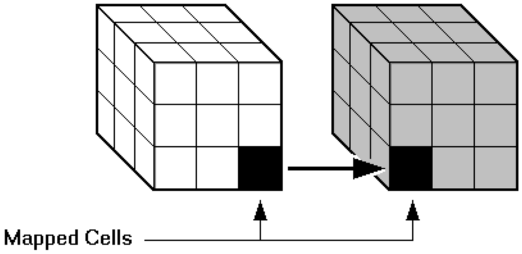
Linked Partitions

A linked partition connects two different databases with a data cell. When you click the linked cell in the data source, you drill across to a second database, the data target, and view the data there. If you are using Spreadsheet Add-in, for example, a new sheet opens displaying the dimensions in the second database. You can then drill down into these dimensions.

Unlike replicated or transparent partitions, linked partitions do not restrict you to viewing data in the same dimensionality as the target database. The database that you link to can contain very different dimensions than the database from which you connected.

To prevent users from seeing privileged data, you must establish security filters on both the data source and the data target. See "Setting Up Security for Partitioned Databases" on page 339.

Figure 138: Linked Partition



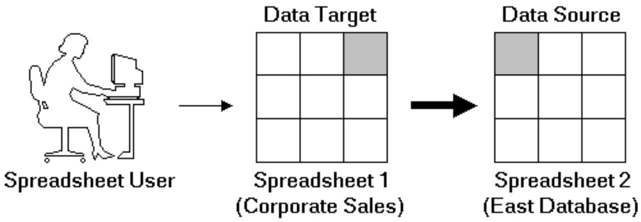
There are no performance considerations for linked partitions, beyond optimizing the performance of each linked database.

For example, if TBC grew into a large company, they would have several business units. Some data, such as profit and sales, exists in each business unit. TBC could store profit and sales in a centralized database so that the profit and sales for the entire company are available at a glance. The database administrator could link individual business unit databases to the corporate database. For an example of creating a linked partition, see "Scenario 3: Linking Two Databases" on page 347.

A user in such a scenario could perform these tasks:

- View the general profit and sales at the corporate level in a spreadsheet at the data target.
- Drill across to individual business units, such as east. This would open a new spreadsheet.
- Drill down in the new spreadsheet to more detailed data.

Figure 139: Source and Target for Linked Partition



For linked partitions, the spreadsheet that the user first views is connected to the data target, and the spreadsheet that opens when the user drills across is connected to the data source. This is the opposite of replicated and transparent databases, where users move from the data target to the data source.

Advantages of Linked Partitions

Linked partitions allow users to navigate to databases that contain different dimensions. Following are the advantages of using a linked partition:

- You can view data in a different context; that is, you can navigate between databases containing many different dimensions.
- You do not have to keep the data source and data target outlines closely synchronized, because less of the outline is shared.
- A single data cell can allow the user to navigate to more than one database. For example, the Total Profit cell in the Accounting database could link to the Profit cells in the databases of each business unit.
- Performance may improve, because you’re accessing the database directly and not through a data target.

Disadvantages of Linked Partitions

Linked partitions are not always the ideal partition type. Following are the disadvantages of using a linked partition:

- You must create an account for users on each database or default access to the destination database (such as through a guest account). See “Setting Up Security for Partitioned Databases” on page 339.
- Users must access the linked database using Essbase Release 5-aware tools. If you have custom built your tools, you must extend them using the Essbase Release 5 Grid API.

When to Use a Linked Partition

Use a linked partition when you want to connect databases with different dimensionality.

Linked Partitions and Port Usage

When accessing a linked partition, Essbase tries to use the end user's (user1) login information to connect to the source database. If user1 does not have access to the source database, Essbase looks for the linked partition default user name and password. If these defaults are not specified, user1 is requested to enter login information to access the source database. Port usage varies depending on the number of different user names being used to access the various source and target databases (and whether those databases are contained within the same or different servers).

Questions to Help You Choose a Partition Type

Table 21 should help you choose which type of partition to use.

Table 21: Types of Partition

Situation	Partition Type to Use
Decreasing network activity is more important than increasing disk space.	Replicated
It is acceptable for users to access data that's not up to the minute.	Replicated
You are concerned about a single failure disrupting an entire OLAP application.	Replicated
You do batch updates and simple aggregations.	Replicated
You must access the data using front-end tools that are not Distributed OLAP-aware.	Replicated or Transparent
You perform frequent updates and calculations.	Transparent
Users must update the data at the data target.	Transparent
Decreasing disk space is more important than increasing network activity.	Transparent or Linked
It is important that users access the absolute latest version of the data.	Transparent or Linked
Users want to view data in different application contexts.	Linked

Table 21: Types of Partition (Continued)

Situation	Partition Type to Use
You want to map attributes associated with base dimensions.	Transparent or Linked

After you have answered the questions in Table 21, you can compare the partition types in the following table.

Feature	Replicated	Transparent	Linked
Up-to-the-minute data		x	x
Reduced network traffic	x		x
Reduced disk space		x	x
Increased calculation speed	x		
Smaller databases		x	x
Improved query speed	x		x
Invisible to end users	x	x	
Access to databases with different dimensionality			x
Easier to recover	x		
Less synchronization required			x
Ability to query data based on its attributes		x	x

Using Attributes in Partitions

You can use attribute functions for partitioning on attribute values. But you cannot partition an attribute dimension. Use attribute values to partition a database when you want to access members of a dimension according to their characteristics.

For example, in the Sample Basic database, you cannot partition the Pkg Type attribute dimension. But you can create a partition that contains all the members of the Product dimension that are associated with either or both members (Bottle and

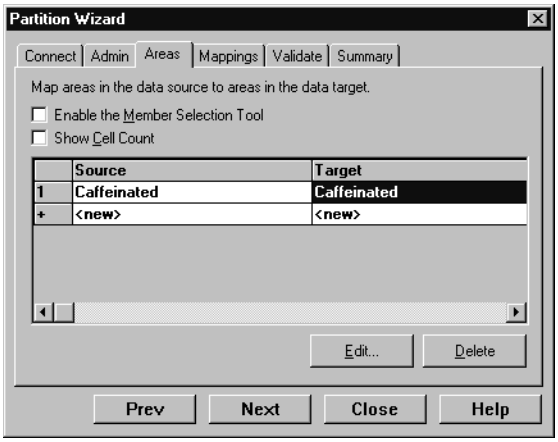
Can) of the Pkg Type dimension. If you create a partition that contains members associated with Can, you can access data only on Product members that are packaged in cans; namely, 100-10, 100-20, and 300-30.

You can use the @ATTRIBUTE command and the @WITHATTR command to define partitions.

For example, to extract data on all members of the Product dimension that are associated with the Caffeinated attribute dimension, you can create a partition such as @ATTRIBUTE (Caffeinated). But you cannot partition the Caffeinated attribute dimension.

Based on the previous example, this partition is incorrect:

Figure 140: Incorrect Partitioning



Partition Wizard

Map areas in the data source to areas in the data target.

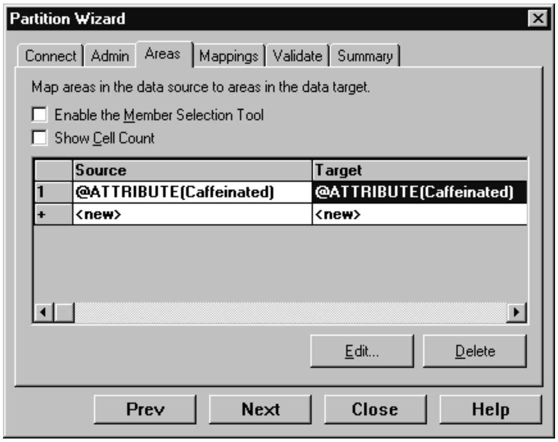
☐ Enable the Member Selection Tool

☐ Show Cell Count

	Source	Target
1	Caffeinated	Caffeinated
+	<new>	<new>

Based on the previous example, this partition is correct:

Figure 141: Correct Partitioning



Map areas in the data source to areas in the data target.

☐ Enable the Member Selection Tool

☐ Show Cell Count

	Source	Target
1	@ATTRIBUTE(Caffeinated)	@ATTRIBUTE(Caffeinated)
+	<new>	<new>

For more information about these commands, refer to the section on Calculation Commands in the *Technical Reference* in the `docs` directory.

For more information on attribute dimensions, see Chapter 9, “Working with Attributes.”

Setting Up Security for Partitioned Databases

Users accessing replicated, transparent or linked partitions may need to view data stored in two or more databases. The following sections describe how to set up security for each partition type so that users do not view or change inappropriate data.

Setting Up Security for Replicated Partitions

To set up security for users in a replicated partition, you should:

- Create filters at the data target to determine what end users can view and update as local data, just as you would for a standard Essbase database. See Chapter 15, "Managing Security for Users and Applications."
- Determine whether the partition is updateable at the data target.
  - If the partition is updateable, users can make changes to it, but these changes do not flow back to the data source and Essbase overwrites them when the administrator updates the replicated partition.
  - If the partition is not updateable, users cannot make changes to it at the data target.

  This setting overrides user filters that allow the user to update data. By default, replicated partitions are not updateable at the data target. See "Specifying the Partition Type and Connection Information" on page 353.

To set up security for the administrative account in a replicated partition, you should:

- Create an administrative account at both the data source and the data target. Essbase uses this administrative account to log into the data source to retrieve data when you update a replicated partition and to perform outline synchronization operations. See "Setting the User Name and Password" on page 354.
- If necessary, set up read filters on the administrative account at the data source to determine which data Essbase reads when replicating.
- If necessary, set up write filters on the administrative account at the data target to determine which data Essbase writes to when replicating.

Setting Up Security for Transparent Partitions

To set up security for users in a transparent partition, you should create filters at the data target to determine what end users can view and update as local data, just as you would for a standard Essbase database. Remember that, unlike replicated partitions, end users can update transparent partitions unless you create filters preventing them from doing so. See Chapter 15, "Managing Security for Users and Applications."

The administrative account performs all read and write operations requested by the data target for the data source. For example, when end users request data at the data target, the administrative account retrieves the data. When end users update data at the data target, the administrative account logs into the data source and updates the data there.

You can create filters on the administrative account in addition to filters on the end users. Filters on the administrative account can ensure that no one at the data target can view or update inappropriate data. For example, the administrator at the corporate database could restrict write access on certain cells to avoid relying on administrators in the various regions to set up security correctly for each end user.

To set up security for the administrative account in a transparent partition, you should:

- Create an administrative account at both the data source and the data target. Essbase uses this administrative account to log into the data source to retrieve data and to perform outline synchronization operations. See “Setting the User Name and Password” on page 354.
- Set up read filters on the administrative account at the data source to determine which data Essbase retrieves for end users.
- Set up write filters on the administrative account at the data source to determine which data Essbase updates for end users.

Setting Up Security for Linked Partitions

Users accessing linked databases may need to connect to two or more databases. To facilitate drill across access you can:

- Create accounts for each user on each database. For example, if Mary accesses data in a Company and an East database, create an account with the same login and password for Mary on both the Company and East databases. See “Creating, Editing, and Copying Users and Groups” on page 426.
- Create a default account that users can use when accessing target databases. For example, if users access data through a data source named Company and a data target named East, create a guest account for the East database with the appropriate privileges. Once you have created the account, use the guest account login and password as the default login when creating the linked partition. For more information on using default accounts in partitions, see “Specifying the Partition Type and Connection Information” on page 379.

For information on creating user accounts and filters, see Chapter 15, “Managing Security for Users and Applications.”

When a user drills across on data to a data target, Essbase logs the user into the data target using the following steps:

1. Checks to see if the user has an account on the data target with the same name and password. If so, Essbase logs the user in using that account.
2. Checks to see if you have specified a default account on the data target when you created your partition. If you did, Essbase logs the user in using that account.
3. Opens a login window prompting the user to enter a new login and password. Once the user enters a valid login and password, Essbase logs the user in using that account.

Scenarios for Designing Partitioned Databases

The following sections describe some basic ways to partition a database:

- From the top down. Top-down partitioning involves splitting a large database into several smaller ones.
- From the bottom up. Bottom-up partitioning involves connecting databases that contain related information.
- By linking related database. Linking databases allows users to drill across to databases with a very different dimensionality and then drill down to more detailed data.

Scenario 1: Partitioning an Existing Database

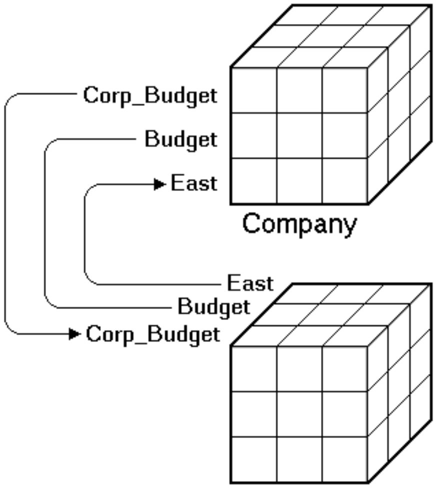
Let’s assume that TBC, the fictional soft drink company upon which the Sample Basic database is based, started out with a centralized database. As the eastern region grew, however, this was no longer feasible. The networks to the eastern region could not handle the large flow of data. Users were constantly waiting for data that they needed to make decisions. One day, the network went down and users at the eastern region couldn’t access the data at all.

Everyone agreed that the eastern region needed to access its own data directly, without going through the company database. In addition, TBC decided to change where budgeting information was stored. The corporate budget would stay at company headquarters, but the eastern region budget would move to the eastern region's database.

So, let's assume that TBC decided to ask you to partition their large centralized database into two smaller databases: Company and East.

This scenario is based on the Samppart application, which contains the Company database, and the Sampeast application, which contains the East database. Both are shipped with Essbase:

This illustration shows a subset of the partitioned databases. The arrows indicate flow from the data source to the data target. The Company database is the data source for the Corp_Budget member and the data target for the East and the East Actual members. The East database is the data source for its East and Actual members and the data target for the Corp_Budget member.



➤ Use this procedure to create a partition based on this example:

1. Determine which data to partition.

   TBC's Sample Basic database contains five standard dimensions: Year, Measures, Product, Market, and Scenario.

   - Partition the database along the East member of the Market dimension to give the eastern region more control over the contents of its database.
   - Partition the database along the Actual and Corp_Budget members of the Scenario dimension.

2. Choose the data source and the data target:

   – For Corp_Budget, use Company as source and East as Target. This is because the company owns the corporate budget, it will be the source.

   – For Eastern Region and Actual, East is the source and Company is the target, because the eastern region needs to update its market and actual information.

3. Decide the type of partition to use.

- For East, use transparent because the data target (Company) needs up-to-the-minute data.
- For Corp_Budget, use transparent because the data target (East) needs up-to-the minute data.
- For East Actual, use replication because the data target (Company) does not need up-to-the-minute data.

4. Finally, create the partitioned databases by:

   - Creating the new Sampeast application.
   - Creating the new East database by cutting the Company outline and pasting it into the East outline. Then delete the extra members (that is, South, West, and Central) and promote East.
   - If necessary, editing existing data sources, rules files, calculation scripts, report scripts, and outlines.
   - Creating the partitions.
   - Loading data into the new partitions.

Now that the corporate database is partitioned, users and database administrators see the following benefits:

- Faster response times, because they are competing with fewer users for the data and they are accessing the data locally.
- Database administrators can control the down time of their local databases, making them easier to maintain.
- Access to more data—now users can connect to both the eastern and corporate budgets.
- Higher quality data, because the corporate budget and eastern budget are now synchronized, they use the same data.

Scenario 2: Connecting Existing Related Databases

Let us assume that TBC has several databases, such as Inventory, Payroll, Marketing, and Sales. Users viewing the Sample Basic database want to share data with and navigate to those other databases and you, the database administrator, want to synchronize related data. It is impractical to combine all of the databases into one database, because:

- So many users access it that performance is slow.
- You could not find a down time to administer the database.
- No one has control over their own data, because it is centrally managed.
- The database is very sparse, because so much of the data is unrelated.

By connecting the databases instead, you can:

- Leverage work that’s already been completed.
- Synchronize the data.

So you decide to connect multiple databases.

Note: This scenario is not shipped with Essbase.

1. Determine which data to connect. First, connect the Inventory database.
   - Replicate the Opening_Inventory and Ending_Inventory members from the Measures dimension of the Inventory database into the Measures dimension of the Sample Basic database.
   - Do not replicate the Number_On_Hand, Number_Shipped, and Number_Returned members in the Measures dimension of the Inventory database to the Sample Basic database.
   - Add a link to the Inventory database so that users can view these more detailed measures if they need to.
   - Create a partition containing data from the Payroll, Marketing, and Sales databases in the Sample Basic database.
2. Choose the data source and the data target. In the case of the Opening_Inventory and Ending_Inventory members, the Inventory database is the data source and the Sample Basic database is the data target.
3. Decide which type of partition to use.

   You decide to use a replicated partition for the Opening_Inventory and Ending_Inventory members because the network connection is slow.
4. Connect the Payroll, Marketing, and Sales databases. Perform the tasks in Step 1 through Step 3 for each database.
5. Finally, create the partitioned databases by:
   - Editing existing data sources, rules files, calculation scripts, report scripts, and outlines
   - Creating the partitions
   - If necessary, loading data into the new partitions

Now that the Sample Basic database is partitioned, users and database administrators see the following benefits:

- Database administrators can control the down time of their local databases, making them easier to maintain.
- Access to more data—now users can link to new databases.
- Higher quality data, because the databases are now synchronized, that is, they use the same data.

Scenario 3: Linking Two Databases

Let us assume that TBC, the fictional soft drink company upon which the Sample Basic database is based, has two main databases the Sample Basic database and TBC Demo. Both databases have similar outlines, but TBC Demo has two additional dimensions: Channel, which describes where a product is sold, and Package, which describes how the product is packaged.

The database administrator for the Sample Basic database notices that more and more users are requesting that she add channel information to the Sample Basic database. But, since she doesn't own the data for channel information, she is reluctant to do so. She decides instead to allow her users to link to the TBC Demo database which already contains this information.

Note: This scenario is not shipped with Essbase.

Here are the steps to take:

1. Determine which data to link.

   The database administrator decides to link the Product dimension of the Sample Basic database to the Product dimension of TBC Demo. Users can then drill across to TBC Demo and view the Channel and Package information.

2. Choose the data source and the data target. Because users start at the Sample Basic database, it is considered the data target. Likewise, because users move to TBC Demo, it is considered the data source.

   Note: This is the opposite of replicated and transparent databases, where users move from the data target to the data source.

3. Decide on the type of partition to use.

Members	**Partition Type**	**Reasons**
Product	Linked	The database administrator wants to link from the Product dimension.

4. Finally, create the partition:
    - Establish a link from the Product member of the Sample Basic database to the Product dimension of the TBC Demo database. Remember to map the extra dimensions from TBC Demo, Channel and Product, to `void` in the Sample Basic database. For more information, see "Mapping Data Source Members to Data Target Members" on page 359.
    - Set up a guest account on TBC Demo that gives the users who connect from the Sample Basic database privileges to access the Channel and Package dimensions. For more information on creating accounts, see "Granting Permissions to Users and Groups" on page 416. For more information on assigning those accounts to linked partitions, see "Specifying the Partition Type and Connection Information" on page 379.

Now that the databases are linked, users and database administrators see the following benefits:

- Users have access to more data than before.
- The database administrator for the Sample Basic database does not need to maintain the TBC Demo database, all she needs to do is check the link periodically to make sure that it still works.

Chapter 14

Building and Maintaining Partitions

When you build a new partition, each database in the partition uses a partition definition file to record all information about the partition, such as its data source and data target and the areas to share. This chapter describes how to create a replicated, transparent, or linked partition.

CAUTION: You must design partitions carefully. We strongly recommend that you read Chapter 13, "Designing Partitioned Applications" before creating partitions.

This chapter contains the following sections on creating partitions:

Maintaining a partition after you create it can involve synchronizing the data source and the data target outlines, modifying the partition, or updating replicated partitions.

This chapter contains the following sections on maintaining partitions:

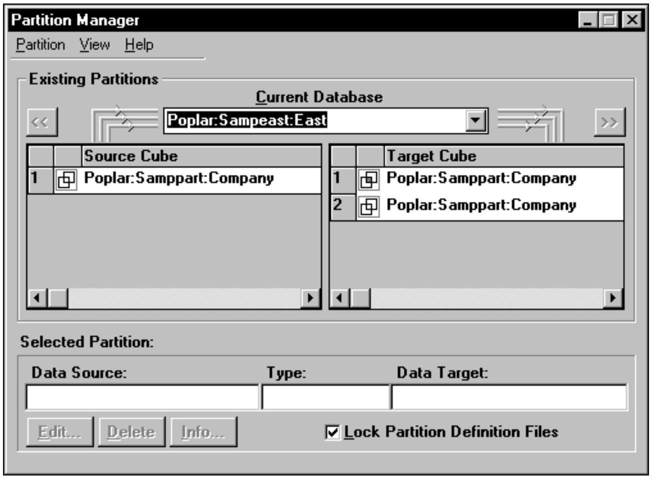
- "Introducing Outline Synchronization" on page 394
- "Editing Partitions" on page 403
- "Introducing the Updating of Replicated Partitions" on page 404
- "Updating Replicated Partitions Using Application Manager" on page 406
- "Updating Replicated Partitions Using MaxL or ESSCMD" on page 408
- "Troubleshooting Partitions" on page 408

Opening the Partition Manager

➤ To open the Partition Manager:

1. Select the database to partition; for example, East.
2. Choose Database > Partition Manager:

Figure 142: Partition Manager



Current Database: Poplar:Sampeast:East

		Source Cube
1	⊞	Poplar:Samppart:Company

		Target Cube
1	⊞	Poplar:Samppart:Company
2	⊞	Poplar:Samppart:Company

Data Source:	Type:	Data Target:

☑ Lock Partition Definition Files

Click Help for information about items in the Partition Manager.

The Partition Manager contains the following parts:

- Menus help you create and save new partitions; edit, delete, and open existing partitions; choose types of partitions displayed; and view online help.
- The Source Cube list box lists all the database partitions for which the currently selected database is the data target.
- The Target Cube list box lists all the database partitions for which the currently selected database is the data source.
- The arrow buttons, << and >>, change the currently selected database to the database in the selected partition. The buttons let you scroll through and select different databases.
- Other buttons let you edit, delete, or display information about a partition. To do so, select the partition in the Source Cube or Target Cube list box and click the appropriate button.
- An option to enable you to lock partition definition files so that other users cannot edit them while you are editing them. To do so, check the Lock Partition Definition Files box.

Creating a New Partition

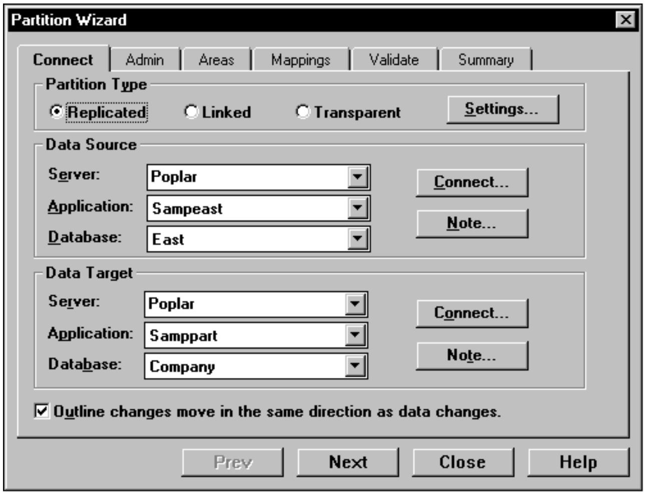
To create a new partition, choose Partition > New in the Partition Manager. Click the Help button for detailed information about each item in the Partition Wizard.

The Partition Wizard is part of the Partition Manager and contains several pages that you fill out to create a new partition. You must have Designer privileges or higher to create a partition.

Use the Connect page of the Partition Wizard to perform these tasks:

- Choose the type of partition type to use.
- Choose the server, application, and database containing the data source and the data target of the new partition.
- Specify which outline to synchronize outline changes to.

Figure 143: Connect Page of the Partition Wizard



Partition Wizard

Partition Type

◉ Replicated ○ Linked ○ Transparent

Data Source

Server: Poplar

Application: Sampeast

Database: East

Data Target

Server: Poplar

Application: Samppart

Database: Company

☑ Outline changes move in the same direction as data changes.

For more information on using Dynamic Time Series members in partitions, see “Using Dynamic Time Series Members in Partitions” on page 814.

Note: If you try to create a partition using a calculation function that returns a set of members, and the value returned by the calculation function evaluates to an empty set, the partition definition does not validate. An error is written to the application log stating that the region definition evaluated to an empty set.

Creating a Replicated Partition

A replicated partition is a copy of a portion of the data source that is stored in the data target. For more information, see "Replicated Partitions" on page 319.

The examples used in this section are based on the example described in "Scenario 1: Partitioning an Existing Database" on page 342. This example replicates the Eastern Region's Actual member in the Sampeast database (the data source) to the Samppart Company database (the data target).

Creating a replicated partition involves the following steps:

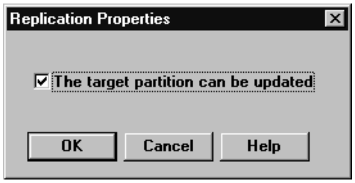
- "Specifying the Partition Type and Connection Information" on page 353
- "Setting the User Name and Password" on page 354
- "Defining the Areas in a Partition" on page 356
- "Mapping Data Source Members to Data Target Members" on page 359
- "Validating the Partition" on page 386
- "Saving the Partition Definition" on page 388

Specifying the Partition Type and Connection Information

➤ To set up a replicated partition:

1. Create a new partition. See "Creating a New Partition" on page 351.
2. Choose Replicated from the Connection page of the Partition Wizard.
3. Click the Settings button to open the Replication Properties dialog box.

Figure 144: Replication Properties Dialog Box



Replication Properties

☑ The target partition can be updated

4. Check “The target partition can be updated” to allow users to update the data at the data target or clear “The target partition can be updated” to prevent users from updating the data at the data target.

   Before setting this, consider that:

   - Essbase overwrites changes that users make at the data target when you update the replicated partition.
   - If users can’t update a replicated area, they cannot calculate it, load data into it, or change information in it using the Spreadsheet Add-in. Data in this area can only be changed at the data source.
   - Setting a partition as not updatable overrides security filters that let users update the partition.

5. Click OK.

6. Choose the server, application, and database names for the data source and the data target. If you are not connected to the appropriate server, click Connect. For our example, choose Sampeast and East as the data source and Samppart and Company as the data target. The server should correspond to the server on which your Samppart and Sampeast applications reside.

   Note: Do not use network aliases when creating partitions unless you are *certain* that they are propagated to all computers on your system. If you’re not certain, use the full server name. This is especially important for linked partitions, because this is the host name that clients connected to the data target use to find the data source.

7. If desired, enter notes about the data source or the data target by clicking the Note button to open the Note dialog box. Enter your notes and click OK.

8. To propagate changes in the data source’s outline to the data target’s outline, check the Outline changes move in the same direction as data changes box. To propagate changes from the data target’s outline to the data source’s outline, clear the box. If you’re not sure which outline to use as the source outline, see “Introducing Outline Synchronization” on page 394.

Setting the User Name and Password

You must specify a user name and password for Essbase to use when communicating between the data source and the data target. Essbase uses this user name and password to:

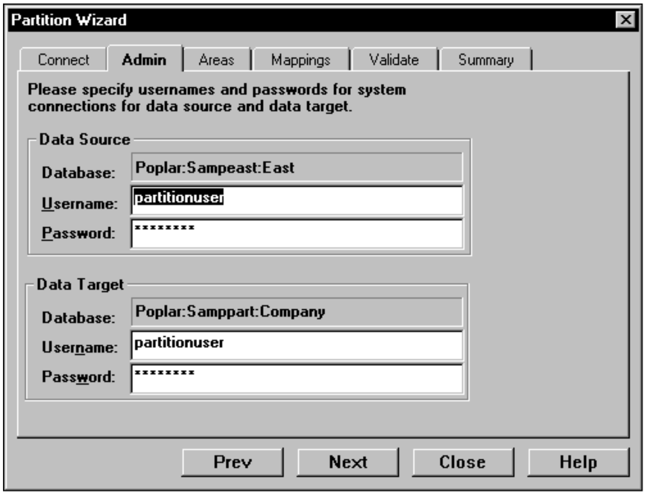
- Transfer data between the data source and the data target for replicated and transparent partitions. Local security filters apply to prevent end users from seeing privileged data.
- Synchronize database outlines for all partition types.

See "Setting Up Security for Partitioned Databases" on page 339.

➤ To set the user name and password for the data source and the data target:

1. From the Connection page of the Partition Wizard, click Next or Admin to move to the Admin page.

Figure 145: Admin Page of the Partition Wizard



Partition Wizard

Please specify usernames and passwords for system connections for data source and data target.

Data Source

Database: Poplar:Sampeast:East

Username: partitionuser

Password: ********

Data Target

Database: Poplar:Samppart:Company

Username: partitionuser

Password: ********

2. Enter the user name and password to use as the default login to the data source; for example, partitionuser and password. When you enter the password, the Partition Wizard masks it with asterisks (*) so that the password is hidden.
3. Enter the user name and password to use as the default login to the data target; for example, partitionuser and password. When you enter the password, the Partition Wizard masks it with asterisks (*) so that the password is hidden.

Note: This user name and password must be identical to the user name and password defined for the data source.

Defining the Areas in a Partition

The Areas page of the Partition Wizard lets you define or edit the areas of the data source to share with the data target. An *area* is a subcube within a database. For example, an area could be all Measures at the lowest level for Actual data in the Eastern Region. A partition is composed of one or more areas.

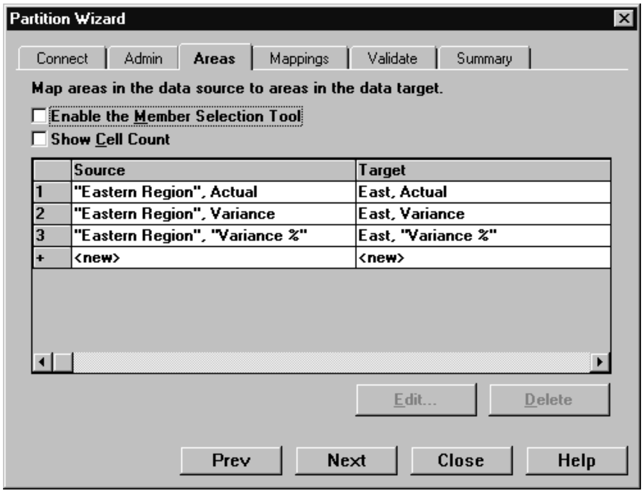
When you define a replicated area, make sure that both the data source and data target contain the same number of cells. This verifies that the two partitions have the same shape. For example, if the area covers 18 cells in the data source, there should be an area covering 18 cells in the data target into which to put those values.

Note: The cell count does not include those of attribute dimensions.

➤ To define an area in a partition:

1. From the Admin page of the Partition Wizard, click Next or Areas to move to the Areas page.

Figure 146: Areas Page of Partition Wizard



Map areas in the data source to areas in the data target.

□ Enable the Member Selection Tool

□ Show Cell Count

	Source	Target
1	"Eastern Region", Actual	East, Actual
2	"Eastern Region", Variance	East, Variance
3	"Eastern Region", "Variance %"	East, "Variance %"
+	<new>	<new>

2. Define the member or member combinations for an area by entering them in the Area Definition dialog box or choosing them from the Member Selection dialog box. If you're not sure which areas to partition, see "Determining Which Data to Partition" on page 318.

   There must be a one-to-one correspondence between cells in the areas for the data source and the data target on each line.

CAUTION: Do not select Use Aliases when creating area definitions and mapping. Although Partition Manager will validate the aliases, the partitions will not work.

Manually Defining an Area

➤ To enter the members in an area:

1. In the Areas page of the Partition Manager, make sure that the Enable the Member Selection Tool box is cleared.
2. Double-click the New field in the Source box (Figure 146) to open the Area Definition dialog box.
3. Enter the member name for the data source and click OK. For our example, enter:

   ```
   "Eastern Region", Actual
   ```

   Note: You can enter Essbase functions in the Area Definition dialog box. For example, you could share all descendants of East by entering `@DESCENDANTS(East)` or define areas based on user-defined attributes (UDAs). For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

4. Repeat Step 2 and Step 3 for the Target box. For our example, enter:

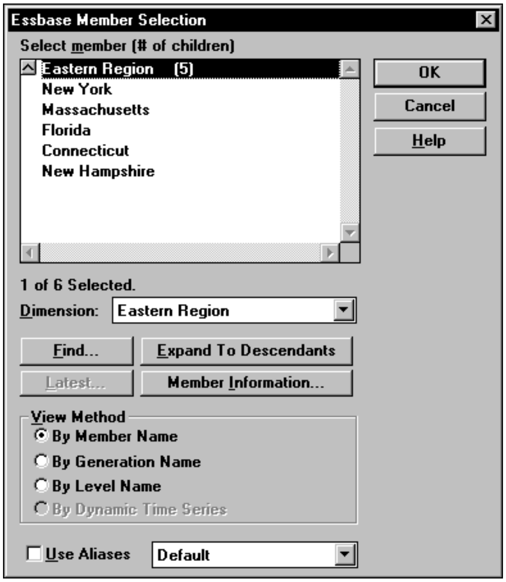
   ```
   East, Actual
   ```

Entering an Area Using the Member Selection Dialog Box

➤ To choose the members that make up an area from the Member Selection dialog box:

1. In the Areas page of the Partition Manager, make sure that the Enable Member Selection Tool box is checked.
2. Double-click the New field in the Source box (Figure 146) to open the Member Selection dialog box.

Figure 147: Member Selection Dialog Box



Essbase Member Selection

Select member (# of children)

- Eastern Region (5)
- New York
- Massachusetts
- Florida
- Connecticut
- New Hampshire

1 of 6 Selected.

Dimension: Eastern Region

View Method
- ● By Member Name
- ○ By Generation Name
- ○ By Level Name
- ○ By Dynamic Time Series

□ Use Aliases Default

For more information about the parts of the Member Selection dialog box, click Help.

3. Choose the dimension and member to partition. For our example, choose Eastern Region in the Market dimension and Actual in the Scenario dimension.
4. Click Add. Essbase adds the member to the Rules list.

   By default, the member's children and descendants are *not* added to the Rules list in Figure 147. To add the member's children or descendants to the Rules list, select the member in the Rules list and press the right mouse button to open a menu. Choose All Descendants.
5. Repeat Step 2 through Step 4 for the Target box. For our example, choose East in the Market dimension and Actual in the Scenario dimension.
6. When you are finished, click OK.

Note: You can only use the Member Selection dialog box to enter new members, not to edit existing members. To edit existing members, use the Member Name dialog box. See "Manually Defining an Area" on page 357.

Mapping Data Source Members to Data Target Members

Essbase must be able to map all shared data source members to data target members, to create a partition. If both, the data source and the data target, contain the same number of members and use the same member names, Essbase automatically maps the members. Skip to "Validating the Partition" on page 386.

Note: Use dimension member names instead of their aliases to create a valid partition.

It is recommended that the data source member names and the data target member names are the same. This reduces the maintenance requirements for the partition, especially when the partition is based on member attributes. See "Introducing Outline Synchronization" on page 394 for more information.

Mapping Members with Different Names

If the data source outline and data target outline contain different members or the members have different names in each outline, you must map the data source members to the data target members. In the following example, the first two member names are identical but the third member name is different:

Source **Target**

```
Product          Product
      Cola             Cola
Year             Year
     1998             1998
Market           Market
     East             East_Region
```

Because you know that East in the data source corresponds to Eastern_Region in the data target, map East to Eastern_Region. Then, all references to Eastern_Region in the data target point to East in the data source. For example, if the data value for Cola, 1998, East is 15 in the data source, the data value for Cola, 1998, Eastern_Region is 15 in the data target.

Mapping Data Cubes with Extra Dimensions

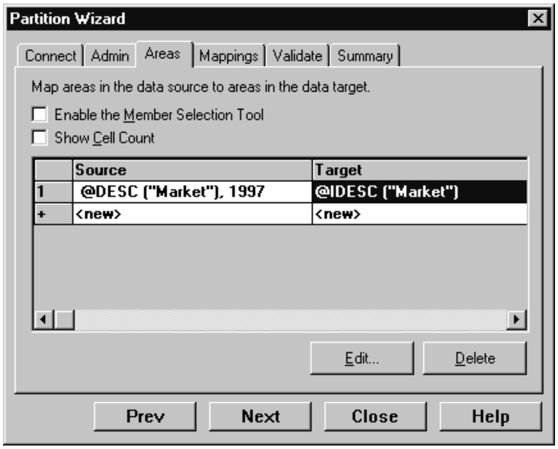
There can be instances when the number of dimensions in the data source and in the data target vary. The following example illustrates a case where there are more dimensions in the outline of the data source than in the outline of the data target:

Source **Target**

```
Product          Product
      Cola             Cola
Market           Market
     East             East
Year
     1999
     1998
     1997
```

If you want to map member 1997 of the Year dimension from the data source to the data target, you can map it to Void in the data target. But first, you must define the areas of the data source to share with the data target in the Areas tab of the Partition Wizard:

Figure 148: Defining Areas for Data Cubes with Extra Dimensions



Map areas in the data source to areas in the data target.

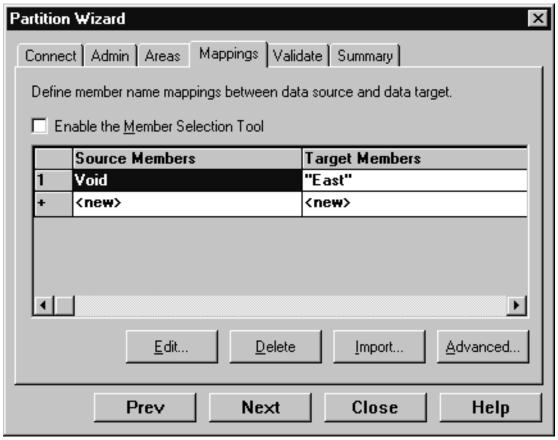
☐ Enable the Member Selection Tool

☐ Show Cell Count

	Source	Target
1	@DESC ("Market"), 1997	@IDESC ("Market")
+	<new>	<new>

You can then map the data source member to Void in the data target in the Mappings tab of the Partition Wizard:

Figure 149: Mapping Data Cubes with Extra Dimensions



Define member name mappings between data source and data target.

☐ Enable the Member Selection Tool

	Source Members	Target Members
1	Void	"East"
+	<new>	<new>

Enter the member name you are mapping to Void using either of these methods:

- Select the member name in the Member Select dialog box. See "Using the Member Selection Dialog Box to Enter Mappings" on page 365.
- Type the member name, using double quotes, in the Target Members column. See "Manually Defining Mappings" on page 385.

The corresponding Source Members column automatically displays "Void." Entering "Void" yourself may cause errors.

If you do not include at least one member from the extra dimension in the Areas definition, you will receive an error message when you attempt to validate the partition.

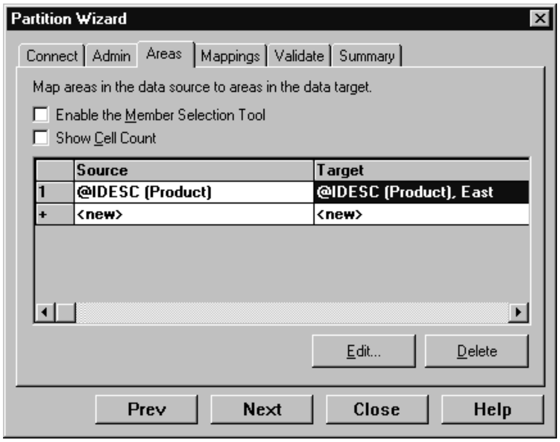
Note: When you map a member from an extra dimension, the partition results reflect data only for the mapped member. In the above example, the Year dimension contains three members: 1999, 1998, and 1997. If you map member 1997 from the data source to the data target, then the partition results reflect Product and Market data only for 1997. Product and Market data for 1998 and 1999 will not be extracted.

The following example illustrates a case where the data target includes more dimensions than the data source:

Source	**Target**
Product	Product
Cola	Cola
	Market
	East
Year	Year
1997	1997

In such cases, you must first define the shared areas of the data source and the data target:

Figure 150: Defining Areas for Data Cubes with Extra Dimensions



Map areas in the data source to areas in the data target.

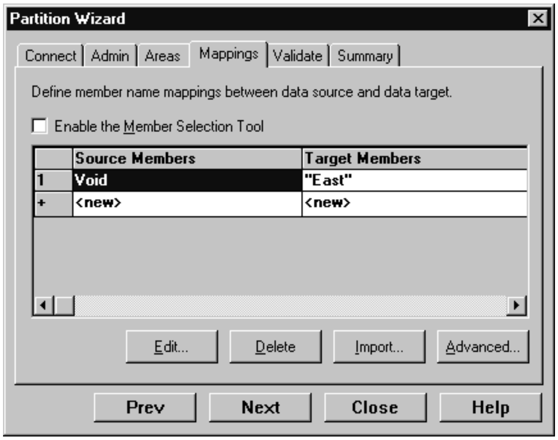
□ Enable the Member Selection Tool

□ Show Cell Count

	Source	Target
1	@IDESC (Product)	@IDESC (Product), East
+	<new>	<new>

You can then map member East from the Market dimension of the data target to Void in the data source:

Figure 151: Mapping Extra Dimensions in Partitioning



Define member name mappings between data source and data target.

□ Enable the Member Selection Tool

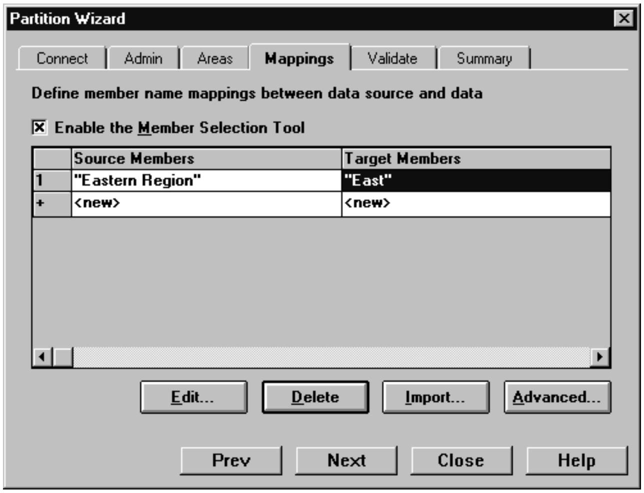
	Source Members	Target Members
1	Void	"East"
+	<new>	<new>

If member East from the Market dimension in the data target is not included in the target areas definition, you will receive an error message when you attempt to validate the partition.

➤ To map member names:

1. From the Areas page of the Partition Wizard, click Next or Mappings to move to the Mappings page.

Figure 152: Mappings Page of the Partition Wizard



Define member name mappings between data source and data

☒ Enable the Member Selection Tool

	Source Members	Target Members
1	"Eastern Region"	"East"
+	<new>	<new>

2. Map data source members to data target members in any of these ways:

   - Enter the name in the Member Name dialog box. See "Manually Defining Mappings" on page 365.
   - Select the member name in the Member Select dialog box. See "Entering an Area Using the Member Selection Dialog Box" on page 358.
   - Import the member mappings from an external data file. See "Importing Member Mappings" on page 367.

Note: You cannot map attributes dimension members in replicated partitions. For more information, refer to "Rules for Transparent Partitions" on page 326. You can, however, map attributes in transparent and linked partitions. For information on using attributes in partitions, see "Using Attributes in Partitions" on page 337. For information on mapping attributes, see "Mapping Attributes Associated with Members" on page 376.

Manually Defining Mappings

➤ To enter the member in the data source and the member that it maps to in the data target:

1. In the Mappings page of the Partition Wizard, make sure that the Enable the Member Selection Tool box is *cleared*.
2. Double-click the New field in the Source Members box (Figure 152) to open the Member Name dialog box.
3. Enter the member name for the data source and click OK. For our example, enter:

   ```
   "Eastern Region"
   ```
4. Double-click the New field in the Target Members box to open the Member Name dialog box.
5. Enter the member name to map it to in the data target and click OK. For our example, enter:

   ```
   East
   ```

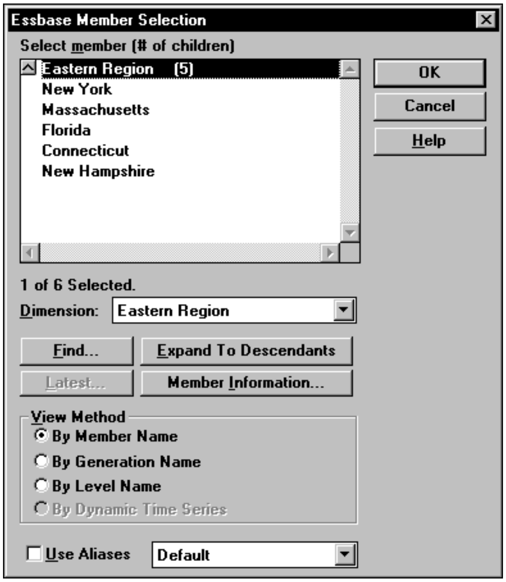
Using the Member Selection Dialog Box to Enter Mappings

➤ To choose the member in the data source and the member that it maps to in the data target from the Member Selection dialog box:

1. In the Mappings page of the Partition Wizard, make sure that the Enable Member Selection Tool box is checked.

2. Double-click the New field in the Source Members box (Figure 152) to open the Member Selection dialog box:

Figure 153: Member Selection Dialog Box



Essbase Member Selection

Select member (# of children)

Eastern Region (5)
New York
Massachusetts
Florida
Connecticut
New Hampshire

1 of 6 Selected.

Dimension: Eastern Region

View Method
● By Member Name
○ By Generation Name
○ By Level Name
○ By Dynamic Time Series

□ Use Aliases Default

For more information about the parts of the Member Selection dialog box, click Help.

3. Choose the dimension or member to map. For our example, choose the Eastern Region dimension.

4. Click OK.

5. Repeat Step 2 through Step 4 for the Target Members box. For our example, choose East in the Market dimension.

6. When you finish, click OK.

Note: For the linked partition described in “Scenario 3: Linking Two Databases” on page 347, you must also map the Package dimension to `Void`.

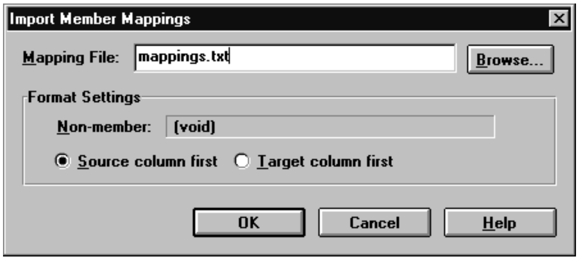
You can only use the Member Selection dialog box to enter new members, not to edit existing members. To edit existing members, use the Member Name dialog box. See “Manually Defining an Area” on page 357.

Importing Member Mappings

➤ To import the names from an external file:

1. In the Mappings page of the Partition Wizard, click Import to open the Import Member Mappings dialog box.

   Figure 154: Import Member Mappings Dialog Box

   Import Member Mappings

   Mapping File: mappings.txt

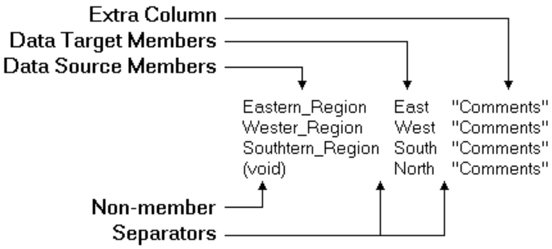
   Format Settings

   Non-member: (void)

   ● Source column first ○ Target column first

2. Choose the mapping file to import. Mapping files must end in `.TXT`. A sample member file must contain all of the following (except extra columns):

   Figure 155: Member Mapping Import File

   ```
   Eastern_Region   East   "Comments"
   Wester_Region    West   "Comments"
   Southtern_Region South  "Comments"
   (void)           North  "Comments"
   ```

   - Extra column—the file can contain extra columns that do not contain member names.
   - Data target members column—lists the member names in the data target. Member names containing spaces must be in quotes.
   - Data source members column—lists the member names in the data source. Member names containing spaces must be in quotes.

- Non-member column—missing members. Use when you are mapping an extra member in the data source to `Void` in the data target or vice versa.
- Separators—separate the columns. Separators can be tabs or spaces.

3. Click OK.

Creating a Transparent Partition

A transparent partition allow users to access data from the data source as though it were stored in the data target. The data is, however, stored at the data source, which can be in another application or in another Essbase database or on another OLAP Server. For more information, see "Transparent Partitions" on page 325.

The examples used in this section are based on the example described in "Scenario 1: Partitioning an Existing Database" on page 342. This example creates a transparent partition containing the Corp_Budget member between the Samppart Company database (the data source) and the Sampeast database (the data target).

Creating a transparent partition involves the following steps:

- "Specifying the Partition Type and Connection Information" on page 368
- "Setting the User Name and Password" on page 369
- "Defining the Areas in a Transparent Partition" on page 369
- "Mapping Data Source Members to Data Target Members" on page 373
- "Validating the Partition" on page 386
- "Saving the Partition Definition" on page 388

Specifying the Partition Type and Connection Information

➤ To set up a transparent partition:

1. Create a new partition. See "Creating a New Partition" on page 351.
2. Choose Transparent from the Connection page of the Partition Wizard.
3. Choose the server, application, and database names for the data source and the data target. If you are not connected to the appropriate server, click **Connect**. For our example, choose the Samppart Company database as the data source

and the Sampeast East database as the data target. The server should correspond to the server on which your Samppart and Sampeast applications reside.

Note: Do not use network aliases when creating partitions unless you are *certain* that they are propagated to all computers on your system. If you're not certain, use the full server name. This is especially important for linked partitions, because this is the host name that clients connected to the data target use to find the data source.

4. If desired, enter notes about the data source or the data target by clicking the Note button to open the Note dialog box. Enter your notes and click OK.
5. To propagate changes in the data source's outline to the data target's outline, check the Outline changes move in the same direction as data changes box. To propagate changes from the data target's outline to the data source's outline, clear the box. If you're not sure which outline to use as the source outline, see "Introducing Outline Synchronization" on page 394.

Setting the User Name and Password

The procedure for this is identical to creating the user name and password for a replicated partition. See "Setting the User Name and Password" on page 354.

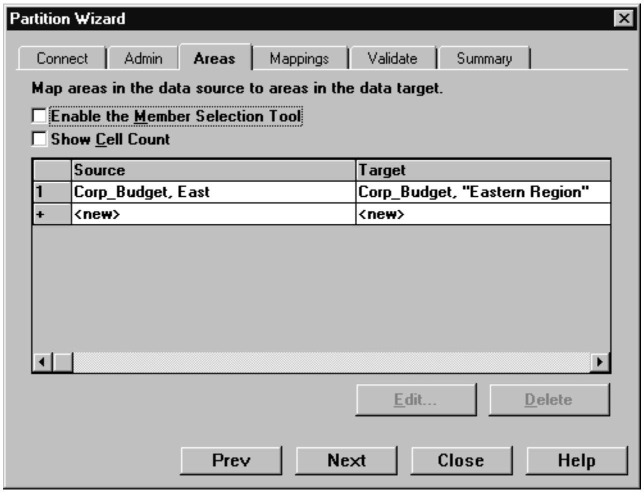
Defining the Areas in a Transparent Partition

The Areas page of the Partition Wizard lets you define or edit the areas of the data source to share with the data target. For more information about areas in a partition, see "Defining the Areas in a Partition" on page 356.

➤ To define an area in a transparent partition:

1. From the Admin page of the Partition Wizard, click Next or Areas to move to the Areas page.

Figure 156: Areas Page of Partition Wizard



Map areas in the data source to areas in the data target.

□ Enable the Member Selection Tool

□ Show Cell Count

	Source	Target
1	Corp_Budget, East	Corp_Budget, "Eastern Region"
+	<new>	<new>

2. Define the member or member combinations for an area by entering them in the Area Definition dialog box or choosing them from the Member Selection dialog box. If you're not sure which areas to partition, see "Determining Which Data to Partition" on page 318.

   There must be a one-to-one correspondence between cells in the areas for the data source and the data target on each line.

Manually Defining an Area

➤ To enter the members in an area:

1. In the Areas page of the Partition Manager, make sure that the Enable the Member Selection Tool box is *cleared*.

2. Double-click the New field in the Source box to open the Area Definition dialog box.

3. Enter the member name for the data source and click OK. For our example, enter:

   ```
   Corp_Budget, East
   ```

   Note: You can enter Essbase functions in the Area Definition dialog box. For example, you could share all descendants of East by entering `@DESCENDANTS(East)` or define areas based on UDAs. For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

4. Repeat Step 2 and Step 3 for the Target box. For our example, enter:

   ```
   Corp_Budget, "Eastern Region"
   ```

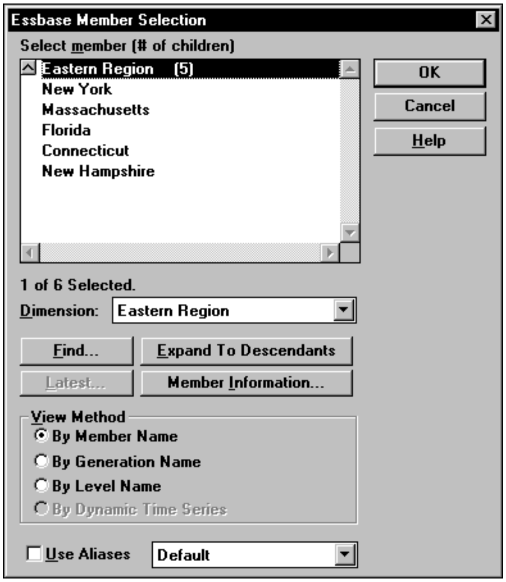
Using the Member Selection Dialog Box to Enter an Area

➤ To choose the members that make up an area from the Member Selection dialog box:

1. In the **Areas** page of the Partition Manager, make sure that the Enable Member Selection Tool box is checked.

2. Double-click the New field in the Source box to open the Essbase Member Selection dialog box.

Figure 157: Member Selection Dialog Box



Essbase Member Selection

Select member (# of children)

Eastern Region (5)
New York
Massachusetts
Florida
Connecticut
New Hampshire

1 of 6 Selected.

Dimension: Eastern Region

View Method
● By Member Name
○ By Generation Name
○ By Level Name
○ By Dynamic Time Series

□ Use Aliases Default

For more information about the parts of the Member Selection dialog box, click Help.

3. Choose the dimension and member to partition. For our example, choose Corp_Budget in the Scenario dimension and East in the Market dimension.

4. Click Add. Essbase adds the member to the Rules list.

   By default, the member’s children and descendants are *not* added to the Rules list. To add the member’s children or descendants to the Rules list, select the member in the Rules list and press the right mouse button to open a menu. Choose All Descendants.

5. Repeat Step 2 through Step 4 for the Target box. For our example, choose Corp_Budget in the Scenario dimension and Eastern Region in the Market dimension.

6. When you are finished, click OK.

Note: You can only use the Member Selection dialog box to enter new members, not to edit existing members. To edit existing members, use the Member Name dialog box. See "Manually Defining an Area" on page 357.

Mapping Data Source Members to Data Target Members

Essbase must be able to map all shared data source members to data target members to create a partition. If both the data source and the data target contain the same number of members and use the same member names, Essbase automatically maps the members. Skip to "Validating the Partition" on page 386.

For a detailed introduction to member mapping, see "Mapping Data Source Members to Data Target Members" on page 359.

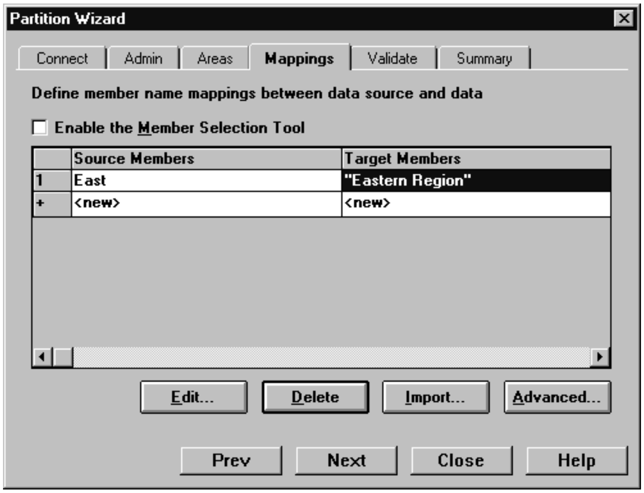
For information on using attributes in partitions, see "Using Attributes in Partitions" on page 337. For information on mapping attributes associated with members of a base dimension, see "Mapping Attributes Associated with Members" on page 376.

Note: When you create a replicated or transparent partition using a shared member, use the real member names in the mapping. Essbase maps the real member, not the shared one, from the data source.

➤ To map member names for a transparent partition:

1. From the Areas page of the Partition Wizard, click Next or Mappings to move to the Mappings page.

Figure 158: Mappings Page of the Partition Wizard



Define member name mappings between data source and data

□ Enable the Member Selection Tool

	Source Members	Target Members
1	East	"Eastern Region"
+	<new>	<new>

2. You can map data source members to data target members in any of the following ways:

   - Enter the name in the Member Name dialog box. See "Manually Defining the Mapping" on page 375.
   - Select the member name in the Member Select dialog box. See "Using the Member Selection Dialog Box to Enter Mappings" on page 365.
   - Import the member mappings from an external data file. See "Importing Member Mappings" on page 367.

Manually Defining the Mapping

➤ To enter the member in the data source and the member that it maps to in the data target:

1. In the Mappings page of the Partition Wizard, make sure that the Enable the Member Selection Tool box is *cleared*.
2. Double-click the New field in the Source Members box (Figure 158) to open the Member Name dialog box.
3. Enter the member name for the data source and click OK. For our example, enter:

   ```
   East
   ```

4. Double-click the New field in the Target Members box to open the Member Name dialog box.
5. Enter the member name to map it to in the data target and click OK. For our example, enter:

   ```
   "Eastern Region"
   ```

Using the Member Selection Dialog Box to Enter Mappings

➤ To choose the member in the data source and the member that it maps to in the data target from the Member Selection dialog box:

1. In the Mappings tab of the Partition Wizard, make sure that the Enable Member Selection Tool box is checked.
2. Double-click the New field in the Source Members box (Figure 158) to open the Member Selection dialog box.

   For information about the parts of the Member Selection dialog box, click Help.

3. Select the dimension or member to map. For our example, choose East in the Market dimension.
4. Click OK.

5. Repeat Step 2 through Step 4 for the Target Members box. For our example, choose Eastern Region in the Market dimension.
6. When you finish, click OK.

Note: You can only use the Member Selection dialog box to enter new members, not to edit existing members. To edit existing members, use the Member Name dialog box. See "Manually Defining an Area" on page 357.

Mapping Attributes Associated with Members

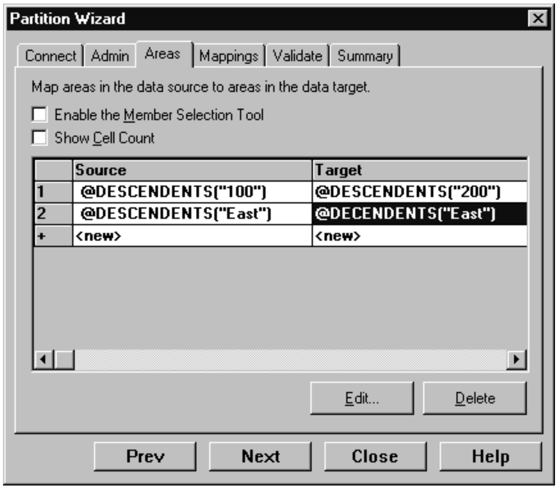
You must accurately map attribute dimensions and members from the data source to the data target to ensure that partitioning is valid.

In the following example, the outline for the data source contains a Product dimension with a member 100 (Cola). Children 100-10 and 100-20 are associated with member TRUE of the Caffeinated attribute dimension, and child 100-30 is associated with member FALSE of the Caffeinated attribute dimension.

The data target outline has a Product dimension with a member 200 (Cola). Children 200-10 and 200-20 are associated with member Yes of the With_Caffeine attribute dimension, and child 200-30 is associated with No of the With_Caffeine attribute dimension.

First define the areas to be shared from the data source to the data target in the Areas tab of the Partition Manager as shown in Figure 159:

Figure 159: Defining Areas for Mapping Attributes



Partition Wizard

Map areas in the data source to areas in the data target.

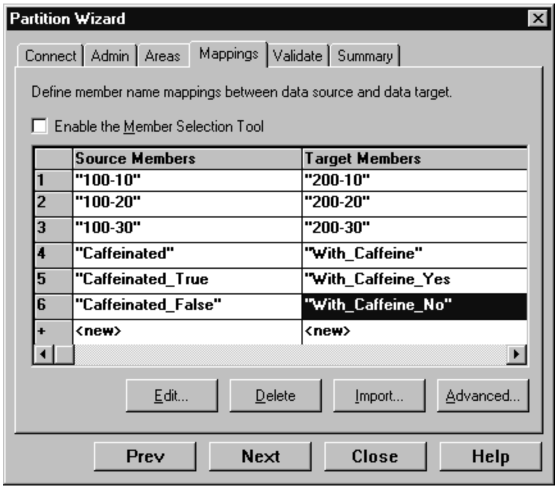
☐ Enable the Member Selection Tool

☐ Show Cell Count

	Source	Target
1	@DESCENDENTS("100")	@DESCENDENTS("200")
2	@DESCENDENTS("East")	@DECENDENTS("East")
+	<new>	<new>

Then map attributes in the Mapping tab of the Partition Manager:

Figure 160: Mapping Attributes



Define member name mappings between data source and data target.

□ Enable the Member Selection Tool

	Source Members	Target Members
1	"100-10"	"200-10"
2	"100-20"	"200-20"
3	"100-30"	"200-30"
4	"Caffeinated"	"With_Caffeine"
5	"Caffeinated_True	"With_Caffeine_Yes
6	"Caffeinated_False"	"With_Caffeine_No"
+	<new>	<new>

If you map attribute Caffeinated_True to attribute With_Caffeine_No, you receive an error message during validation. You must associate caffeinated cola from the data source to caffeinated cola in the data target.

Note: You can only map attributes for transparent and linked partitions. An attempt to map attributes in replicated partitions results in an error message.

There can be instances where an attribute dimension or an attribute member exists in the outline of the data source but not in the outline of the data target, or vice versa. For example:

Source **Target**

```
Caffeinated
      True
      False
```

In such cases, you have the following choices:

- Create the Caffeinated attribute dimension and its members in the outline of the data target and associate them with the Product dimension. You can then map the attributes from the data source to the data target.
- Map the Caffeinated attribute dimension in the data source to Void in the data target.

For more information on mapping, see “Mapping Data Cubes with Extra Dimensions” on page 360. For more information on attributes, see Chapter 9, “Working with Attributes.” For more information on using attributes in partitions, see “Using Attributes in Partitions” on page 337.

Note: You *must* validate the partition to ensure that the results are accurate. When you create a replicated or transparent partition using a shared member, use the real member names in the mapping. Essbase maps the real member, not the shared one, from the data source.

Creating a Linked Partition

A linked partition sends users from a cell in one database to a cell in another database. This gives users a different perspective on the data. For more information, see “Linked Partitions” on page 333.

The examples used in this section are based on the example described in “Scenario 3: Linking Two Databases” on page 347. This example links the Product dimension in the Sample Basic database to the TBC Demo database. This scenario is *not* shipped with Essbase.

Creating a linked partition involves the following steps:

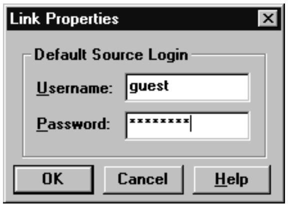
- “Specifying the Partition Type and Connection Information” on page 379
- “Setting the User Name and Password” on page 380
- “Defining the Areas in a Partition” on page 380
- “Mapping Data Source Members to Data Target Members” on page 383
- “Validating the Partition” on page 386
- “Saving the Partition Definition” on page 388

Specifying the Partition Type and Connection Information

➤ To set up a linked partition:

1. Create a new partition. See “Creating a New Partition” on page 351.
2. Choose Linked from the Connection page of the Partition Wizard.
3. Click the Settings button to open the Link Properties dialog box.

Figure 161: Link Properties Dialog Box



Username: guest

Password: ********

4. Enter the default login information for the data source. The user’s client application, such as the Spreadsheet Add-in, uses this information to connect to the data target if the user does not have login privileges to the source database using the same user name and password as the target database. For more information, see “Setting Up Security for Linked Partitions” on page 341.
5. Choose the server, application, and database names for the data source and the data target. If you are not connected to the appropriate server, click Connect. For our example, choose the TBC Demo database as the data source and the Sample Basic database as the data target. Remember, however, that the TBC Demo database is not shipped with Essbase.

Note: Do not use network aliases when creating partitions unless you are *certain* that they are propagated to all computers on your system. If you’re not certain, use the full server name. This is especially important for linked partitions, because this is the host name that the data target uses to find the data source.

6. If desired, enter notes about the data source or the data target by clicking the **Note** button to open the **Note** dialog box. Enter your notes and click **OK**. These notes display in the linked objects box to help users select the database to drill across to.
7. To propagate changes in the data source's outline to the data target's outline, check the **Outline changes move in the same direction as data changes** box. To propagate changes from the data target's outline to the data source's outline, clear the box. If you're not sure which outline to use as the source outline, see "Introducing Outline Synchronization" on page 394.

Setting the User Name and Password

The procedure for this is identical to creating the user name and password for a replicated partition. See "Setting the User Name and Password" on page 354.

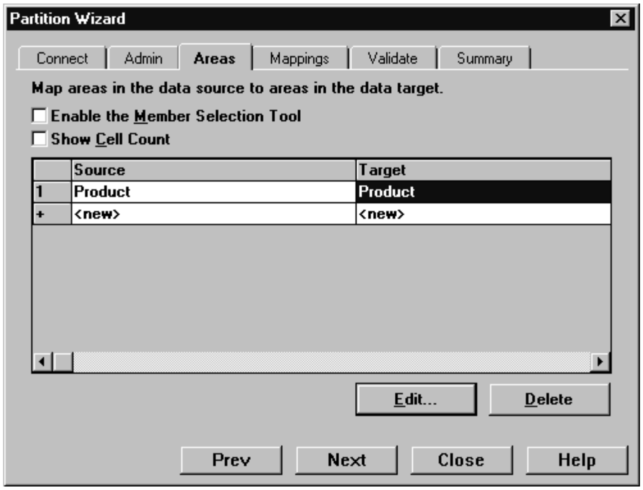
Defining the Areas in a Partition

The Areas page of the Partition Wizard lets you define or edit the areas of the data source to share with the data target. For more information about areas, see "Defining the Areas in a Partition" on page 356.

➤ To define an area in a linked partition:

1. From the Admin page of the Partition Wizard, click Next or Areas to move to the Areas page.

   Figure 162: Areas Page of Partition Wizard



Map areas in the data source to areas in the data target.

□ Enable the Member Selection Tool

□ Show Cell Count

	Source	Target
1	Product	Product
+	<new>	<new>

2. Define the member or member combinations for an area by entering them in the Area Definition dialog box or choosing them from the Member Selection dialog box. If you’re not sure which areas to partition, see “Determining Which Data to Partition” on page 318.

   There must be a one-to-one correspondence between cells in the areas for the data source and the data target on each line.

Manually Defining an Area

➤ To enter the members in an area:

1. In the Areas tab of the Partition Manager, make sure that the Enable the Member Selection Tool box is *cleared*.
2. Double-click the New field in the Source box to open the Area Definition dialog box.
3. Enter the member name for the data source and click OK. For our example, enter:

   ```
   Product
   ```

Note: You can enter Essbase functions in the Area Definition dialog box. For example, you could share all descendants of East by entering `@DESCENDANTS(East)` or define areas based on UDAs. For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

4. Repeat Step 2 and Step 3 for the Target box. For our example, enter:

   ```
   Product
   ```

Using the Member Selection Dialog Box to Enter an Area

➤ To choose the members that make up an area from the Member Selection dialog box:

1. In the Areas tab of the Partition Manager, make sure that the Enable Member Selection Tool box is checked.
2. Double-click the New field in the Source box to open the Member Selection dialog box. For more information about the parts of the Member Selection dialog box, click Help.
3. Choose the dimension and member to partition. For our example, choose the Product dimension.

4. Click **Add**. Essbase adds the member to the Rules list.

   By default, the member's children and descendants are not added to the Rules list. To add the member's children or descendants to the Rules list, select the member in the Rules list and press the right mouse button to open a menu. Choose All Descendants.

5. Repeat Step 2 through Step 4 for the Target box. For our example, choose the Product dimension.

6. When you are finished, click **OK**.

Note: You can only use the Member Selection dialog box to enter new members, not to edit existing members. To edit existing members, use the Member Name dialog box. See "Manually Defining an Area" on page 357.

Mapping Data Source Members to Data Target Members

Essbase must be able to map all shared data source members to data target members to create a partition. If both the data source and the data target contain the same number of members and use the same member names, Essbase automatically maps the members. Skip to "Validating the Partition" on page 386.

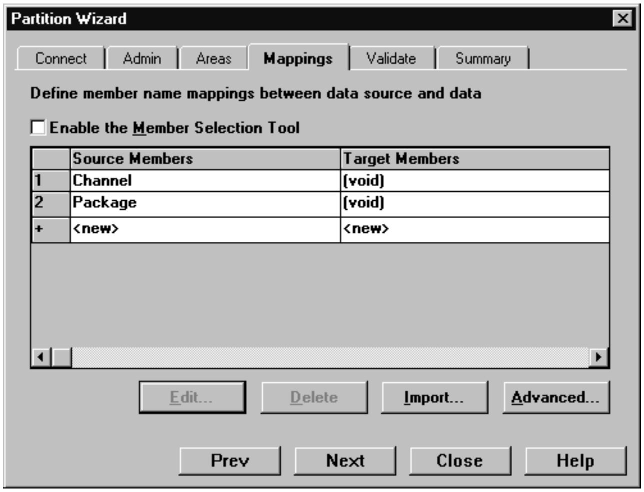
For a detailed introduction to member mapping, see "Mapping Data Source Members to Data Target Members" on page 359.

For information on using attributes in partitions, see "Using Attributes in Partitions" on page 337. For information on mapping attributes associated with members of a base dimension, see "Mapping Attributes Associated with Members" on page 376.

➤ To map member names for a linked partition:

1. From the Areas page of the Partition Wizard, click Next or Mappings to move to the Mappings page.

Figure 163: Mappings Page of the Partition Wizard



Define member name mappings between data source and data

☐ Enable the Member Selection Tool

	Source Members	Target Members
1	Channel	(void)
2	Package	(void)
+	<new>	<new>

2. You can map data source members to data target members in any of the following ways:

- Enter the name in the Member Name dialog box. See “Manually Defining Mappings” on page 365.
- Select the member name in the Member Select dialog box. See “Entering an Area Using the Member Selection Dialog Box” on page 358.
- Import the member mappings from an external data file. See “Importing Member Mappings” on page 367.

Manually Defining Mappings

➤ To enter the member in the data source and the member that it maps to in the data target:

1. In the Mappings page of the Partition Wizard, make sure that the Enable the Member Selection Tool box is *cleared*.
2. Double-click the New field in the Source Members box (Figure 163) to open the Member Name dialog box.
3. Enter the member name for the data source and click OK. For our example, enter:

   ```
   Channel
   ```

4. Double-click the New field in the Target Members box to open the Member Name dialog box.
5. Enter the member name to map it to in the data target and click OK. For our example, enter:

   ```
   Void
   ```

   For the linked partition described in "Scenario 3: Linking Two Databases" on page 347, you must also map the Package dimension to `Void`.

Using the Member Selection Dialog Box to Enter Mappings

➤ To choose the member in the data source and the member that it maps to in the data target from the Member Selection dialog box:

1. In the Mappings page of the Partition Wizard, make sure that the Enable Member Selection Tool box is checked.
2. Double-click the **New** field in the Source Members box (Figure 163) to open the Member Selection dialog box. For more information about the parts of the Member Selection dialog box, click Help.
3. Select the dimension or member to map. For our example, choose the Channel dimension.
4. Click OK.

5. Repeat Step 2 through Step 4 for the Target Members box. For our example, choose Void. For the linked partition described in "Scenario 3: Linking Two Databases" on page 347, you must also map the Package dimension to `Void`.
6. When you finish, click OK.

Note: You can only use the Member Selection dialog box to enter new members, not to edit existing members. To edit existing members, use the Member Name dialog box. See "Manually Defining an Area" on page 357.

Validating the Partition

When you create a partition, validate it to ensure that it's correct before you use it.

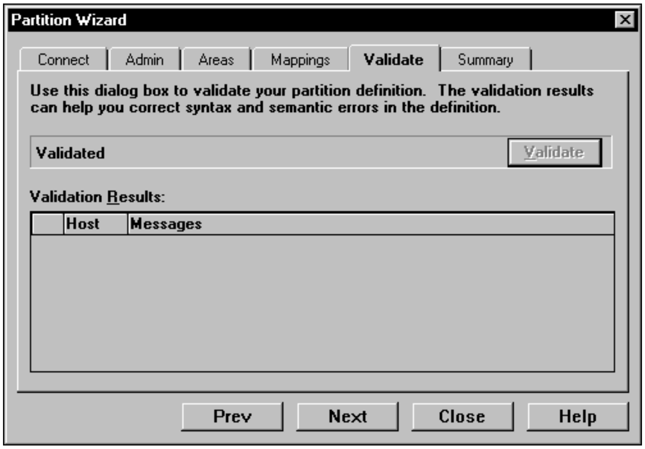
Note: In order to validate a partition, you must have Designer privileges or higher.

To save the partition without validating it, see "Saving the Partition Definition" on page 388.

➤ To validate a partition:

1. From the Mappings page in the Partition Wizard, click Next or Validate to move to the Validate page.

Figure 164: Validate Page of the Partition Wizard



Partition Wizard

Use this dialog box to validate your partition definition. The validation results can help you correct syntax and semantic errors in the definition.

Validated

Validation Results:

	Host	Messages

2. Click Validate. Essbase validates the partition. This may take some time.
3. If you receive a warning, double-click the warning message to move to the tab in the Partition Wizard where you can correct the problem shown in the warning message.

Tip: You can also use the VALIDATEPARTITIONDEFFILE command in ESSCMD to perform this task. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

When Essbase validates a partition definition, it checks on the server for the data source and the data target to ensure that:

- The area definition is valid (contains no syntax errors).
- The specified data source members are valid and map to valid members in the data target.
- All connection information is correct; that is, the server names, database names, application names, user names, and password information.
- For linked partitions, the default user name and password that you provide are correct.
- For replicated and transparent partitions, a replication target does not overlap with a replication target; a replication target does not overlap with a transparent target; a transparent target does not overlap with a transparent target; and a replication target does not overlap with a transparent target.
- For replicated and transparent partitions, the cell count for the partition is the same on the data source and the data target.
- For replicated and transparent partitions, the area dimensionality matches the data source and the data target.

Note: You must validate a transparent partition that is based on attribute values to ensure that the results are complete. Essbase does not display an error message when results are incomplete.

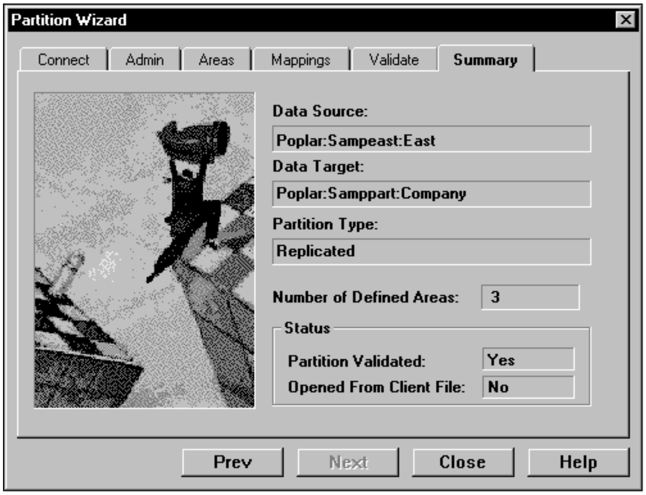
Saving the Partition Definition

You can save partition definitions even if they have not been validated.

➤ To save a partition definition:

1. From the Validate page of the Partition Wizard, click Next or Summary to open the Summary page.

Figure 165: Summary Page of the Partition Wizard



Partition Wizard

Data Source: Poplar:Sampeast:East

Data Target: Poplar:Samppart:Company

Partition Type: Replicated

Number of Defined Areas: 3

Status

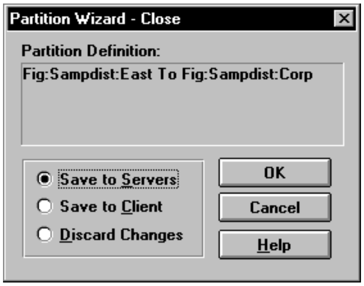
Partition Validated: Yes

Opened From Client File: No

Check the information in the Summary page to make sure that the partition is defined correctly. If it is not, click the appropriate page in the Partition Wizard to change the settings.

2. Click Close to open the Close dialog box.

Figure 166: Partition Wizard Close Dialog Box



Partition Wizard - Close

Partition Definition: Fig:Sampdist:East To Fig:Sampdist:Corp

● Save to Servers
○ Save to Client
○ Discard Changes

3. Choose where to save the partition definition:
   - Select Save to Servers to save the partition definition, which is stored in two .DDB files, to both the data source server and the data target server.
   - Select Save to Client to save the partition definition, which is stored in a single .DDB file, to your client machine. To open the partition definition later, choose Partition > Open From Client in the Partition Manager.
   - Select Discard Changes to throw away any changes that you made to a new or existing partition and close the Partition Manager.
4. Click OK to commit the definition to the specified location.

Note: If you are creating a replicated partition, you must populate it after you create it. To populate a replicated partition, see "Introducing the Updating of Replicated Partitions" on page 404.

Testing Partitions

➤ To test a partition:

- View the data targets using the Spreadsheet Add-in or other tool to make sure that the user sees the correct data.
- When testing a linked partition, make sure that Essbase links you to the expected database and that the default user name and password works correctly.

Creating Advanced Area-Specific Mappings (Optional)

If you can map all of the members in your data source to their counterparts in the data target using standard member mapping, then you do not need to perform advanced area-specific mapping. Skip to "Validating the Partition" on page 386.

If, however, you need to control how Essbase maps members at a more granular level, you may need to use area-specific mapping. Area-specific mapping maps members in one area to members in another area only in the context of a particular area map.

Use area-to-area mapping when you want to:

- Map data differently depending on where it is coming from.
- Map more than one member in your data source to a single member in your data target.

Since Essbase cannot determine how to map multiple members in the data source to a single member in the data target, you must logically determine how to divide your data until you can apply one mapping rule to that subset of the data. Then use that rule in the context of area-specific mapping to map the members.

Example 1: Advanced Area-Specific Mapping

The data source and data target contain the following dimensions and members:

```
Source             Target
Product            Product
      Cola               Cola
Market             Market
      East               East
Year               Year
      1998               1998
      1999               1999
                   Scenario
                         Actual
                         Budget
```

The data source does not have a Scenario dimension. Instead, it assumes that past data is actual data and future data is forecast, or budget, data.

You know that 1998 in the data source should correspond to 1998, Actual in the data target and 1999 in the data source should correspond to 1999, Budget in the data target. So, for example, if the data value for Cola, East, 1998 in the data source is 15, then the data value for Cola, East, 1998, Actual in the data target should be 15.

Because mapping works on members, not member combinations, you cannot simply map 1998 to 1998, Actual. You must define the area (1998 and 1998, Actual) and then create area-specific mapping rules for that area.

The data source does not have Actual and Budget members, so you must also map these members to Void in the data target.

➤ To create this example mapping:

1. Create the 1998 and 1999 and 1998, Actual and 1999, Budget areas:

   a. In the Areas tab of the Partition Wizard, define the area as 1998 in the data source and as 1998, Actual in the data target.

   b. In the Areas tab of the Partition Wizard, define the area as 1999 in the data source and as 1999, Budget in the data target.

   If you do not know how to use the Areas tab of the Partition Wizard, see "Defining the Areas in a Partition" on page 356.

2. Use advanced area-specific mapping to map the members that are missing in the data source (Actual and Budget) to `Void` so that the dimensionality is complete when going from the data source to the data target. For more information on making the dimensionality complete, see "Mapping Data Source Members to Data Target Members" on page 359.

   a. Click the Advanced button in the Mappings tab of the Partition Wizard.

   b. Use the arrow buttons to scroll the area selection to 1998 to Actual, 1998.

   c. Map `Void` in the data source to Actual in the data target.

   d. Click the left arrow button to define a new area-specific mapping. Scroll to 1999 to Budget, 1999.

   e. Map `Void` in the data source to Budget in the data target.

   f. Click Close.

Now Essbase maps the 1998 values to 1998, Actual and the 1999 values to 1999, Budget.

Example 2: Advanced Area-Specific Mapping

You can also use advanced area-specific mapping if the data source and data target are structured very differently but contain the same kind of information.

This strategy works, for example, if your data source and data target contain the following dimensions and members:

```
Source          Target
Market          Customer_Planning
      NY              NY_Actual
      CA              NY_Budget
                      CA_Actual
                      CA_Budget
Scenario
      Actual
      Budget
```

NY and Actual in the data source should correspond to NY_Actual in the data target and NY and Budget in the data source should correspond to NY_Budget in the data target. So, for example, if the data value for NY, Budget in the data source is 28, then the data value for NY_Budget in the data target should be 28.

Because mapping works on members, not member combinations, you cannot simply map NY, Actual to NY_Actual. You must define the area (NY and Actual, and NY_Actual) and then create area-specific mapping rules for that area.

Since the data target does not have NY and CA members, you must also map these members to Void in the data target so that the dimensionality is complete when going from the data source to the data target. For more information on making the dimensionality complete, see “Mapping Data Source Members to Data Target Members” on page 359.

➤ To create this example mapping:

1. Create the areas: NY, Actual, Budget and NY_Actual, NY_Budget; and CA, Actual, Budget and CA_Actual, CA_Budget.

   a. In the Areas tab of the Partition Wizard, define the area as NY, Actual, Budget in the data source and as NY_Actual, NY_Budget in the data target.

      If you don't know how to use the Areas tab of the Partition Wizard, see "Defining the Areas in a Partition" on page 356.

   b. Repeat these steps for CA, Actual, Budget and CA_Actual, CA_Budget.

   You split these into two areas so that you can map the members differently in each area in the next step.

2. Use advanced area-specific mapping to map the members that are missing in the data source (NY and CA) to `Void`.

   a. Click the Advanced button in the Mappings tab of the Partition Wizard.

   b. Use the arrow buttons to scroll the area selection to NY, Actual, Budget to NY_Actual, NY_Budget.

   c. Map NY in the data source to `Void` in the data target.

   d. Map Actual in the data source to NY_Actual in the data target.

   e. Map Budget in the data source to NY_Budget in the data target.

   f. Use the arrow buttons to scroll the area selection to CA, Actual, Budget to CA_Actual.

   g. Map CA in the data source to `Void` in the data target.

   h. Map Actual in the data source to CA_Actual in the data target.

   i. Map Budget in the data source to CA_Budget in the data target.

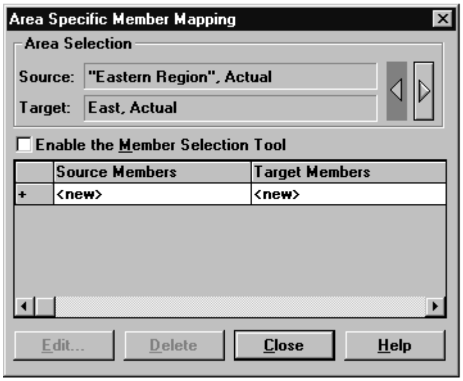
Now Essbase maps NY, Actual to NY_Actual; NY, Budget to NY_Budget; CA, Actual to CA_Actual; and CA, Budget to CA_Budget. That is, Essbase maps Actual to NY_Actual for the entire Actual, NY area and Actual to CA_Actual for the entire Actual, CA area. Essbase maps the Actual member to different members depending on the area.

Specifying Area-Specific Mapping

➤ To specify area-specific mapping:

1. Click Advanced in the Mappings tab of the Partition Wizard to open the Area Specific Member Mapping dialog box.

Figure 167: Area Specific Member Mapping Dialog Box



Area Specific Member Mapping

Area Selection

Source: "Eastern Region", Actual

Target: East, Actual

□ Enable the Member Selection Tool

	Source Members	Target Members
+	<new>	<new>

2. Enter the data source member in the Source Members text box.
3. Enter the data target member in the Target Members text box.
4. Map missing dimensions, `Void`, to their existing counterparts.
5. If desired, use the arrow keys to move to other area mappings.
6. When you are finished, click Close.

Introducing Outline Synchronization

When you partition a database, Essbase must be able to map each dimension and member in the data source outline to the appropriate dimension and member in the data target outline. After you map the two outlines to each other, Essbase can make the data in the data source available from the data target as long as the outlines are synchronized.

If you make changes to one of the outlines, the two outlines are no longer synchronized. Although Essbase does try to make whatever changes it can to replicated and transparent partitions when the outlines are not synchronized, Essbase may not be able to make the data in the data source available in the data target.

However, Essbase tracks changes that you make to your outlines and provides tools to make it easy to keep your outlines synchronized. This section describes the steps you need to take to synchronize your outlines.

Source Outline and Target Outline

Before you can synchronize your outlines, you must determine which outline is the source outline and which is the target outline.

- The *source outline* is the outline that outline changes are taken from.
- The *target outline* is the outline that outline changes are applied to.

By default, the source outline is from the same database as the data source; that is, outline and data changes flow in the same direction. For example, if the East database is the data source and the Company database is the data target, then the default source outline is East.

You can also use the data target outline as the source outline. You might want to do this if the structure of the outline (its dimensions, members, and properties) is maintained centrally at a corporate level, while the data values in the outline are maintained at the regional level (for example, East). This allows the database administrator to make changes in the Company outline and apply those changes to each regional outline when she synchronizes the outline.

To set the source outline, see “Specifying the Partition Type and Connection Information” on page 353.

Shared area changes can be propogated, target outline changes cannot:

- Shared area in the source outline, you can propagate these changes to the target outline when you synchronize the outlines.
- Target outline, those changes cannot be propagated back to the source outline when you synchronize the outlines. To move these changes up to the source outline, use the Outline Editor. See Chapter 7, “Creating and Changing Database Outlines.”

Essbase updates as many changes as possible to the target outline. If Essbase cannot apply all changes, a warning message prompts you to see the Application Server log for details. Messages that pertain to outline synchronization are prefixed with OUTLINE SYNC.

How Essbase Tracks Changes

The following table describes what happens when you change the source outline and then synchronize the target outline with the source outline:

Action You Take	Action Essbase Takes
Make changes to the source outline	**1.** Records the changes in a change log named `ESS`*`xxxxx`*`.CHG`, where *xxxxx* is the number of the partition. If you have more than one partition on a source outline, Essbase creates a change log for each partition. **2.** Creates or updates the outline change timestamp for that partition in the `.DDB` file. Each partition defined against the source outline has a separate timestamp in the `.DDB` file.
Pull changes from the outline source	**1.** Compares the last updated timestamp in the target outline's `.DDB` file to the last updated timestamp in the source outline's `.DDB` file. Essbase updates the target timestamp when it finishes synchronizing the outlines using the last updated time on the *source outline*, even if the two outlines are on servers in different time zones. **2.** If the source outline has changed since the last synchronization, Essbase retrieves those changes from the source outline's change log and places them in the target outline's change log. The change logs may have different names on the source outline and the target outline.
Select the changes to apply to the target outline	**1.** Applies the changes to the target outline. **2.** Updates the timestamp in the target outline's `.DDB` file, using the time from the source outline.

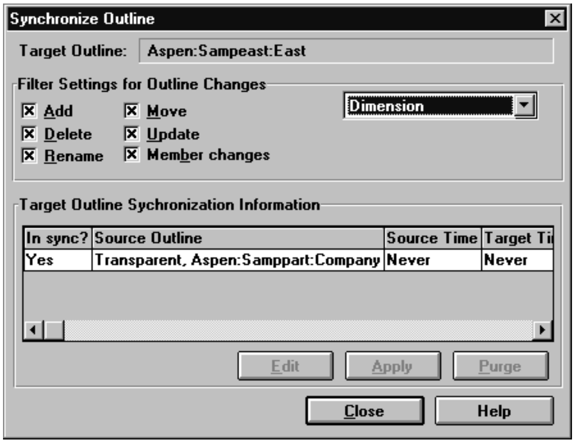
If you choose to *not* apply some changes, you cannot apply those changes later.

Synchronizing Outlines

➤ To synchronize a target outline to a source outline:

1. Connect to the source outline server and the target outline server, and select the target outline in the Essbase desktop.
2. Choose Database > Synchronize Outline to open the Synchronize Outline dialog box. Click the Help button for detailed information about each item in the Synchronize Outline dialog box:

Figure 168: Synchronize Outline Dialog Box



Synchronize Outline

Target Outline: Aspen:Sampeast:East

Filter Settings for Outline Changes

☒ Add ☒ Move
☒ Delete ☒ Update
☒ Rename ☒ Member changes

Dimension

Target Outline Sychronization Information

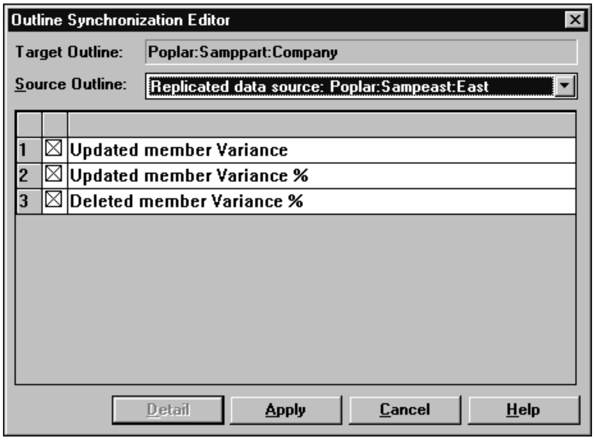
In sync?	Source Outline	Source Time	Target Ti
Yes	Transparent, Aspen:Samppart:Company	Never	Never

3. Choose the types of changes to retrieve from the source outline. These changes are not applied until you click Apply. The Apply button is enabled only when outline synchronization is necessary. If the database outlines are identical, the Apply button is not enabled.
   a. Choose Dimension from the list box. Select the types of changes to retrieve by checking their boxes.
   b. Choose Member from the list box. Select the types of changes to retrieve by checking their boxes.
   c. Choose Member Properties from the list box. Select the types of changes to retrieve by checking their boxes.

   Important: Clearing any of the Filter Settings for Outline Changes options, especially the Add and Move options, may result in losing additional changes. For example, if the database administrator creates a parent and children on the source outline and you clear the parent, the children may not get added to the target outline. If Essbase cannot apply all changes, a warning message prompts you to see the Application Server log file for details. Messages that pertain to outline synchronization are prefixed with OUTLINE SYNC.

4. To optionally choose which changes to accept or reject, click Edit to open the Outline Synchronization Editor dialog box.

Figure 169: Synchronization Editor Dialog Box



Outline Synchronization Editor

Target Outline: Poplar:Samppart:Company

Source Outline: Replicated data source: Poplar:Sampeast:East

1	☒	Updated member Variance
2	☒	Updated member Variance %
3	☒	Deleted member Variance %

This dialog box lists each change made to the source outline since you last synchronized the outlines. For information about each item in the dialog box, click Help.

a. Clear changes that you do not want to propagate. Be careful about not propagating changes. For example, if the database administrator creates a parent and children on the source outline and you clear the parent, you must also clear all of the children or the apply process fails.

   Important: Rejecting specific changes in the Outline Synchronization Editor may result in losing additional changes. For example, if the database administrator creates a parent and children on the source outline and you clear the parent, the children may not get added to the target outline. If the children are not added to the target outline, a warning message prompts you to see the Application Server log for details. Messages that pertain to outline synchronization are prefixed with OUTLINE SYNC.

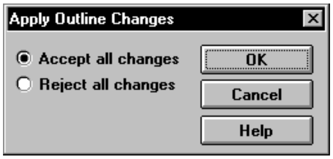
b. For more information about a specific change, double-click the change to open a Detail dialog box containing a brief description of the change.

c. When you are finished, click Apply. Essbase propagates the accepted change records.

d. Proceed to Step 6.

5. If you skipped Step 4, click Apply to open the Apply Outline Changes dialog box.

Figure 170: Apply Outline Changes Dialog Box



Apply Outline Changes

● Accept all changes
○ Reject all changes

6. Choose whether to accept or reject all changes and click OK.

   If you choose to accept changes, the outline synchronization may take some time to complete. If you choose to reject all changes, Essbase updates the timestamp and does not ask you to accept those changes the next time you synchronize the outlines.

7. To purge out-of-date change logs on both the target outline and the source outline, click Purge.

   Essbase deletes all records from the change log that have been applied or rejected. If all records have been applied or rejected, Essbase deletes the change log as well. Essbase does *not* purge records that have not yet been applied to the target outline.

8. Click Close to close the Synchronize Outline dialog box.

Essbase keeps all change logs until you purge them. You should purge change logs periodically, before they become too large.

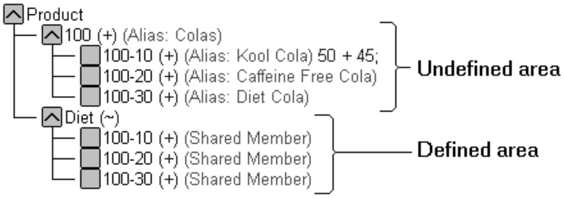
Every time you change the source outline or target outline, you should re-validate your partition definitions. See “Validating the Partition” on page 386.

Note: If you have moved a dimension or member past another dimension or member that is not in the partition, the move may not be accurately reflected in the target outline. See the last entry in the table in “Troubleshooting Partitions” on page 408.

Shared Members in Outline Synchronization

An actual member and its shared members in the source outline are propagated to the target outline if at least one actual or shared member is defined in the partition area. In the example illustrated in Figure 171, the partition definition is @IDESC(“Diet”). The parent 100 and its children (100-10, 100-20, 100-30) are not defined in the partition area. The parent Diet and its children (100-10, 100-20, 100-30) are defined in the partition area. The children of Diet are shared members of the actual members.

Figure 171: Shared Members and Outline Synchronization



```
Product
  100 (+) (Alias: Colas)
    100-10 (+) (Alias: Kool Cola) 50 + 45;
    100-20 (+) (Alias: Caffeine Free Cola)
    100-30 (+) (Alias: Diet Cola)
  Diet (~)
    100-10 (+) (Shared Member)
    100-20 (+) (Shared Member)
    100-30 (+) (Shared Member)
```

If you make a change to an actual member in the undefined partition area, such as adding an alias to the 100-10 actual member, that change is propagated to the target outline because it is associated with a shared member in the defined partition area.

The reverse is also true. If a shared member is not in the partition area and its actual member is, a change to the shared member in the undefined area is propagated to the target outline.

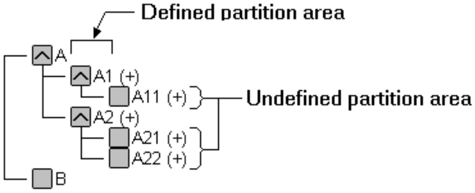
Any change made to a member that does not have at least one actual member (or shared member) in the defined partition area is not propagated to the target outline. For example, in Figure 171, a change to the parent 100 is not propagated to the target outline because it is in the undefined partition area and does not have an associated shared member in the defined partition area.

If a shared member is included in the partition area, then it is recommended to include its parent. In the above example, the parent Diet is included in the outline because its children are shared members and in the defined partition area.

Implied shared members are treated the same as shared members in Outline Synchronization. Actual members and their implied shared members in the source outline are propagated to the target outline if at least one actual or implied shared member is defined in the partition definition.

Using the partition definition as @CHILD("A") for the example illustrated in Figure 172, A1 and A2 are in the defined partition area, and A11, A21, A22 are in the undefined partition area. Although A11 (implied shared member) is in the undefined partition area, a change to A11 is propagated to the target outline because its parent, A1, is in the defined partition area. The change to the children A21 and A22 is not propagated to the target outline because these members are not defined in the partition area and are not associated with a member that is in the defined partition area.

Figure 172: Implied Shared Members and Outline Synchronization



The reverse is true again. If A1 is not defined in the partition area and its implied shared member is, then any change to A1 is propagated to the target outline.

You can synchronize outlines using ESSCMD. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

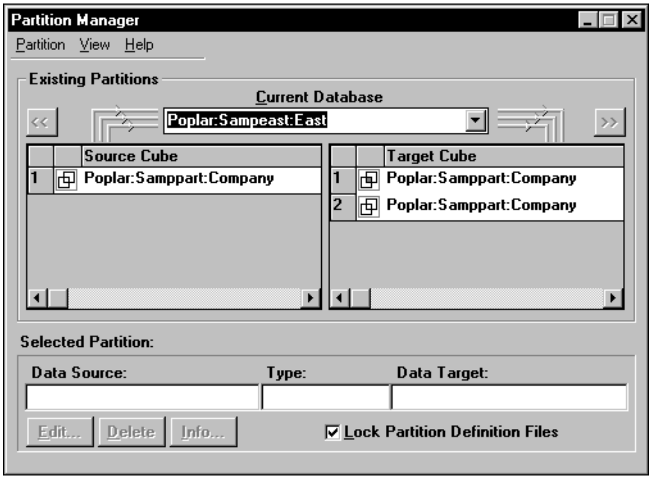
Task	**ESSCMD to Use**
Retrieve the outline change log from the source outline. You can choose specific changes from the source outline. You can choose to: • Get all changes from the outline. If you say Yes, Essbase retrieves all changes. • Get all dimension changes. If you say Yes, Essbase retrieves all changes and proceeds to the next step. If you say No, Essbase asks you which dimension changes to retrieve. You can retrieve: new dimensions, deleted dimensions, updated dimensions, moved dimensions, and renamed dimensions. • Get all member changes. If you say Yes, Essbase retrieves all changes and proceeds to the next step. If you say No, Essbase asks you which member changes to retrieve. You can retrieve: new members, deleted members, renamed members, or moved members. • Get all member property changes. If you say Yes, Essbase retrieves all changes and proceeds to the next step. If you say No, Essbase asks you which member property changes to retrieve. You can retrieve changed member: statuses, aliases, unary calculation symbols, account types, currency conversion information, user defined attributes, calculation formulas, level numbers, and generation numbers.	`GETPARTITIONOTLCHANGES`
Apply the changes in the change log to the target outline.	`APPLYOTLCHANGEFILE`
Reset the timestamp on both the source and the target outline change logs.	`RESETOTLCHANGETIME`
Purge entries that have been applied to the target outline from the change logs.	`PURGEOTLCHANGEFILE`

Editing Partitions

➤ To edit a partition:

1. Connect to the data source server and the data target server and select the data target outline in the Essbase desktop.
2. Choose Database > Partition Manager to open the Partition Manager.
   The Partition Manager displays all of the partitions defined for the current database. The Partition Manager shown in Figure 173 displays the replicated partition that you created between the Sampeast East and Samppart Company databases earlier in this chapter.

Figure 173: Partition Manager



Current Database: Poplar:Sampeast:East

	Source Cube
1	Poplar:Samppart:Company

	Target Cube
1	Poplar:Samppart:Company
2	Poplar:Samppart:Company

Data Source:

Type:

Data Target:

☑ Lock Partition Definition Files

3. Select the partition to edit and select the operation to perform on that partition:

Action To Take	Button to Click
Edit the partition	Edit or double-click the partition to open the Partition Wizard. Use the same tabs to edit the partition as you did to create it. For more information, see Chapter 14, "Building and Maintaining Partitions."
Delete the partition	Delete. Essbase deletes the partition definition from the partition's `.DDB` file on the data source and data target servers.
View more information about the partition	Info to open the Partition Information dialog box. You can also use the PRINTPARTITIONDEFFILE command in ESSCMD. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Introducing the Updating of Replicated Partitions

The database administrator should regularly update data in a replicated partition. How frequently you update replicated partitions depends on users' requirements for up-to-the-minute data. Essbase keeps track of when the data source was last changed and when the data target was last updated so that you can determine when to update replicated partitions. This information is saved at the data source. Either database administrator—that of the data source site or that of the data target site—can be responsible for replicating data.

Essbase also tracks which cells in a partition are changed. You can choose to update:

- **Just the cells that have changed since the last replication.** It's fastest to update just the cells that have changed.

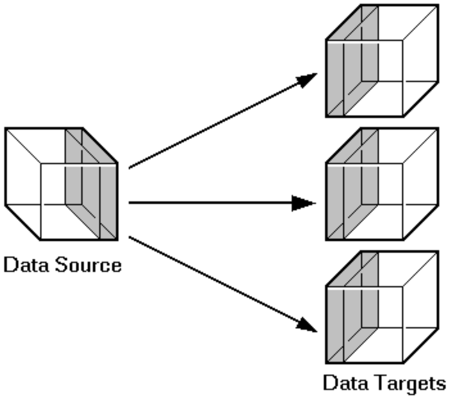
  Note: If you've deleted data blocks on the data source, Essbase updates all data cells at the data target even if you choose to update only changed cells. You can delete data blocks at the data source by using the CLEARBLOCK command in a calculation script; using "Clear combinations" in your rules file during a data load; issuing CLEAR UPPER, CLEAR INPUT or RESETDB commands in the Application Manager; restructuring the database keeping only level 0 or input data; or deleting sparse members.

- **All cells.** This is much slower. You may need to update all cells if you are recovering from a disaster where the data in the data target has been destroyed or corrupted.

You can replicate:

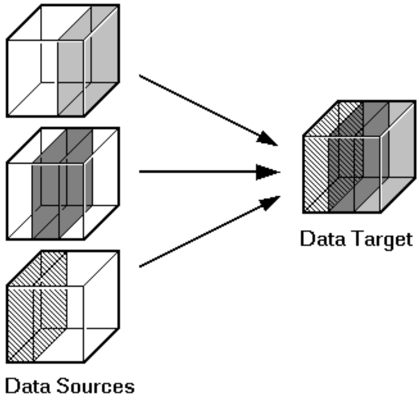
- All data targets connected to a data source. For example, if you replicate all data targets connected to the Sampeast East database, Essbase updates the Budget, Actual, Variance, and Variance % members in the Samppart Company database:

Figure 174: Put to Targets



- All data sources connected to a data target. For example, if you replicate from all data sources connected to the Samppart Company database, Essbase pulls the Budget, Actual, Variance, and Variance % members from the Sampeast East database and updates them in the Samppart Company database.

Figure 175: Get from Sources



Updating Replicated Partitions Using Application Manager

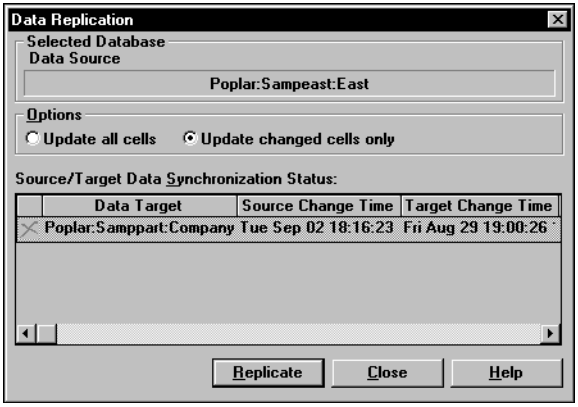
➤ To populate or update a replicated partition:

1. Connect to the data source server or the data target server and select the database in the Essbase desktop.

   If you opened the data source, choose Database > Replicate Data > To Targets. If you opened the data target, choose Database > Replicate Data > From Sources.

This opens the Data Replication dialog box.

Figure 176: Data Replication Dialog Box



Data Replication

Selected Database

Data Source

Poplar:Sampeast:East

Options

○ Update all cells ● Update changed cells only

Source/Target Data Synchronization Status:

	Data Target	Source Change Time	Target Change Time
✕	Poplar:Samppart:Company	Tue Sep 02 18:16:23	Fri Aug 29 19:00:26

The Data Replication dialog box lists all replicated partitions that are connected to this database and when they were last replicated. If the data source has changed since you last updated the data target, you should replicate the new information. You can choose which partitions to replicate.

Click the Help button for detailed information about each item in the Data Replication dialog box.

2. In the Options group, choose to update all data cells or just those that have changed. It is fastest to update just the changed cells. If you are not sure, see "Introducing the Updating of Replicated Partitions" on page 404.
3. Click Replicate. The replication may take some time to complete.

You must have Designer privileges or higher on the database from which you are initiating the replication. For example, if you are replicating from sources, you must have Designer privileges or higher on the target database. Likewise, if you are replicating to targets, you must have Designer privileges or higher on the source database.

Updating Replicated Partitions Using MaxL or ESSCMD

You can easily update replicated partitions using **refresh replicated partition** in MaxL. See the *Technical Reference* in the `docs` directory for information.

You can update replicated partitions using ESSCMD. See the *Technical Reference* in the `docs` directory for more information about the ESSCMD commands in the following table.

Task	ESSCMD to Use
Retrieve only the cells that have changed from the data source and update them on the current data target.	`GETUPDATEDREPLCELLS`
Retrieve all replicated cells from the data source and update them on the current data target.	`GETALLREPLCELLS`
Send only the cells that have changed from the current data source to the data target.	`PUTUPDATEDREPLCELLS`
Send all replicated cells from the current data source to the data target.	`PUTALLREPLCELLS`

Troubleshooting Partitions

The following table lists common problems that you may encounter when using partitions.

Symptom	Possible Causes	Solutions
When replicating to multiple data targets, some are not replicated.	The connection between the data source and one of the data targets was lost during the replication operation.	Retry the replication operation. If one database is unavailable, replicate into just the databases that are available.
Not all information arrived at the data target.	The data source and the data target outlines are no longer mappable.	Synchronize outlines for the data source and the data target and try again.

Symptom	Possible Causes	Solutions
You keep running out of ports.	Partitions connect to other databases using ports.	Purchase more ports.
When you try to access a partition, you cannot connect to it.	Someone has deleted, renamed, or moved the application containing the database to which you are trying to connect.	Open the Partition Manager, select the partition having problems, and click the Edit button. A dialog box opens asking you to repair the connection by specifying the new application name or location.
Partitioned databases can no longer connect to each other.	Your host names may not match. Did you use the `hosts` file to provide aliases to your local machine?	Make sure that the host names are synchronized between the servers and the Application Manager.
Essbase overwrites user edits.	Users are changing data at a replicated partition that you overwrite each time that you update the partition.	Set the partition to not allow user updates or explain to users why their data disappears.
The Synchronize Outline dialog box does not reflect outline change; that is, it lists the outlines as being in sync even though one outline has changed.	Was the target outline changed, but not the source outline? Essbase only propagates changes to the source outline. Does the outline change affect a defined partition? Essbase does not propagate changes to the outline that do not affect any partitions.	Open the Partition Manager and click Edit to examine the partition definition.
Data is confusing.	Your partition may not be set up correctly.	Check your partition to make sure that you are partitioning the data that you need.
Moved members or dimensions in the source outline are not reflected properly in a synchronized target outline.	Moving a dimension past a dimension that is not in the partition may not get propagated to the target during outline synchronization. Also, moving a member past a member that is not in the partition may not get propagated.	If possible, structure your outline so that members and dimensions are not moved across partitions. If this is not possible, change the target outline to reflect the source outline moved members or dimensions.

Part IV Designing and Building a Security System

This part describes how to control access to data in Essbase applications by designing and building a security system:

- Chapter 15, “Managing Security for Users and Applications,” describes how to create a security plan to manage security at the user and group layer, the global access layer, and the server level.
- Chapter 16, “Controlling Access to Database Cells,” describes how to define security filters and assign them to users.
- Chapter 17, “Security Examples,” contains detailed examples that illustrate how to implement a security system for Essbase applications.

Chapter 15

Managing Security for Users and Applications

Essbase provides a comprehensive system for managing access to applications, databases, and other objects. The Essbase security system provides protection in addition to the security available through your local area network.

This chapter contains the following sections:

About Security and Permissions

The Essbase security system addresses a wide variety of database security needs with a multi layered approach to enable you to develop the best plan for your environment. Various levels of permission can be granted to users and groups or defined at the system, application, or database scope. You can apply security in the following ways:

- **Users and groups:** To grant permissions to individual users and groups of users. When higher, these permissions take precedence over minimum permissions defined for applications and databases. Ordinary users have no inherent permissions. Permissions can be granted to users and groups by editing them or by using the **grant** statement in MaxL. For more information, see "Granting Permissions to Users and Groups" on page 416.

You can create users who log in using an external authentication protocol instead of the Essbase password. If you want users to use an outside authentication module such as LDAP, you must add an entry to the server configuration file essbase.cfg, and create the Essbase users with a reference to the external authentication. For more information, see "Managing External Authentication" on page 442.

- **Application and database settings:** To set common permissions for all users of an application or database, you can set minimum permissions that all users can have at each application or database scope. Users and groups with lower permissions than the minimum will gain access; users and groups with higher granted permissions are not affected. You can also temporarily disable different kinds of access using application settings. For more information, see "Managing Global Security for Applications and Databases" on page 442.
- **Server-wide settings:** Create and manage login restrictions for the entire OLAP Server. View and terminate current sessions and requests running on the entire OLAP Server or only on particular applications and databases. For more information, see "Managing User Activity at the Server Level" on page 451.
- **Database filters:** Define database permissions that users and groups can have for particular members, down to the individual data value (cell). To learn more about filters, see Chapter 16, "Controlling Access to Database Cells."

Table 22 describes all security permissions and the tasks that can be performed with those permissions.

Table 22: Description of Essbase Permissions

Permission	Affected Scope	Description
No Access or None	Entire system, application, or database	No inherent access to any users, groups, or resources, unless access is granted globally or by means of a filter. No access is the default when creating an ordinary user. Users with "No Access" can change their own passwords.
Read	Database	Ability to read data values.
Write	Database	Ability to read and update data values.

Table 22: Description of Essbase Permissions (Continued)

Permission	Affected Scope	Description
Execute (or Calculate)	Entire system, application, database, or single calculation	Ability to calculate, read, and update data values for the assigned scope, using the assigned calculation. Supervisors, application designers for the application, and database designers for the database can run calculations without being granted execute access.
Database Designer	Database	Ability to modify outlines, create and assign filters, alter database settings, and remove locks/terminate sessions and requests on the database. A user with Database Designer permission in one database does not necessarily have that permission in another.
Application Designer	Application	Ability to create, delete, and modify databases within the assigned application. Ability to modify the application settings, including minimum permissions, remove locks on the application, terminate sessions and requests on the application, and modify any object within the application. You cannot create or delete an application unless you also have been granted the system-level Create/Delete Applications permission. A user with Application Designer permission in one application does not necessarily have that permission in another.
Filter Access	Database	Ability to access specific data according to the restrictions of a filter assigned to the user or group. The filter definition specifies, for subsets of a database, whether Read, Write, or No Access is allowed for each subset. A user or group can be granted only one filter per database. Filters can be used in conjunction with other permissions. For more information, see Chapter 16, “Controlling Access to Database Cells.”

Table 22: Description of Essbase Permissions (Continued)

Permission	Affected Scope	Description
Create/delete applications	Entire system	Ability to create and delete applications and databases within those applications, and control permissions, locks, and resources for applications created. Includes designer permissions for the applications and databases created by this user.
Create/Delete Users, Groups	Entire system	Ability to create, delete, edit, or rename all users and groups having equal or lesser permissions than their own.
Supervisor	Entire system	Full access to the entire system and all users and groups.

Granting Permissions to Users and Groups

You can define security permissions for individual users and groups. Groups are collections of users that share the same minimum permissions. Users inherit all permissions of the group and can additionally have access to permissions that exceed those of the group. Users and groups are managed on a server-by-server basis: users defined on a server exist for all applications and databases on the server.

Note: User and group management is also available when you use Essbase Administration Services. For more information, see *Essbase Administration Services Online Help*.

Permissions can be granted to users and groups in the following ways:

- Specifying a user or group type upon creation or by editing the user or group. This method specifies system-level permissions that span all applications and databases. See "Specifying User and Group Types" on page 417.
- Granting permission to access applications or databases. See "Granting Application and Database Access to a User or Group" on page 419.
- Granting designer permissions to users or groups. See "Granting Designer Permissions to a User or Group" on page 421.

Specifying User and Group Types

One major way to assign permissions to users and groups is to define user and group types when you create or edit (modify the permissions of) the users and groups. You define these types in the New User or Edit User dialog boxes (see Figure 185 and Figure 187, respectively).

In Application Manager, users and groups can be created in different ways to specify their system-level (data-independent) permissions. These methods are represented in Application Manager and in Essbase Administration Services Console as user types. A description of these types follows. When using MaxL, you do not create user types: you grant the system-level permissions after the user is created; initially, as an ordinary user with no permissions.

For detailed instructions about creating users and groups in Application Manager, see "Creating, Editing, and Copying Users and Groups" on page 426.

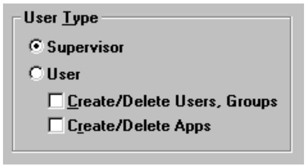
Creating a User with Supervisor Permission

A user with Supervisor permission has full access to the entire system and all users and groups.

The user who installs Essbase on the server is designated the Essbase System Supervisor for that server. Essbase requires that at least one user on each server has Supervisor permission. Therefore, you cannot delete or downgrade the permission of the last supervisor on the server.

When creating or editing a user in Application Manager, the selection shown in Figure 177 gives the user Supervisor permission:

Figure 177: Supervisor permission



User Type

● Supervisor

○ User

□ Create/Delete Users, Groups

□ Create/Delete Apps

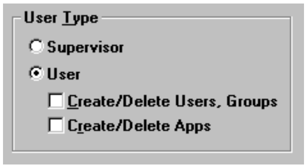
When using MaxL, create an ordinary user and then grant the role of supervisor in a separate statement. For example:

```
create user admin identified by 'password';
grant supervisor to admin;
```

Creating an Ordinary User (No Permission)

Users with ordinary permission have no inherent access to any users, groups, or resources. When creating or editing a user in Application Manager, the selection shown in Figure 178 gives the user no permission:

Figure 178: Ordinary User Permission



User Type

○ Supervisor

● User

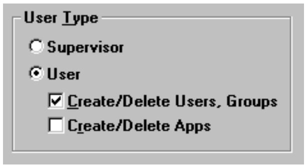
□ Create/Delete Users, Groups

□ Create/Delete Apps

Creating a User Who Can Create and Delete Users and Groups

Users with Create/Delete Users, Groups permission can create, delete, edit, or rename users and groups with equal or lower permissions only. When creating or editing a user in Application Manager, the selection shown in Figure 179 gives the user Create/Delete Users, Groups permission:

Figure 179: Create/Delete Users, Groups Permission



User Type

○ Supervisor

● User

☑ Create/Delete Users, Groups

□ Create/Delete Apps

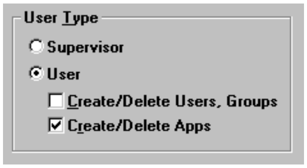
Creating a User Who Can Create and Delete Applications

Users with Create/Delete Applications permission can create and delete applications and control permissions and resources applicable to those applications or databases they created.

Users with Create/Delete Applications permission cannot create or delete users, but they can manage application-level permission for those applications that they have created. For more information on application-level permission, see "Managing Global Security for Applications and Databases" on page 442.

When creating or editing a user in Application Manager, the selection shown in Figure 180 gives the user Create/Delete Applications permission:

Figure 180: Create/Delete Applications Permission



User Type

○ Supervisor

● User

□ Create/Delete Users, Groups

☑ Create/Delete Apps

Granting Application and Database Access to a User or Group

If you need to grant resource-specific permissions to users and groups that are not implied in any user types, you can grant the specific application or database permissions to users when creating or editing them in Application Manager or Administration Services. Using MaxL, you grant the permissions after the user is created by using the **grant** statement.

There is no need to grant permissions to users or groups that are already Supervisors—they have full privileges to all resources on the OLAP Server.

For details about resource-specific permissions that can be granted, see Table 23 on page 424.

To grant or modify application or database permissions for a group, follow the instructions below, substituting the word "group" for "user."

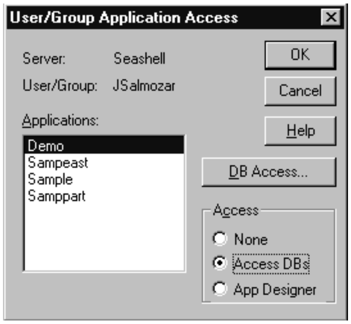
➤ To grant or modify application or database permissions for a user:

1. Using Application Manager, select Security > Users/Groups.
2. Select a user name, then click Edit User.

3. In the Edit User dialog box, click App Access. (This button is disabled if the user being edited is a Supervisor.)

   Essbase displays this dialog box:

   Figure 181: User/Group Application Access Dialog Box



User/Group Application Access

Server: Seashell

User/Group: JSalmozar

Applications:
- Demo
- Sampeast
- Sample
- Samppart

Access
- ○ None
- ● Access DBs
- ○ App Designer

   The Applications list box shows all applications defined on the server to which you have access. When you select an application, the user's current access level for the selected application appears in the Access group.

4. From the Applications list box, select the appropriate application and access option:

   - To give the user no access to the selected application or any of its databases, select None. Note that users can still inherit access if the applications or databases have a higher minimum permission setting than None, or if they are granted a filter that upgrades permission to particular cells.
   - To define specific database access within the selected application, select Access DBs, and then click DB Access.
   - To give the user ownership to the selected application only, select App Designer.

     For information about permissions, see Table on page 15-414.

5. Click OK to save the settings.

Granting Designer Permissions to a User or Group

Users and groups can be granted Application Designer or Database Designer permission for particular applications or databases. These permissions are useful for assigning administrative privileges to users who need to be in charge of particular applications or databases, but who only need ordinary user privileges for other purposes.

For a given database, users or groups can also be granted any of the following permissions: None, Filter Access, Read Only, Read/Write, and Calculate. (See Table on page 15-414.)

You need to grant database access to other users if:

- The users have not been granted sufficient user permission to access databases (see the permissions shown in Table on page 15-414).
- The database in question does not allow users sufficient access through its minimum-permission settings (see "Setting Application and Database Minimum Permissions" on page 448).
- The users do not have sufficient access granted to them through filters (see Chapter 16, "Controlling Access to Database Cells.").

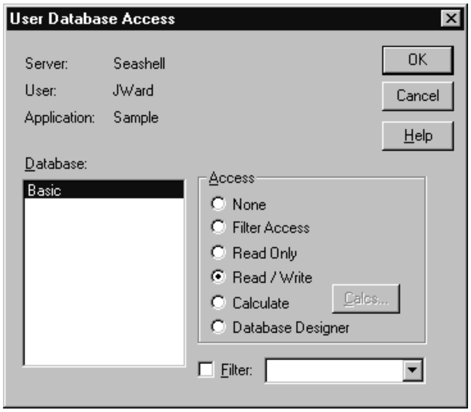
➤ To grant a user or a group access to databases within the selected application, use this procedure:

1. In the Edit User dialog box, click App Access. The User/Group Application Access dialog box is displayed: see Figure 181.
2. In the User/Group Application Access dialog box, select Access DBs. The DB Access button becomes available.

   Note: The DB Access button is disabled when the selected user is a Supervisor or Application Designer for the selected database, because these users already have Database Designer access to every database within the application.

3. Click DB Access. Essbase displays the User Database Access dialog box, as shown in Figure 182.

Figure 182: User Database Access Dialog Box



User Database Access

Server: Seashell

User: JWard

Application: Sample

Database: Basic

Access
○ None
○ Filter Access
○ Read Only
● Read / Write
○ Calculate
○ Database Designer

□ Filter:

The Database list box shows all databases defined within the application to which you have access. After you select a database, the access the user has for the selected database is displayed in the Access group.

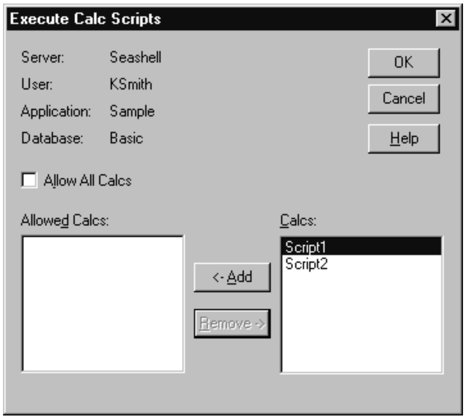
If the user or group is not a Supervisor, you can grant the following permissions, shown in Table 23 on page 424:

4. Select the appropriate database and the permissions you want to grant to the user or group.

If you select the Calculate access level, the Calcs button is enabled. Clicking the Calcs button enables you to grant execution permission to a user for specific calculation scripts or all calculation scripts. Using Application

Manager, Spreadsheet Add-in, or MaxL, users can execute any calculation scripts granted to them. When you click Calcs, Essbase displays the Execute Calc Scripts dialog box, as shown in Figure 183.

Figure 183: Execute Calc Scripts Dialog Box



Server: Seashell
User: KSmith
Application: Sample
Database: Basic

□ Allow All Calcs

Allowed Calcs:

Calcs:
Script1
Script2

When selected, the Allow All Calcs check box gives the user permission to run all calculation scripts stored on the application or database. Any calculation scripts defined afterward are added to the user’s calculate privileges. Individual calculation script permission can be granted or revoked by leaving Allow All Calcs unselected, selecting the name of the script in one of the list boxes, and clicking Add or Remove.

Note: By default, a Supervisor, Application Designer, or Database Designer can run all calculation scripts.

5. Click OK to close the dialog box and save the settings.

Table 23: Permissions Available at the Database Scope

Database permission	**Description**
None	Indicates no access to any object or data value in a database.
Filter Access	Indicates that data access is restricted to those filters assigned to the user. (For information about filters, see Chapter 16, “Controlling Access to Database Cells.”) The Filter check box grants a filter object to a user or group. A user or group can be granted only one filter per database. Selecting this option or any other option except None enables the selection of a filter object from the list box.
Read Only	Indicates read permission: the ability to retrieve all data values. Report scripts can also be run.
Read / Write	Indicates that all data values can be retrieved and updated (but not calculated). The user can run, but cannot modify, Essbase objects.
Calculate	Indicates that all data values can be retrieved, updated, and calculated with the default calculation or any calculation for which the user has been granted permission to execute.
Database Designer	Indicates that all data values can be retrieved, updated, and calculated. In addition, all database-related files can be modified.

Creating, Deleting, and Changing Users and Groups

To help manage security between users and groups, the following user-management tasks are available at varying degrees to users with different permissions:

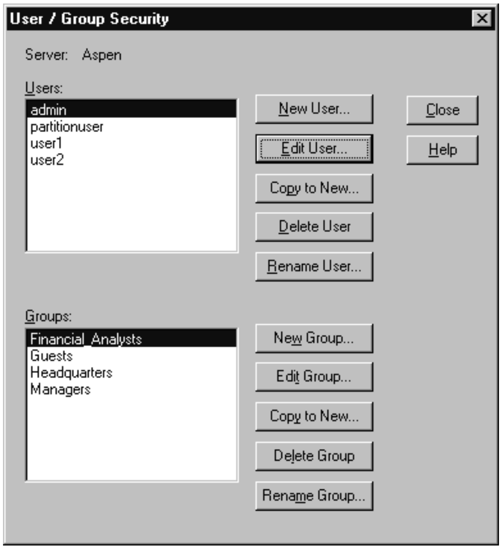
- Create new users and groups
- Edit (modify the permissions of) existing users and groups
- Copy the security profiles of existing users and groups
- Delete users and groups
- Rename users and groups

➤ To manage users and groups using Application Manager:

1. Connect to the server that houses the users or groups.
2. Select Security > Users/Groups.

   Essbase displays the User/Group Security dialog box, as shown in Figure 184:

Figure 184: User/Group Security Dialog Box



User / Group Security

Server: Aspen

Users:
admin
partitionuser
user1
user2

Groups:
Financial_Analysts
Guests
Headquarters
Managers

The Users list box fills with the names of all users currently defined on this server. Similarly, the Groups list box fills with the names of all groups defined on this server. The five buttons to the right of each list box enable you to perform the functions of user and group management.

For more information on managing users and groups, see "Creating, Editing, and Copying Users and Groups" on page 426, "Copying an Existing Security Profile" on page 434, "Deleting Users and Groups" on page 437, or "Renaming Users and Groups" on page 439.

For information about lock management, and password and user name management, see "Managing User Activity at the Server Level" on page 451.

Tip: You can view lists of users and groups using methods other than Application Manager:

Tool	Instructions	For more information
MaxL	**display user** or **display group**	*Technical Reference* in the `docs` directory
ESSCMD	LISTUSERS or LISTGROUPS	*Technical Reference* in the `docs` directory
Administration Services Console	Enterprise View > Security node	*Essbase Administration Services Online Help*

Creating, Editing, and Copying Users and Groups

When you create, edit, or copy a user or a group, you define a security profile. This is where you define the extent of the permissions that users and groups have in dealing with each other and in accessing applications and databases. For more specific data-level security, see Chapter 16, "Controlling Access to Database Cells."

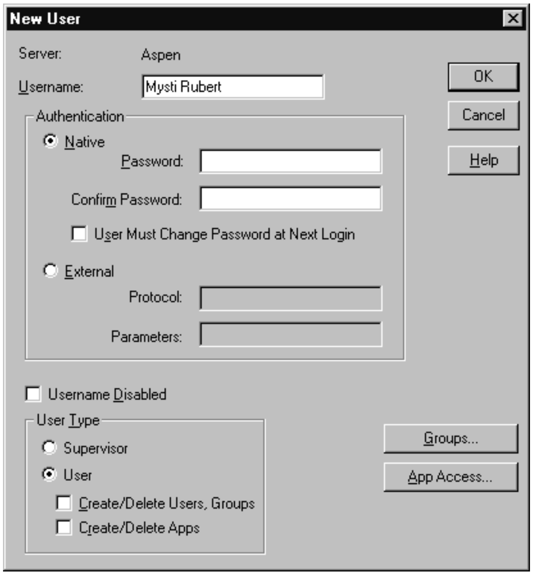
Creating a New User

To create a user means to define the user's name, password, and permission. You can also specify group membership for the user, and you can specify that the user is required to change the password at the next login attempt, or that the user name is disabled, preventing the user from logging in.

➤ To create a new user using Application Manager:

1. Select Security > Users/Groups.
2. In the User/Group Security dialog box, select New User. Essbase displays the New User dialog box, as shown in Figure 185:

Figure 185: New User Dialog Box



New User

Server: Aspen

Username: Mysti Rubert

Authentication

● Native

Password:

Confirm Password:

□ User Must Change Password at Next Login

○ External

Protocol:

Parameters:

□ Username Disabled

User Type

○ Supervisor

● User

□ Create/Delete Users, Groups

□ Create/Delete Apps

3. Type the name of the new user in the Username text box.
4. Select the type of authentication:
   - For most sites, select Native authentication. Essbase authenticates the user with this option.
   - For sites that are using external authentication, select External, and go now to "Managing External Authentication" on page 442.
5. In the Password text box, type the user's password. The password must be at least six characters long.

   As you type, Essbase masks your entry with asterisks.

6. Retype the user's password in the Confirm Password text box. You must type the password exactly as you did in the Password text box.

   Note: Passwords are not case-sensitive.

7. To force the user to change the password at the next login attempt, select User Must Change Password at Next Login. This option enables you to assign a generic password to all new users, which they will have to change when they begin using the system.

   At the next login attempt, the user is prompted to change the password in the Change Password dialog box, shown in Figure 188.

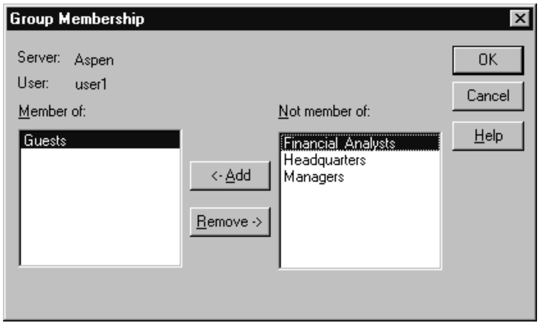
8. To lock the user out from the system, select Username Disabled. Only a system administrator (a user with Supervisor permission) can re-enable the user name.

9. Select the type of user to create from the User Type group. If you are not sure which type of user to create, see "Specifying User and Group Types" on page 417. If you do not have sufficient permission to create a class of user, those options are disabled.

10. To assign the user to a group, click Groups.

    Essbase displays the Group Membership dialog box as shown in Figure 186.

Figure 186: Group Membership Dialog Box



Group Membership

Server: Aspen

User: user1

Member of:

Guests

Not member of:

Financial Analysts
Headquarters
Managers

The Not member of list box contains the names of all groups on the server to which this user does not belong.

a. Select a group from the list and click Add to add the user to that group. Similarly, you can click Remove to remove a user from a group listed in the Member of list box.

b. Click OK to finalize the addition or removal and return to the New User dialog box.

11. To assign application and database access to the user, click App Access from the New User dialog box. See "Granting Application and Database Access to a User or Group" on page 419 for more information.

12. Click OK to add the new user to the server. Essbase updates the Users list box (in the User/Group Security dialog box) with the new user.

Tip: You can create users and add them to groups using methods other than Application Manager:

Tool	**Instructions**	**For more information**
MaxL	**create user** or **alter user**	*Technical Reference* in the `docs` directory
ESSCMD	CREATEUSER, ADDUSER, REMOVEUSER	*Technical Reference* in the `docs` directory
Essbase Administration Services Console	Enterprise View > right-click Users node > add a user.	*Essbase Administration Services Online Help*

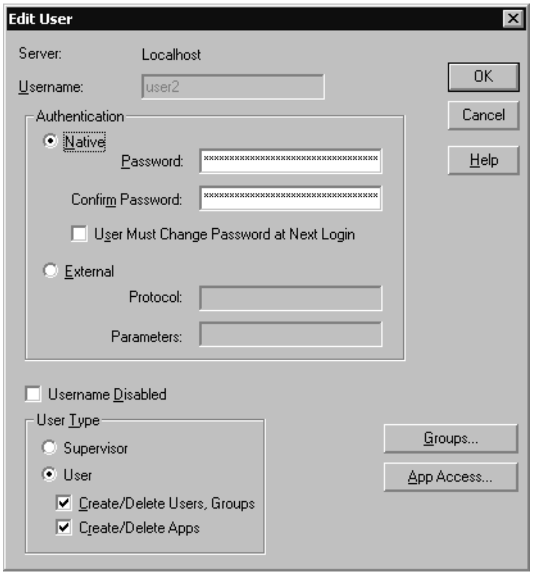
Editing a User

To edit a user means to modify the security profile established when the user was created. The dialog boxes for editing a user and for creating a new one are exactly the same (except for their titles).

1. Select Security > Users/Groups.
2. Select the user whose profile you want to edit and click Edit User in the User/Group Security dialog box.

   Essbase displays the Edit User dialog box as shown in Figure 187.

   Figure 187: Edit User Dialog Box



Edit User

Server: Localhost

Username: user2

Authentication

● Native

Password: ************************************

Confirm Password: ************************************

☐ User Must Change Password at Next Login

○ External

Protocol:

Parameters:

☐ Username Disabled

User Type

○ Supervisor

● User

☑ Create/Delete Users, Groups

☑ Create/Delete Apps

3. To change the user’s password, type the new password in the Password text box. The password must be at least six characters long. As you type, Essbase masks your typing with asterisks.
4. Retype the user’s password in the Confirm Password text box. You must type the password exactly as you did in the Password text box.

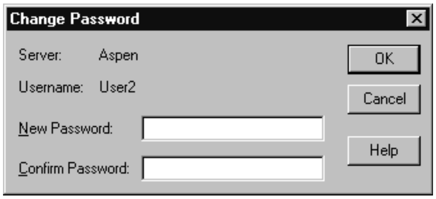
   Note: Passwords are not case-sensitive.

5. To force the user to change the password at the next login attempt, check User Must Change Password at Next Login.

   Note: If you are changing a user for external authentication, see the instructions in "Managing External Authentication" on page 442.

   When this user tries to log in using the old password, he or she will be prompted to first change the password in the Change Password dialog box, as shown in Figure 188.

   Figure 188: Change Password Dialog Box

   Change Password

   Server: Aspen

   Username: User2

   New Password:

   Confirm Password:

6. To prevent the user from logging in, check Username Disabled (in the Edit User dialog box). Only a system administrator (a user with Supervisor permission) can re-enable the username.
7. To change the user's group membership, click Groups.
8. To change the user's application access, click App Access.
9. Click OK to save the user's new security profile on the server. Essbase updates the server security file with the changes.

You cannot change the *names* of users from the Edit User dialog box. Use the Rename User button, described in "Renaming Users and Groups" on page 439.

Tip: You can change a user’s password or other properties using methods other than Application Manager:

Tool	Instructions	For more information
MaxL	**alter user**	*Technical Reference* in the `docs` directory
ESSCMD	SETPASSWORD	*Technical Reference* in the `docs` directory
Administration Services Console	In Enterprise View, right-click a user, and edit the user.	*Essbase Administration Services Online Help*

Creating or Editing a Group

A group is a collection of users who share the same minimum access permissions. Placing users in groups can save you the time of assigning identical permissions to users again and again.

Note: A member of a group may have permissions beyond those assigned to the group, if permissions are also assigned individually to that user.

The process for creating, editing, or copying groups is the same as that for users, except that there are no group passwords. You define group names and permissions just as you would for users.

When you create a new user, you can assign the user to a group. Similarly, when you create a new group, you can assign users to the group. You must define a password for each user; there are no passwords for groups.

➤ To create a new group or edit an existing group using Application Manager:

1. Select Security > Users/Groups.

   Essbase displays the User/Group Security dialog box (see Figure 184 on page 425).

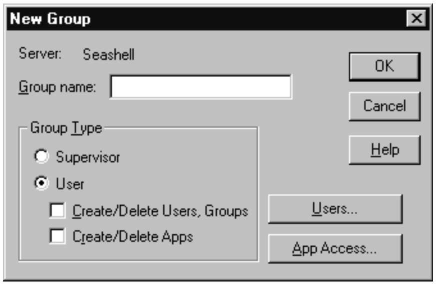
2. To create a new group, click New Group.

   To edit an existing group, select the group you want to edit and click Edit Group. Then follow these instructions; they are the same as for creating a new group.

   Note: You cannot rename a group from the Edit Group dialog box; use the Rename Group button, described in "Renaming Users and Groups" on page 439.

   Essbase displays the New Group dialog box, as shown in Figure 189, or the Edit Group dialog box:

   Figure 189: New Group Dialog Box



Server: Seashell

Group name:

Group Type

○ Supervisor

● User

□ Create/Delete Users, Groups

□ Create/Delete Apps

3. In the Group name text box, type the name of the group.
4. Select the type of group to create from the Group Type group. If you do not have sufficient permission to assign a group to a certain type, those options are not selectable.
5. To assign users to the group:
   - Click Users. The Group Membership dialog box is displayed, with the names of all users on the server listed either as members or non-members.
   - Select which users should be in the group.

     Note: You cannot add users to a group having higher permissions than your own.

   - For more information on using the Group Membership dialog box to assign users to groups, click Help or see the instructions accompanying Figure 186.
   - Click **OK** to assign the users, or click Cancel to assign no users at this time.

6. To assign application and database access to the group, click App Access from the New Group dialog box. Click Help, or see "Granting Application and Database Access to a User or Group" on page 419 for more information.
7. Click OK to add the new group. Essbase updates and displays the Groups list box in the User/Group Security dialog box.

To view a list of users in a group, click Edit Group and then click Users. The Members list box of the User/Group Security dialog box contains a list of the group's users.

Tip: You can create groups and view or change group membership using methods other than Application Manager:

Tool	Instructions	For more information
MaxL	**display user in group** or **create group**.	*Technical Reference* in the `docs` directory
ESSCMD	LISTGROUPUSERS or CREATEGROUP	*Technical Reference* in the `docs` directory
Administration Services Console	In Enterprise View, right-click a user, and edit the user	*Essbase Administration Services Online Help*

Copying an Existing Security Profile

An easy way to create a new user with the same permissions as another user is to copy the security profile of an existing user. The new user is assigned the same user type, group membership, and application/database access as the original user.

You can also create new groups by copying the security profile of an existing group. The new group is assigned the same group type, user membership, and application access as the original group.

Note: "Copy to New" filters any security permissions the creator does not have from the copy. For example, a user with Create/Delete Users permission cannot create a new supervisor by copying the profile of an existing supervisor.

Copying a User Profile

To copy a user or group means to duplicate the security profile of an existing user or group, and to give it a new name. It is helpful to copy users and groups because it saves you the time of reassigning permissions in cases where you want them to be identical.

➤ To create a new user by copying the security profile of an existing user:

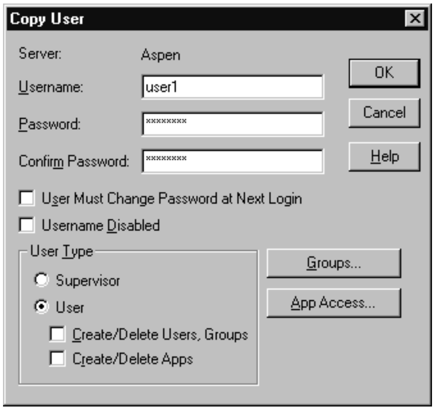
1. Using Application Manager, select Security > Users/Groups.

   Essbase displays the User/Group Security dialog box (see Figure 184).

2. Select the name of the user whose profile you want to copy.

3. Click Copy to New. Essbase displays this dialog box:

   Figure 190: Copy User Dialog Box



Server: Aspen

Username: user1

Password: ********

Confirm Password: ********

□ User Must Change Password at Next Login

□ Username Disabled

User Type

○ Supervisor

● User

□ Create/Delete Users, Groups

□ Create/Delete Apps

4. In the Username text box, type the name of the new user.

5. Enter a password in the Password text box, different from the original user's password if desired. The password must be at least six characters long. As you type, Essbase masks your typing with asterisks.

6. Retype the user's password in the Confirm Password text box. You must type the password exactly as you did in the Password text box.

   Note: Passwords are not case-sensitive.

7. To force the new user to change the password at the next login attempt, select User Must Change Password at Next Login.

8. To prevent the new user from logging in, select Username Disabled. (This option and the preceding option are also available from the New User and Edit User dialog boxes.)

   Only a user with Supervisor permission can reactivate the user name.

9. To change the new user’s system-wide permission, select the new permission from the User Type group. If you do not have sufficient permission to grant a certain permission to the user, that option is not selectable.

10. To change the new user’s group membership, click Groups.

11. To change the new user’s application access, click App Access.

12. Click OK to save the new user’s security profile on the server. Essbase updates the server security file with the changes.

Copying a Group Profile

➤ To create a new group by copying the security profile of an existing group:

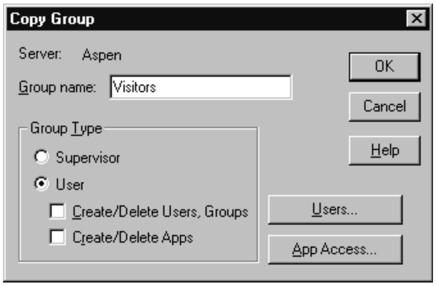
1. Using Application Manager, select Security > Users/Groups.

   Essbase displays the User/Group Security dialog box (see Figure 184 on page 425).

2. Select the name of the group you want to copy.

3. Click Copy to New. Essbase displays this dialog box:

Figure 191: Copy Group Dialog Box



Copy Group

Server: Aspen

Group name: Visitors

Group Type

○ Supervisor

● User

□ Create/Delete Users, Groups

□ Create/Delete Apps

4. Type the name of the new group in the Group name text box.

5. To change the new group's user-type (system-level permission), choose the new type from the Group Type area. If you do not have sufficient permission to assign a certain permission to the group, that option is disabled.

6. To create the new group's membership list, select Users.

   The Group Membership dialog box is displayed.

   For more information on using the Group Membership dialog box to assign users to groups, click Help, or see the instructions accompanying Figure 186.

7. To grant application-wide or database-wide permissions to all members of the group, click App Access.

8. Click OK to save the new group's security profile on the OLAP Server. Essbase updates the server security file with your changes.

Deleting Users and Groups

➤ To delete a user using Application Manager:

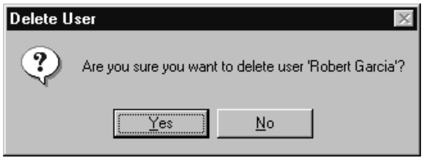
1. Choose Security > Users/Groups.

   Essbase displays the User/Group Security dialog box (see Figure 184 on page 425).

2. Select the name of the user you want to delete in the Users list box.

3. Click Delete User. Essbase displays this confirmation box:

Figure 192: Delete User Confirmation Box



Delete User

Are you sure you want to delete user 'Robert Garcia'?

4. Click Yes to delete the user, or click No to cancel the delete operation.

   If you choose to delete the user, Essbase updates the Users list box and the server security file with your changes. Essbase automatically deletes users from all groups to which they belong.

➤ To delete a group using Application Manager:

1. Select Security > Users/Groups.

   Essbase displays the User/Group Security dialog box (see Figure 184).

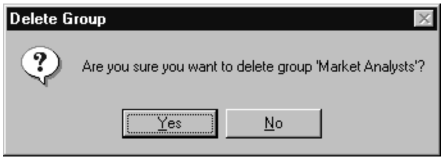
2. In the Groups list box, select the name of the group you want to delete.

3. Click Delete Group.

   Members of the group are not affected by this operation, except that they will no longer be a member of the deleted group.

   When you click Delete Group, Essbase displays the Delete Group confirmation box as shown in Figure 193.

   Figure 193: Delete Group Confirmation Box

   Delete Group

   Are you sure you want to delete group 'Market Analysts'?

4. Click Yes to delete the group, or click No to cancel the delete operation.

   If you choose to delete the group, Essbase updates the Groups list box and the server security file with your changes.

Tip: You can delete users and groups using methods other than Application Manager:

Tool	Instructions	For more information
MaxL	**drop group** or **drop user**	*Technical Reference* in the `docs` directory
ESSCMD	DELETEGROUP or DELETEUSER	*Technical Reference* in the `docs` directory
Administration Services Console	In Enterprise View, right-click the user, and select delete	*Essbase Administration Services Online Help*

Renaming Users and Groups

➤ To rename a user using Application Manager:

1. Select Security > Users/Groups.

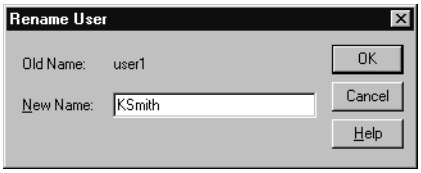
   Essbase displays the User/Group Security dialog box (see Figure 184).

2. Select the name of the user in the Users list box.

3. Click Rename User.

   Essbase displays the Rename User dialog box as shown in Figure 194.

   Figure 194: Rename User Dialog Box



Rename User

Old Name: user1

New Name: KSmith

4. Type the new name for the user in the New Name text box.

5. Click OK to rename the user.

   Essbase updates the Users list box and the server security file with your changes. User names are automatically updated in all groups to which the user belongs.

➤ To rename a group using Application Manager:

1. Select Security > Users/Groups.

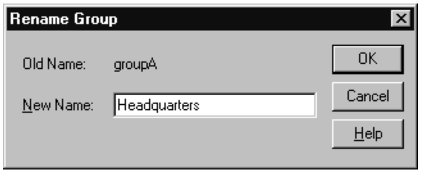
   Essbase displays the User/Group Security dialog box (see Figure 184).

2. Select the name of the group in the Groups list box.

3. Click Rename Group.

   Essbase displays the Rename Group dialog box as shown in Figure 195.

   Figure 195: Rename Group Dialog Box

   Old Name: groupA

   New Name: Headquarters

4. Type the new name for the group in the New Name text box.
5. Click OK to rename the group.

   Essbase updates the Groups list box and the server security file with your changes. Members of the group are not affected by this operation.

Tip: You can rename users and groups using methods other than Application Manager:

Tool	Instructions	For more information
MaxL	**alter group** and **alter user**	*Technical Reference* in the `docs` directory
ESSCMD	RENAMEUSER	*Technical Reference* in the `docs` directory
Administration Services Console	In Enterprise View, right-click the user, and select Rename.	*Essbase Administration Services Online Help*

Viewing and Modifying User Permissions

In Application Manager, you can grant or modify user and group application and database permissions from an edit-user standpoint or from an application or database security perspective. The results are the same.

➤ To grant privileges by editing a user or group, see “Granting Permissions to Users and Groups” on page 416.

➤ To grant user-and-group permissions from an application perspective:

1. From the Application Desktop window, select the application name.
2. Select Security > Application.

   Essbase displays the Application Access dialog box.
3. To view the current permissions for a user or group, select the user or group name from the Users/Groups list box. The Access group shows the current permission for the selected user or group. Click Help for information on each setting.
4. To modify the current application permissions for the selected user or group, select the appropriate permission from the Access group.
5. To define access to databases within the application, select DB Access from the Application Access dialog box.
6. Click OK to save the settings.

➤ To grant user-and-group permissions from a database perspective:

1. From the Application Desktop window, select the application and database name.
2. Select Security > Database. Essbase displays the Database Access dialog box. The Users/Groups list box shows all users and groups on the server. To view the current settings for the user or group, select a name from the list. Click Help for information on each setting.
3. To modify the current permission for a user or group, select the user or group name from the list box and select the permission you want to grant. For information about permissions available to access databases, see Table 23 on page 424.
4. Click OK to save the settings.

Note: If a user has insufficient permission to access the data in a database, the value does not show up in the spreadsheet, or shows up as `#NOACCESS`.

Managing External Authentication

➤ To use external authentication of users instead of assigning an Essbase password for logins, use this procedure:

1. Check the AUTHENTICATIONMODULE configuration setting in your server configuration file `essbase.cfg`, and if you must make any changes, restart OLAP Server.
2. Create a user with either Application Manager or Essbase Administration Services. This procedure shows the Application Manager method:
   a. Follow the instructions in "Creating a New User" on page 426 through Step 5. If you are modifying an existing user, select Security > Users/Groups from the menu, select the appropriate user from the list which appears, and click Edit User.

   b. In the Authentication box, select External.

   c. Enter the protocol, for example LDAP. This should be the same as the authentication module name in the AUTHENTICATIONMODULE entry in your server configuration file `essbase.cfg`.

   d. Enter the authentication parameters, one or more from the AUTHENTICATIONMODULE parameter list in your server configuration file `essbase.cfg`.

   e. Finish creating the new user with the instructions in "Creating a New User" on page 426, beginning with Step 8. If you are modifying an existing user, make any other relevant changes, then click OK.

Note: For an Essbase Administration Services method of creating users with external authentication, see *Essbase Administration Services Online Help*.

Managing Global Security for Applications and Databases

In addition to granting permissions to users and groups, you can change security settings for entire applications and databases and their related files and resources. Application and database security settings enable you to manage connections and create a lowest-common-security profile for the applications and databases.

This section contains the following subsections:

- "Defining Application Settings" on page 443
- "Setting General Application Connection Options" on page 444
- "Setting Application and Database Minimum Permissions" on page 448

Defining Application Settings

You can define permissions and other security settings that apply to applications by changing the application settings. The settings you define for the application affect all users, unless they have higher privileges granted to them at the user level.

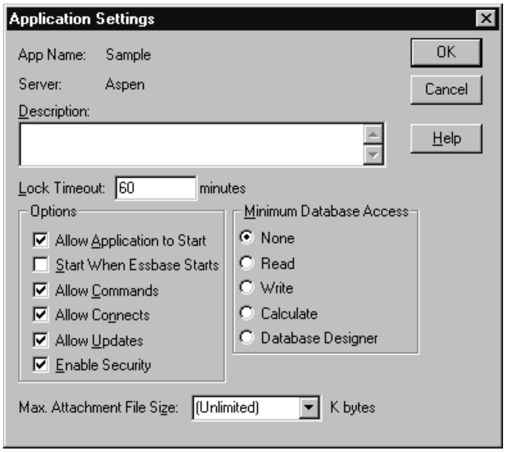
Only users with Supervisor permission (or Application Designer permission for the application) can change application settings.

To define settings for an application:

1. Connect to the appropriate server and select the name of the application you want to secure.
2. Select Application > Settings. (This command is unavailable to users with insufficient privileges to define application settings.)

   Essbase displays the Application Settings dialog box as shown in Figure 196.

Figure 196: Application Settings Dialog Box



App Name: Sample

Server: Aspen

Description:

Lock Timeout: 60 minutes

Options

☑ Allow Application to Start

☐ Start When Essbase Starts

☑ Allow Commands

☑ Allow Connects

☑ Allow Updates

☑ Enable Security

Minimum Database Access

● None

○ Read

○ Write

○ Calculate

○ Database Designer

Max. Attachment File Size: (Unlimited) K bytes

3. Define the settings that you want to apply to the application.

   For information on the setting types, see either "Setting General Application Connection Options" on page 444 or "Setting Application and Database Minimum Permissions" on page 448.

Setting General Application Connection Options

You can define security and connection options using either Application Manager or Administration Services. Select the General tab in the Application Properties window to define the security options using Administration Services.

The following application settings are available in Application Manager:

- A brief description of the application
- A time-out setting for locks
- Options that control application loading, command processing, connections, updates, and security
- Settings that define the minimum permissions to the application. For more information about minimum permissions, see "Setting Application and Database Minimum Permissions" on page 448.

The following settings are available for various levels of application security. For information about how and when disabling of these settings takes effect, see Table 24.

- **Allow Application to Start:** When disabled, prevents all users from starting the application directly or as a result of operations that would start the application; for example, attempting to change application settings or create databases. By default, the application is not prevented from starting.
- **Start When Essbase Starts:** When enabled, the application starts automatically whenever the OLAP Server starts. By default, the application does not start when the OLAP Server starts.

- **Allow commands:** When unchecked, prevents any user from making any requests to databases in the application, including non-data-specific requests such as viewing database information or changing database settings. Supervisors are affected by this setting as a safety mechanism to prevent accidental changes to databases during maintenance operations. By default, commands are enabled.
- **Allow connects:** When unchecked, prevents any user with a permission lower than Application Designer for that application from making connections to databases within the application which would require the databases to be started. This includes explicitly starting the databases, or performing the ESSCMD SELECT command on the databases. By default, connections to databases are allowed.
- **Allow updates:** When unchecked, prevents modification to on-disk database structures; for example, any operation that might have an effect on the data, including updating the data or making outline modifications. Supervisors are affected by this setting as a safety mechanism to prevent accidental updates to databases during maintenance operations. By default, updates are enabled.
- **Enable Security:** When unchecked, Essbase ignores all security settings in the application and treats all users as Application Designers. By default, security is enabled (Essbase permission enforcement is active).

Table 24 describes when the implementation of protective application settings takes effect, how long the effects last, and which users are affected.

Table 24: Scope and Persistence of Application-Protection Settings

Disabled Application Setting	**When the Disabled Setting Takes Effect**	**Which Users are Affected by the Disabled Setting**	**Persistence of the Disabled Setting**
Allow Application to Start	Immediately	All users, including supervisors. Users currently logged in and users who log in later.	The application cannot be started until an administrator re-enables the start-up setting.
Start When Essbase Starts	Immediately	All users.	The application will not start with Essbase unless an administrator enables it.
Allow Commands	Immediately	All users, including supervisors. Users currently logged in and users who log in later.	Commands are disabled until any of the following occurs: 1. The administrator who disabled commands logs out. 2. The application is stopped and restarted. 3. An administrator re-enables commands.
Allow Connects	Immediately, except that disabling connections does not affect users who already have databases loaded.	Users with permissions lower than Application Designer. Users currently logged in and users who log in later. Users already connected to the database are not affected.	Connections are disabled until any of the following occurs: 1. The application is stopped and restarted. 2. An administrator re-enables connections.

Table 24: Scope and Persistence of Application-Protection Settings (Continued)

Disabled Application Setting	When the Disabled Setting Takes Effect	Which Users are Affected by the Disabled Setting	Persistence of the Disabled Setting
Allow Updates	Immediately	All users, including supervisors. Users currently logged in and users who log in later.	Updates are disabled until any of the following occurs: **1.** The administrator who disabled updates logs out. **2.** The application is stopped and restarted. **3.** An administrator re-enables updates.
Enable Security	Immediately	All users, including supervisors. Users currently logged in and users who log in later.	Security is disabled until a user re-enables security.

The application settings can also be accessed in the following interfaces:

- Essbase Administration Console, in the Application Properties dialog box.
- ESSCMD command-line interface: SETAPPSTATE.
- MaxL language interface: **alter application** statement.

Important: If performing maintenance operations that require disabling *commands* or *updates*, make those maintenance operations within the same session as the one in which the setting was disabled.

If you disable commands or updates in a MaxL script, be aware that the end of the script constitutes the end of the session. Calling a nested MaxL or ESSCMD script from the current MaxL script also constitutes the end of the session.

If you disable commands or updates in an ESSCMD script, the end of the script constitutes the end of the session, but calling a nested ESSCMD script from the current ESSCMD script does *not* constitute the end of the session.

CAUTION: *Never* power down or reboot your client computer when you have cleared any of the Allow settings. (Always select Server > Disconnect to log out from the server.) Improper shutdown can cause the application to become inaccessible, which requires a full application shutdown and restart.

If a power failure or system problem causes OLAP Server to improperly disconnect from the Essbase client, and your application is no longer accessible, you must shut down and restart the application. See Chapter 41, "Running Essbase Servers, Applications, and Databases" for more information.

Setting Application and Database Minimum Permissions

Minimum database access permissions can be specified at the application or database level. If specified for an application, minimum database access permissions apply to all databases within the application. When a minimum permission is set to a level higher than None (or No Access) for an application or database, all users on the OLAP Server inherit that permission to access the database or databases.

For example, if an application has Read privilege assigned as the minimum database access level, all users can read any database within that application, even if their individual permissions do not include Read access. Similarly, if a database has a minimum permission setting of None, only users with sufficient granted permissions (granted directly, or implied by filters or group membership) can gain access to the database.

Users with Supervisor, Application Designer, or Database Designer permissions are not affected by minimum-permission settings applied to applications or databases they own. Supervisors have full access to all resources, and Application Designers and Database Designers have full access for their applications or databases.

Users and groups with lower than the minimum permissions inherit at least the minimum permissions for any applications or databases. For information on application and database permissions granted to individual users or groups, see "Granting Application and Database Access to a User or Group" on page 419.

Changes to the minimum permission settings for applications affect only those databases that have lower minimums. In other words, settings defined at a lower level take precedence over more global settings.

The permissions listed in Table 25 are available as minimum settings for applications and databases. Databases of an application inherit the permissions of the applications whenever the application permissions are set higher than those of the database.

Table 25: Minimum Permission Settings Available for Applications and Databases

Permission	Description
None	Specifies that no minimum permission has been defined for the application or database. This is the default global permission for newly created applications and databases.
Read	Specifies Read-Only access to any object or data value in the application or database. Users can view files, retrieve data values, and run report scripts. Read access does not permit data-value updates, calculations, or outline modifications.
Write	Specifies Update access to any data value in the databases of the application, or in one database. Users can view Essbase files, retrieve and update data values, and run report scripts. Write access does not permit calculations or outline modifications.
Calculate	Specifies Calculate and update access to any data value in the databases of the application, or in one database. Users can view files, retrieve, update, and perform calculations based on data values, and run report and calculations scripts. Calculate access does not permit outline modifications.
Designer (for Application or Database)	Specifies Calculate and update access to any data value in the databases of the application, or in one database. In addition, Designer permission enables users to view and modify the outline and files, retrieve, update, and perform calculations based on data values, and run report and calculation scripts.

➤ To set minimum permissions for an application:

1. Connect to the appropriate server and select the name of the application.
2. Select Application > Settings. (This menu command is unavailable to users with insufficient permission to define application settings.)

   Essbase displays the Application Settings dialog box (Figure 196 on page 443).
3. Define the permissions that you want to apply to the application. For information about the permissions, see Table 25 on page 449.

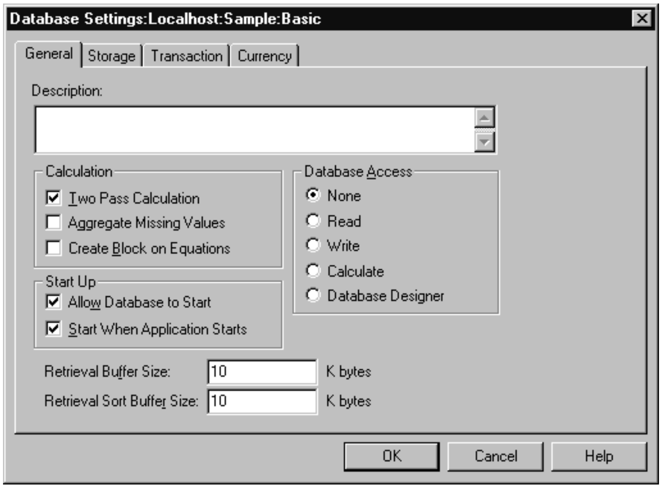
   The minimum permission settings are in the Minimum Database Access group.

➤ To set minimum permissions for a database:

1. Connect to the appropriate server and select the name of the application that contains the database you want to secure.
2. Select the database name.
3. Select Database > Settings. (This menu command is unavailable to users with insufficient privileges to define application settings.)

   Essbase displays the Database Settings dialog box as shown in Figure 197.

Figure 197: Database Settings Dialog Box



Database Settings:Localhost:Sample:Basic

Description:

Calculation

☑ Two Pass Calculation

□ Aggregate Missing Values

□ Create Block on Equations

Start Up

☑ Allow Database to Start

☑ Start When Application Starts

Database Access

● None

○ Read

○ Write

○ Calculate

○ Database Designer

Retrieval Buffer Size: 10 K bytes

Retrieval Sort Buffer Size: 10 K bytes

4. In the Database Access group of the General tab, define the permissions that you want to apply to the database. For information about the permissions, see Table 25 on page 449.

   The minimum permission settings are in the Database Access group.

   Note: Although any user with a minimum of Read access to a database can start the database, only a Supervisor, a user with Application Designer privilege for the application, or a user with Database Designer privilege for the database can stop the database.

Managing User Activity at the Server Level

This section explains how to manage the activities of users connected to the server. The security concepts explained in this section are session and request management, lock management, connection management, and password/user name management. For information about managing security for partitioned databases, see Chapter 13, “Designing Partitioned Applications.”

Disconnecting Users and Terminating Requests

The security system lets you disconnect a user from the Essbase server when you want to perform maintenance tasks.

To view sessions, disconnect sessions, or terminate requests, you must have Supervisor permission or Application Designer permission for the specified application. You can view or terminate only sessions or requests for users with permissions equal to or lesser than your own.

A *session* is the time between login and logout for a user connected to Essbase OLAP Server at the system, application, or database scope. A user can have more than one session open at any given time. For example, a user may be logged on to different databases. If you have the appropriate permissions, you can log off sessions based on any criteria you choose; for example, an administrator can log off a user from all databases or from a particular database.

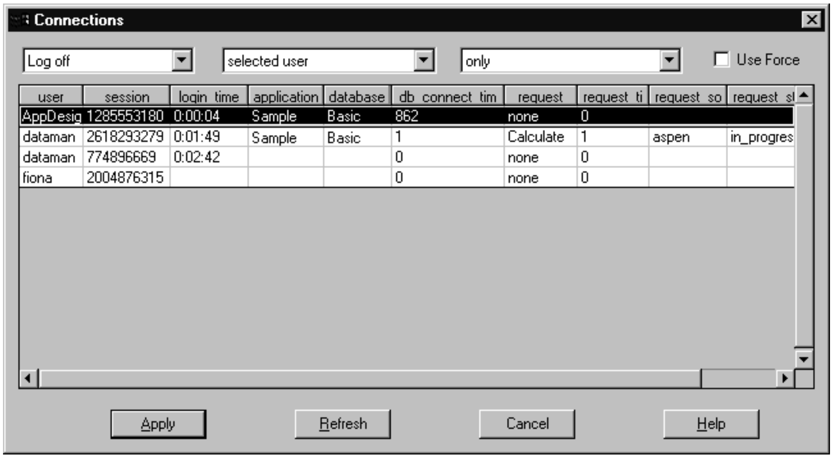
A *request* is a query sent to OLAP Server by a user or by another process; for example, a default calculation of a database, or a restructuring of the database outline. Each session can process only one request at a time.

➤ To disconnect a session or request:

1. In Application Manager, choose Security > Connections.

   Essbase displays the Connections dialog box as shown in Figure 198.

   Figure 198: Connections Dialog Box



user	session	login time	application	database	db connect tim	request	request ti	request so	request st
AppDesig	1285553180	0:00:04	Sample	Basic	862	none	0		
dataman	2618293279	0:01:49	Sample	Basic	1	Calculate	1	aspen	in_progres
dataman	774896669	0:02:42			0	none	0		
fiona	2004876315				0	none	0		

   If you have Supervisor permission, this dialog box lists the following:

   - All users connected to the same server to which you are connected.
   - A session ID number for each session.
   - The login time for each session, in hours:minutes:seconds.
   - The application and database each user is using. If no application and database names are displayed, then the user is logged in but not connected to a specific application.
   - The time connected to a displayed database for each relevant session, in seconds.
   - The type of request being performed in each session, if any.
   - The length of time each request has been processing in hours:minutes:seconds.
   - The name of the computer from which each request is being made.
   - The status of each request: whether it is processing, terminating, or terminated.

Unless you are a Supervisor, you can see only session information for users who are connected to an application for which you have Application Designer permission.

If you have Application Designer permission, this dialog box lists information only for users (including yourself) who are connected to any application for which you have Application Designer permission.

2. To disconnect users or terminate requests, use the drop-down list boxes and optionally select a user session from the grid. (You do not need to select a session from the grid if you are going to terminate multiple sessions at once.)

   Table 26 on page 454 lists the selection combinations available by selecting options from the list boxes, and their equivalent MaxL statements. The **log off** options are for terminating a user's session. The **kill** options are for terminating specific requests within any session without logging out the user's entire session.

   Though not listed below, the Use Force check box is also available when a **log off** option is selected. This check box is useful when you want to terminate a session or sessions while at least one of those sessions is running a request that is currently processing or is not responding. The MaxL equivalent to Use Force is to include the "force" keyword after any statement for logging off a user; for example, **alter system logout session <session-id> force;**.

3. Click Apply to execute the operation indicated by your list-box selections. Click Refresh to update your view to see the latest session and request activity on the OLAP Server. Click Cancel to exit the Connections dialog box without terminating any sessions or requests.

Note: You cannot terminate a restructure process. If you attempt to terminate it, a "command not accepted" error is returned, and the restructure process is not terminated.

Table 26: Session and Request Termination Options

Selections in Application Manager			MaxL equivalents
log off	selected user	only	`alter system logout session <session-id>;`
log off	all users	on selected server	`alter system logout session all;`
log off	all users	on selected application	`alter system logout session on application <app-name>;`
log off	all users	on selected database	`alter system logout session on database <dbs-name>;`
log off	all instances of user	on selected server	`alter system logout session by user <user-name>;`
log off	all instances of user	on selected application	`alter system logout session by user <user-name> on application <app-name>;`
log off	all instances of user	on selected database	`alter system logout session on database <dbs-name>;`
kill	selected request	only	`alter system kill request <session-id>;`
kill	all requests	on selected server	`alter system kill request all;`
kill	all requests	on selected application	`alter system kill request on application <app-name>;`
kill	all requests	on selected database	`alter system kill request on database <dbs-name>;`
kill	all requests from user	on selected server	`alter system kill request by user <user-name>;`
kill	all requests from user	on selected application	`alter system kill request by user <user-name> on application <app-name>;`
kill	all requests from user	on selected database	`alter system kill request by user <user-name> on database <dbs-name>;`

Managing User Locks

Essbase Spreadsheet Add-in users can interactively send data from a spreadsheet to the server. To maintain data integrity while providing multiple-user concurrent access, Essbase enables users to lock data for the purpose of updating it. Users who want to update data must first lock the records to prevent other users from trying to change the same data.

Occasionally, you may need to force an unlock operation. For example, if you attempt to calculate a database that has active locks, the calculation must wait when it encounters a lock. By clearing the locks, you allow the calculation to resume.

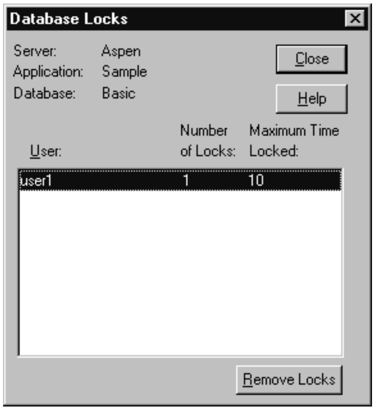
Only Supervisors can view users holding locks and remove their locks.

➤ To view or remove locks:

1. Select Security > Locks.

   Essbase displays the Database Locks dialog box as shown in Figure 199.

   Figure 199: Database Locks Dialog Box



Database Locks

Server: Aspen
Application: Sample
Database: Basic

User:	Number of Locks:	Maximum Time Locked:
user1	1	10

   The Database Locks dialog box displays a list of users who currently have at least one block locked. It also indicates the number of blocks that are locked, and the amount of time, in seconds, that the blocks have been locked.

2. To remove a lock, select the user name and click Remove Locks. Removing a user’s lock forces a logout of that user’s session.

   You can also use the REMOVELOCKS command in ESSCMD to perform this task. See the *Technical Reference* in the `docs` directory for information.

Managing Passwords and User Names

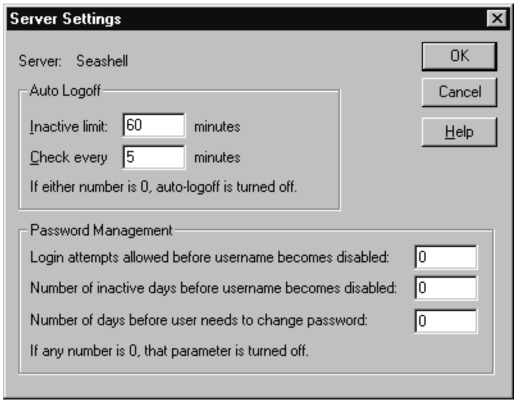
You can place limitations on the number of login attempts users are allowed, on the number of days users may not use Essbase before becoming disabled from the server, and on the number of days users are allowed to have the same passwords. Only system administrators (users with Supervisor privilege) can access these settings. The limitations apply to all users on the server, and are effective immediately upon clicking OK.

➤ To place limitations on users:

1. Select Server > Settings.

   Essbase displays the Server Settings dialog box as shown in Figure 200.

   Figure 200: Server Settings Dialog Box



Server Settings

Server: Seashell

Auto Logoff

Inactive limit: 60 minutes

Check every 5 minutes

If either number is 0, auto-logoff is turned off.

Password Management

Login attempts allowed before username becomes disabled: 0

Number of inactive days before username becomes disabled: 0

Number of days before user needs to change password: 0

If any number is 0, that parameter is turned off.

   The Password Management option group contains the settings for user management. A setting of 0 for any option means that parameter is turned off; therefore, you must enter at least 1 to apply limitations.

2. To limit the number of unsuccessful login attempts you want to allow before the user becomes locked out from the server, enter the maximum number of attempts to allow in the first text box of the Password Management group.

   Note: If you return to the Server Settings dialog box later and change the number of unsuccessful login attempts allowed, Essbase resets the count for all users. For example, if the setting was 15 and you changed it to 20, all users would be allowed 20 *new* attempts. If you changed the setting to 2, a user who had already exceeded that number when the setting was 15 would *not* be locked out. The count returns to 0 for each change in settings.

3. To limit the number of inactive days allowed before the user becomes locked out from the server, enter the number of days in the second text box of the Password Management group.

   The timer starts for all users as soon as you click OK, and it is reset for particular users each time they log in or are reactivated or edited by Supervisors.

4. To limit the number of days any user can log in with the same password, enter the number of days in the third text box.

   The timer starts for all users as soon as you click OK, and it is reset for particular users each time they change their passwords or are reactivated or edited by Supervisors.

Viewing or Activating Disabled User Names

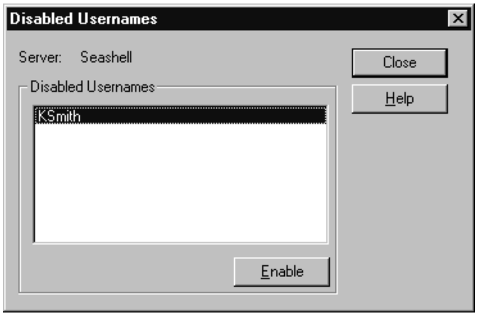
A user name becomes disabled when the user exceeds limitations specified in the Server Settings dialog box (see "Managing Passwords and User Names" on page 456), or because a system administrator has disabled the user name at the user level. To learn how to disable a user name, see "Editing a User" on page 430.

➤ To view or activate currently disabled user names:

1. Select Security > Disabled Usernames.

   Essbase displays the Disabled Usernames dialog box, shown in Figure 201, which lists all disabled user names:

Figure 201: Disabled Usernames Dialog Box



Disabled Usernames

Server: Seashell

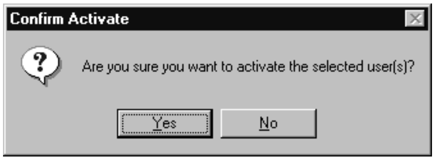
Disabled Usernames

KSmith

2. To activate a user, select the user name from the list box and click Enable.

   Essbase displays the Confirm Activate confirmation box as shown in Figure 202.

   Figure 202: Confirm Activate Confirmation Box

   Confirm Activate

   Are you sure you want to activate the selected user(s)?

3. Click Yes to confirm that you want to activate the selected user name, or click No to leave it disabled.

Note: Only a system administrator (a user with Supervisor permission) can view or reactivate disabled user names.

The ESSBASE.SEC Security File

All information about users, groups, passwords, privileges, filters, applications, databases, and their corresponding directories is stored in the `ESSBASE.SEC` file in your `$ARBORPATH\Bin` directory. Each time you successfully start the Agent, a backup copy of the security file is created as `essbase.bak`.

If you attempt to start the Agent and cannot get a password prompt or your password is rejected, no `.bak` file is created. You can restore from the last successful startup by copying `essbase.bak` to `essbase.sec`. Both files are in the `bin` directory where you installed OLAP Server.

Chapter

16

Controlling Access to Database Cells

When the security levels defined for applications, databases, users, and groups are not enough, Essbase security filters give you control over security at the most detailed level. Filters let you control access to individual data within a database, by defining what kind of access is allowed to which parts of the database, and to whom these settings apply.

If you have the role of Supervisor, you can define and assign any filters to any users or groups. Filters do not affect you.

If you are a user with Create/Delete Applications privilege, you can assign and define filters for applications you created.

If you have the role of Application Designer or Database Designer, you can define and assign filters within your applications or databases. For more information about privilege levels, see Chapter 15, “Managing Security for Users and Applications.”

This chapter contains the following sections:

Permissions Defined by Filters

Filters control security access to data values, or cells. You create filters to accommodate security needs for specific parts of a database. When you define a filter, you are designating a set of restrictions upon particular database cells. When you save the filter, you give it a unique name to distinguish it from other filters, and the server stores it in `ESSBASE.SEC`, the Essbase security file. You can then assign the filters to any users or groups on the server.

For example, a manager designs a filter named RED, and associates it with a particular database to limit access to cells containing profit information. The filter is assigned to a visiting group called REVIEWERS, so that they can read, but cannot alter, most of the database, while they have no access at all to Profit data values.

Filters are composed of one or more access settings for database members. You can specify the following access levels and apply them to data ranging from a list of members to an individual cell.

Access Level	Description
None	No data can be retrieved or updated for the specified member list.
Read	Data can be retrieved but not updated for the specified member list.
Write	Data can be retrieved and updated for the specified member list.

Any cells that are not specified in the filter definition inherit the database access level. Filters can, however, add or remove access assigned at the database level. This is because the filter definition, being more data-specific, indicates a greater level of detail than the more general database access level.

Note: Data values not covered by filter definitions default to the access levels defined for users, and secondly to the global database access levels. For more about global and user security, see Chapter 15, “Managing Security for Users and Applications.”

Calculation access is controlled by minimum global permissions or by permissions granted to users and groups. Users who have calculate access to the database are not blocked by filters: they can affect all data elements that the execution of their calculations would update.

Defining Filters

To define a filter means to do any of the following things:

- Create a new filter
- Edit an existing filter
- Copy an existing filter into a new but identical filter
- Rename an existing filter
- Delete an existing filter

Before defining a filter, you must connect to the server and select the database associated with the filter.

To define a filter for the selected database, choose Security > Filters. Essbase displays the Filters dialog box. Begin with this dialog box for all filter definitions, whether you are creating, deleting, editing, copying, or renaming a filter.

If you want only to view a list of filters for the selected database, this dialog box shows a list of filters.

Tip: You can view lists of filters using methods other than Application Manager:

Tool	Instructions	For more information
MaxL	**display filter**	*Technical Reference* in the `docs` directory
ESSCMD	LISTFILTERS	*Technical Reference* in the `docs` directory

Creating a New Filter

You can create a new filter for each set of access restrictions you need to place on database values. There is no need to create separate filters for users with the same access needs—once you have created a filter, you can assign it to multiple users or groups of users. However, only one filter per database can be assigned to a user or group.

Note: If you use a calcluation function that returns a set of members, such as children or descendants, and it evaluates to an empty set, the security filter is not created. An error is written to the application log stating that the region definition evaluated to an empty set .

➤ To create a filter:

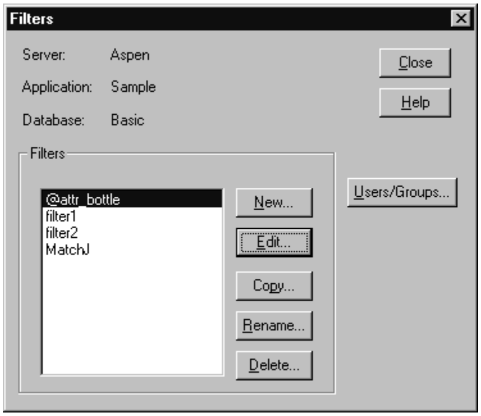
1. Select your current application and database in the Application Desktop window (if they are not already selected).

   To practice creating a filter using a sample, see “Mini-Tutorial” on page 464.

2. Choose Security > Filters.

   Essbase displays the Filters dialog box.

   Figure 203: Filters Dialog Box



Filters

Server: Aspen

Application: Sample

Database: Basic

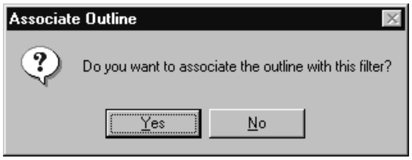
Filters

@attr_bottle
filter1
filter2
MatchJ

3. Click New.

   Essbase displays this confirmation box.

   Figure 204: Associate Outline Confirmation Box



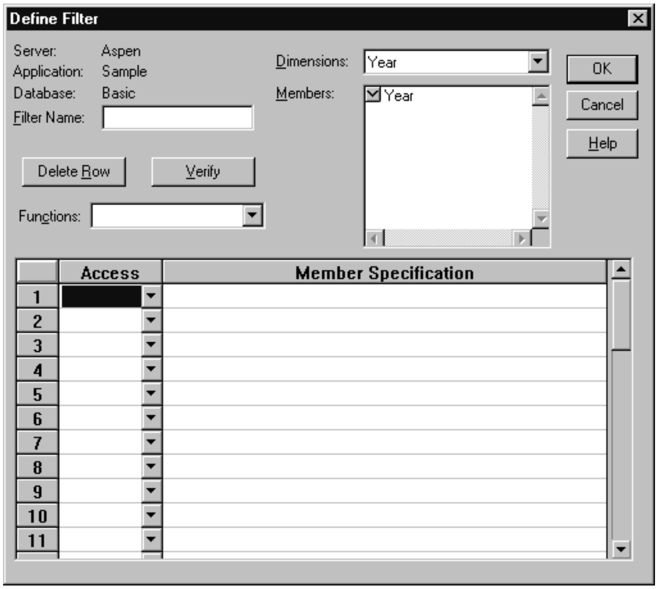
Associate Outline

Do you want to associate the outline with this filter?

4. Click Yes to confirm that you want to associate the current outline with the filter. This will enable the member selection tool so that you do not have to type in all member specifications.

   Essbase displays the Define Filter dialog box.

   Figure 205: Define Filter Dialog Box



Define Filter

Server: Aspen
Application: Sample
Database: Basic
Filter Name:
Dimensions: Year
Members: ☑ Year

   Click Help for information on each option.

5. Type a name for the new filter in the Filter Name text box.

6. To define an access level for whatever object you intend to specify in the corresponding row of the Member Specification column:

   a. Make sure that Row 1 under the Access column is selected, as in Figure 205.

   b. Click the down arrow next to the cell in the Access column to choose an access level of None, Read, or Write.

7. To select a dimension to which you want to specify the access levels:

   a. Choose a dimension from the Dimensions list box.

   b. In the Members list box, double-click on the down-arrow next to a dimension name to see it expand into an outline of its members.

c. Make sure the cursor is in the first row of the filter sheet, labeled Member Specification.

d. Paste a dimension name or member name into the row by selecting the word from the outline in the Members list box.

Note: You do not have to use the member selection tool to specify dimensions and members. If you know the names of members to filter, you can type them in. However, be sure to use quotation marks ("") around any member names containing special characters or spaces. For more information, see "Rules for Naming Dimensions and Members" on page 179.

8. To apply a function to dimensions and members:

   a. Position the cursor in the Member Specification row and select the appropriate function from the Functions list box.

   b. Specify a member for the function by placing the cursor within the parentheses and selecting the member name from the outline in the Members list box.

   The member name should appear within the parentheses.

9. To delete a row, place the insertion point in the row and click Delete Row.

10. To verify that your syntax is correct for the entire filter sheet, click Verify.

Note: For more information about functions and syntax, consult the *Technical Reference* in the `docs` directory.

11. Click OK to save the filter.

Mini-Tutorial

Use this procedure to create a filter for Sample Basic:

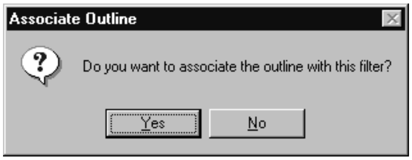
1. Make sure your current application is Sample and your current database is Basic. (Specify these in the Application Desktop window.)

2. Choose Security > Filters. Essbase displays the Filters dialog box (Figure 203).

3. Click New.

   Essbase displays this confirmation box.

   Figure 206: Associate Outline Confirmation Box

   Associate Outline

   Do you want to associate the outline with this filter?

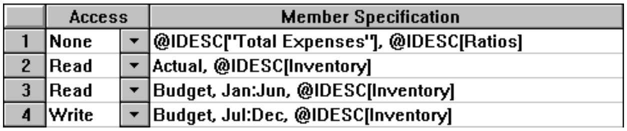
4. Click Yes to confirm that you want to associate the current outline with the filter.

   Essbase displays the Define Filter dialog box (see Figure 205).

5. Type the name Finances in the Filter Name text box.

6. Fill in the filter sheet for the sample filter, Finances, to match this example:

   Figure 207: Sample Member Specifications in the Define Filter Dialog Box

	Access	Member Specification
1	None	@IDESC["Total Expenses"], @IDESC[Ratios]
2	Read	Actual, @IDESC[Inventory]
3	Read	Budget, Jan:Jun, @IDESC[Inventory]
4	Write	Budget, Jul:Dec, @IDESC[Inventory]

   See the instructions for creating a filter (see “Creating a New Filter” on page 461).

   This filter defines the following access plan:

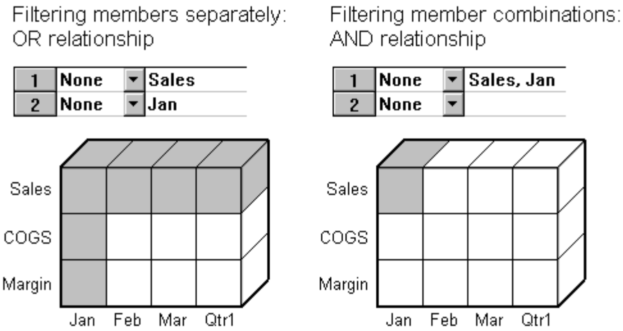
   - No retrieval or update access to any data for members of the Total Expenses branch or the Ratios branch
   - Read-only access to all members of the Inventory branch below Actual
   - Read-only access to all members of the Inventory branch below Budget for the months of Jan through Jun
   - Write access to all members of the Inventory branch below Budget data for months of Jul through Dec

Filtering Whole Members vs. Filtering Member Combinations

The examples of Figure 208 illustrate different ways to control access to database cells. Data can be protected by filtering entire members, or by filtering member combinations.

Filtering members separately, by defining access for each member in a separate row of the Define Filter dialog box, affects whole regions of data for those members. Filtering member combinations, using one row in the Define Filter dialog box, affects data at the member intersections.

Figure 208: How Filters Affect Data: AND/OR Relationships



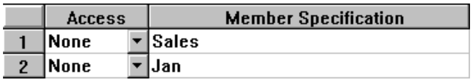
Filtering Members Separately

To filter all the data for one or more members, define access for each member on its own row in the Define Filter dialog box. Filter definitions on separate rows of a filter are treated with an OR relationship.

Example:

User KSmith is assigned this filter:

Figure 209: Filter Blocking Access to Sales or Jan

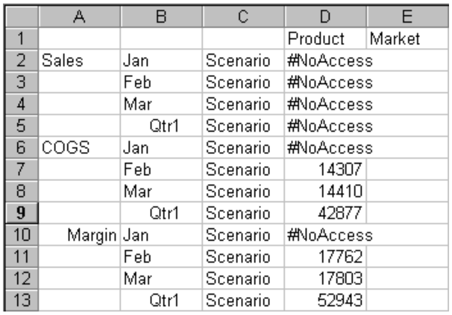


	Access	Member Specification
1	None	Sales
2	None	Jan

The filter blocks access to all members Sales or Jan in the Sample Basic database.

The next time user KSmith connects to Sample Basic, she has no access to data values for the member Sales or for the member Jan. Her spreadsheet view of the profit margin for Qtr1 looks like this view:

Figure 210: Results of Filter Blocking Access to Sales or Jan



	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	

All data for Sales is blocked from view, as well as all data for January, inside and outside of the Sales member. Data for COGS (Cost of Goods Sold), a sibling of Sales and a child of Margin, is available, with the exception of COGS for January.

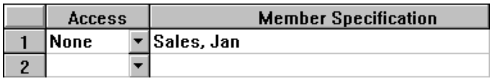
Filtering Member Combinations

To filter data for member combinations, define the access for each member combination using a single row in the Define Filter dialog box. A filter definition using one row and a comma is treated as an AND relationship.

Example:

User RChinn is assigned this filter:

Figure 211: Filter Blocking Access to Sales for Jan

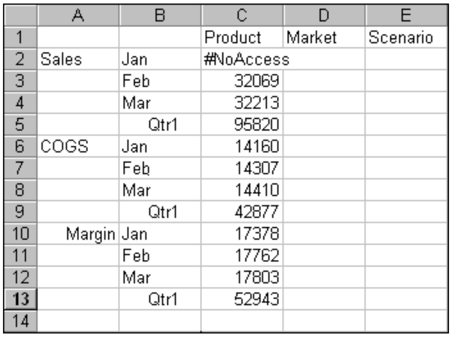


	Access	Member Specification
1	None	Sales, Jan
2		

The filter blocks only the intersection of the members Sales and Jan in the Sample Basic database.

The next time user RChinn connects to Sample Basic, she has no access to the data value at the intersection of members Sales and Jan. Her spreadsheet view of the profit margin for Qtr1 looks like this view:

Figure 212: Results of Filter Blocking Access to Sales, Jan



	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		
14					

Sales data for January is blocked from view. However, Sales data for other months is available, and non-Sales data for January is available.

Filtering with Attribute Functions

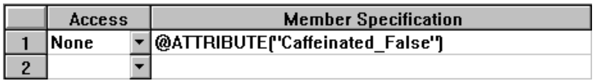
You can use filters to restrict access to data for base members sharing a particular attribute. To filter data for members with particular attributes defined in an attribute dimension, use the attribute member in combination with the @ATTRIBUTE function or the @WITHATTR function.

Note: @ATTRIBUTE and @WITHATTR are member set functions. Most of the member set functions can be used in filter definitions. For more information about functions, see the *Technical Reference* in the `docs` directory.

Example

User PJones is assigned this filter:

Figure 213: Filter Blocking Access to Members with Attribute "Caffeinated_False"

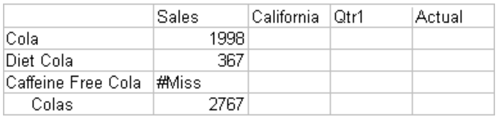


	Access	Member Specification
1	None	@ATTRIBUTE("Caffeinated_False")
2		

The filter blocks access to data for caffeine-free products. “Caffeinated_False” is a Boolean-type attribute member in Sample Basic, in the Pkg Type attribute dimension. This attribute is associated with members in the base dimension Product.

The next time user PJones connects to Sample Basic, he has no access to the data values for any base dimension members associated with Caffeinated_False. His spreadsheet view of first-quarter cola sales in California looks like this view:

Figure 214: Results of Filter Blocking Access to Caffeine-free Products



	Sales	California	Qtr1	Actual
Cola	1998			
Diet Cola	367			
Caffeine Free Cola	#Miss			
Colas	2767			

Sales data for Caffeine Free Cola is blocked from view. Note that Caffeine Free Cola is a base member, and Caffeinated_False is an associated member of the attribute dimension Caffeinated (not shown in the above spreadsheet view).

Editing a Filter

➤ To edit an existing filter:

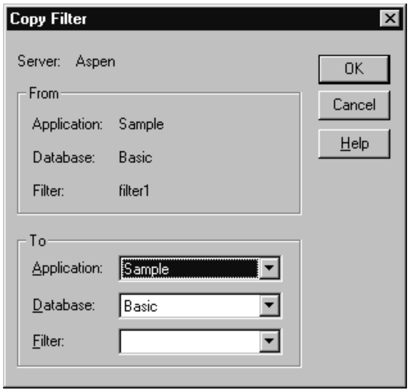
1. Choose Security > Filters. Essbase displays the Filters dialog box (see Figure 203 on page 462).
2. Select the filter you want to edit and click Edit. Essbase displays the Associate Outline Confirmation box, as in Figure 204 on page 462.
3. Click Yes to confirm that you want to associate the current outline with the filter. Essbase displays the Define Filter dialog box, as in Figure 205 on page 463.
4. Edit the filter as you would create one, by filling in the filter definition rows from items selected in the list boxes. See “Creating a New Filter” on page 461.
5. Click OK to save the filter.

Copying a Filter

➤ To copy an existing filter:

1. Choose Security > Filters. The Filters dialog box appears (see Figure 203 on page 462).
2. Select the filter you want to copy and click Copy. Essbase displays the Copy Filter dialog box, as shown in Figure 215.
   - The labels in the From group display the name and location of the original filter.
   - The To group is where you enter the information for the new filter.

Figure 215: Copy Filter Dialog Box



Copy Filter

Server: Aspen

From

Application: Sample

Database: Basic

Filter: filter1

To

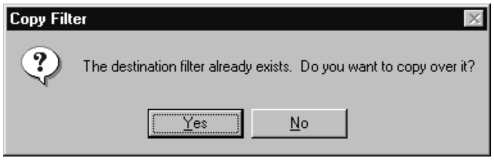
Application: Sample

Database: Basic

Filter:

3. Select the application and database for the new filter from the list boxes in the To group.
4. Type the name of the new filter in the Filter text box, or if you decide not to create a new filter, but would rather update an existing one so that it becomes a copy of the original, select an existing filter name from the list box. Essbase displays this confirmation box:

Figure 216: Copy Filter Confirmation Box



Copy Filter

The destination filter already exists. Do you want to copy over it?

5. Click Yes to confirm that you want to replace the existing filter with the copy.
6. Click OK to save the filter.

Tip: You can create or copy filters using methods other than Application Manager.

Tool	Instructions	For more information
MaxL	**create filter** or **create filter as statements**	*Technical Reference* in the `docs` directory
ESSCMD	COPYFILTER	*Technical Reference* in the `docs` directory

Renaming a Filter

➤ To rename a filter:

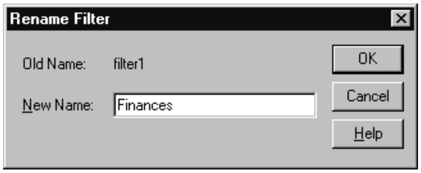
1. Choose Security > Filters.

   Essbase displays the Filters dialog box (see Figure 203 on page 462).

2. Select the filter you want to rename and click Rename.

   Essbase displays this dialog box:

   Figure 217: Rename Filter Dialog Box



Rename Filter

Old Name: filter1

New Name: Finances

3. Type in the new name and click OK to save.

Tip: You can also use the RENAMEFILTER command in ESSCMD to perform this task. See the *Technical Reference* in the `docs` directory for information.

Deleting a Filter

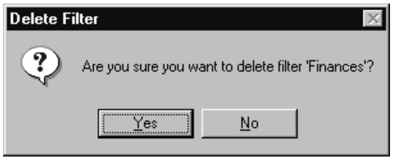
➤ To delete a filter:

1. Choose Security > Filters.

   Essbase displays the Filters dialog box (see Figure 203 on page 462).

2. Select the filter you want to delete and click Delete. Essbase displays this confirmation box:

   Figure 218: Delete Filter Confirmation Box

   Delete Filter

   Are you sure you want to delete filter 'Finances'?

3. Click Yes to confirm that you want to delete the filter named in the confirmation box.

Assigning Filters

Once you have defined filters, you can assign them to users or groups. This lets you manage multiple users who require the same filter settings. Modifications to a filter's definition are automatically inherited by users of that filter.

Filters do not affect users who have the role of Supervisor. Only one filter per database can be assigned to a user or group.

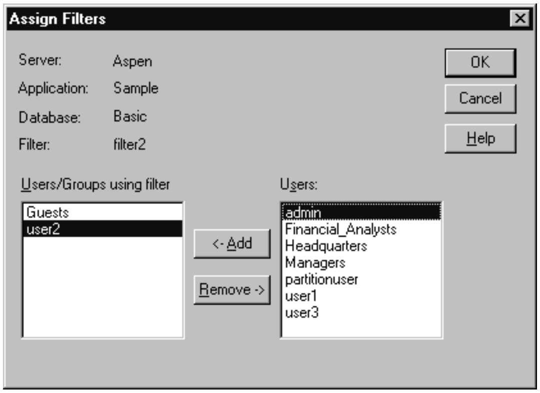
➤ To assign a filter to a user or group:

1. Choose Security > Filters.

   The Filters dialog box appears as in Figure 203 on page 462.

2. Select the filter name you want to assign and click Users/Groups. Essbase displays this dialog box:

Figure 219: Assign Filters Dialog Box



Assign Filters

Server: Aspen

Application: Sample

Database: Basic

Filter: filter2

Users/Groups using filter: Guests, user2

Users: admin, Financial_Analysts, Headquarters, Managers, partitionuser, user1, user3

3. Select a name from the Users list box and click Add. Similarly, you can remove a user or group by selecting the name from the Users/Groups using filter list box and clicking Remove.

4. Click OK.

Overlapping Filter Definitions

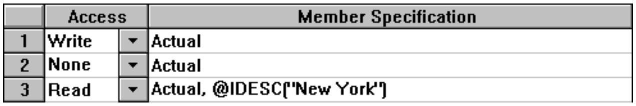
If a filter contains rows that have overlapping member specifications, the inherited access is set by the following rules, which are listed in order of precedence:

1. A filter that defines a more detailed dimension combination list takes precedence over a filter with less detail.

2. If the preceding rule does not resolve the overlap conflict, the highest access level among overlapping filter rows is applied.

For example, this filter contains overlap conflicts:

Figure 220: Filter with Overlap Conflicts



	Access	Member Specification
1	Write	Actual
2	None	Actual
3	Read	Actual, @IDESC("New York")

The third specification defines security at a greater level of detail than the other two. Therefore Read access is granted to all Actual data for members in the New York branch.

Because Write access is a higher access level than None, the remaining data values in Actual are granted Write access.

All other data points, such as Budget, are accessible according to the minimum database permissions.

Note: If you have Write access, you also have Read access.

Changes to members in the database outline are not reflected automatically in filters. You must manually update member references that change.

Overlapping Access Definitions

When the access rights of user and group definitions overlap, the following rules, listed in order of precedence, apply:

1. An access level that defines a more detailed dimension combination list takes precedence over a level with less detail.
2. If the preceding rule does not resolve the overlap conflict, the highest access level is applied.

   Example 1:

   User Fred is defined with the following database access:

   ```
   FINPLAN     R
   CAPPLAN     W
   PRODPLAN    N
   ```

   He is assigned to Group Marketing which has the following database access:

   ```
   FINPLAN     N
   CAPPLAN     N
   PRODPLAN    W
   ```

   His effective rights become:

   ```
   FINPLAN     R
   CAPPLAN     W
   PRODPLAN    W
   ```

Example 2:

User Mary is defined with the following database access:

```
FINPLAN     R
PRODPLAN    N
```

She is assigned to Group Marketing which has the following database access:

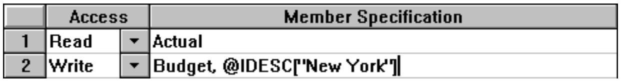
```
FINPLAN     N
PRODPLAN    W
```

Her effective rights become:

```
FINPLAN     W
PRODPLAN    N
```

In addition, Mary uses the filter object RED (for the database FINPLAN). The filter has two filter rows:

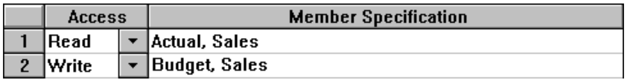
Figure 221: RED Filter for Database FINPLAN



	Access	Member Specification
1	Read ▾	Actual
2	Write ▾	Budget, @IDESC("New York")

The Group Marketing also uses a filter object BLUE (for the database FINPLAN). The filter has two filter rows:

Figure 222: BLUE Filter for Database FINPLAN



	Access	Member Specification
1	Read ▾	Actual, Sales
2	Write ▾	Budget, Sales

The effective rights from the overlapping filters are:

R	Actual
W	For all Budget data in the New York branch
W	For data values that relate to Budget and Sales

The access level for unspecified members is the inherited access level of the database (in this case, Read).

For more sample scenarios, see Chapter 17, “Security Examples.”

Chapter 17

Security Examples

This chapter describes some sample security problems and solutions, which are based on the Sample application on the OLAP Server. These examples use security procedures for which the basic instructions are found in Chapter 15, “Managing Security for Users and Applications.”

Security Problem 1

Three employees need to use Essbase: Sue Smith, Bill Brown, and Jane Jones. Each requires update access to all databases in the Sample application.

Solution:

Because the users need update access to only one application, they do not need to have Supervisor privilege. Because the users do not need to create or delete applications, users, or groups, they do not need to be defined as special types of users with Create/Delete privilege. All these users need is Application Designer privilege for the Sample application.

The supervisor should:

1. Install the Application Manager on each user’s client PC.
2. Create (or edit) Sue, Bill, and Jane as ordinary users, but use the App Access button in the New User dialog box (Figure 185 in Chapter 15, “Managing Security for Users and Applications”) to assign Application Designer privilege to each.

This gives each of the three users full access to the application, regardless of the global application access level.

Note: If the users had already been created without Application Designer privilege, the supervisor could also assign those privileges to the three users by choosing Security > Application.

Security Problem 2

Three employees need to use Essbase: Sue Smith, Bill Brown, and Jane Jones. Sue and Bill require full access to all databases in the Sample application. Jane requires full calculate access to all databases in the Sample application, but she does not need to define or maintain database definitions.

Solution:

The supervisor should:

1. Install the Application Manager on Sue and Bill's client PCs.
2. Create Sue and Bill as ordinary users, and use the App Access button in the New User dialog box (Figure 185 in Chapter 15, "Managing Security for Users and Applications") to assign Application Designer privilege to each.
3. Define global Calculate access for the Sample application as the Minimum Database Access setting in the Application Settings dialog box (see Figure 196 in Chapter 15, "Managing Security for Users and Applications"). This gives all additional users Calculate access to all databases for the application.
4. Create Jane as an ordinary user with no additional privileges. She inherits the Calculate access from the application global setting.

Security Problem 3

Three employees need to use Essbase: Sue Smith, Bill Brown, and Jane Jones. Sue and Bill require full access to all databases in the Sample application. Jane requires full update and calculate access to all databases within the Sample application, but she will not define or maintain the database definitions. Additional users will be added, all of whom will require Read access to all databases.

Solution:

Because the current users have different needs for application and database access, define their user privileges individually. Then, to save time assigning individual Read privileges for future users, make Read the global setting for the application. (It doesn't matter in what order you assign the user privileges and the global access.)

The supervisor should:

1. Install the Application Manager on Sue and Bill's client PCs.
2. Create Sue and Bill as ordinary users, and use the App Access button in the New User dialog box (Figure 185 in Chapter 15, "Managing Security for Users and Applications") to define Application Designer privilege to each.
3. Create Jane as an ordinary user, and use the App Access and the DB Access buttons to set her database access to Calculate.
4. Define global Read access for the Sample application as the Minimum Database Access setting in the Application Settings dialog box (Figure 196 in Chapter 15, "Managing Security for Users and Applications"). This gives all additional users Read access to all databases in the Sample application.

Security Problem 4

Three employees need to use Essbase: Sue Smith, Bill Brown, and Jane Jones. Sue requires full access only to the Sample application; Jane requires calculate access to all members of the Basic database; Bill requires Read access to all members. No other users should have access to the databases.

Furthermore, Jane and Bill need to run report scripts that are defined by Sue.

Solution:

Because the different users have different needs for application and database access, define the global access setting as None, and assign the user privileges individually.

The supervisor should:

1. Install the Application Manager on all three users' client PCs. (Because Jane and Bill need to run the report scripts, they must use the Application Manager.)
2. Create the user Sue as an ordinary user, but use the App Access button in the New User dialog box (Figure 185 in Chapter 15, "Managing Security for Users and Applications") to give her Application Designer privilege for the Sample application.
3. Create Jane as an ordinary user, and use the **App Access** and the **DB Access** buttons to give her Calculate privilege for the Sample application.
4. Create Bill as an ordinary user and use the App Access and the DB Access buttons to give him Read privilege on the Sample application.

Security Problem 5

The Supervisor, Sue Smith, needs to perform some maintenance on the Sample application. She must make changes to the database outline and reload actual data. While she changes the application, Sue must prevent other users from connecting to the application.

Solution:

Sue should:

1. Use the Application Settings dialog box (Figure 196 in Chapter 15, “Managing Security for Users and Applications”) to disable the Allow Commands setting.

   This prevents other users from connecting to the application, and also prevents connected users from performing any further operations.

2. Choose Security > Locks to see if any users have active locks.

   If any users have active locks, Sue’s calculation or data load command might halt, waiting for access to the locked records. Sue can allow the users to complete their updates or clear them directly with the Remove Locks option.

3. After confirming that no users have active locks, proceed to perform maintenance on the application.

IBM® DB2® OLAP Server™ 8.1

Database Administrator's Guide

Volume II: Loading, Calculating, and Retrieving Data

SC18-7002-00

Printed in the U.S.A.

Part V Building Dimensions and Loading Data

This part describes how to move data from external data sources into Essbase databases. It explains the concepts behind dimension builds and data loads, how to create rules files to manipulate the records and fields in data sources, how to dynamically build or change dimensions in outlines, how to load data values into databases, and how to debug dimension builds and data loads. Part IV contains the following chapters:

- Chapter 18, "Introducing Dynamic Dimension Building," provides background material on adding and changing dimensions and members in an outline using a data source and a rules file.
- Chapter 19, "Building Dimensions Using a Rules File," describes how to add or change members in an outline using a data source and a rules file.
- Chapter 20, "Introducing Data Loading," explains data loading, including external data sources, rules files, free-form data loading, and how Hyperion Essbase handles security and multi-user issues while loading data.
- Chapter 21, "Setting up a Rules File to Manipulate Records," describes how to create a rules file for a particular data source, specify record manipulations, and validate and save the rules file using the Data Prep Editor.
- Chapter 22, "Manipulating Fields Using a Rules File," describes how to manipulate fields using a rules file, including how to ignore fields, order fields, map fields to member names, and change data values using the Data Prep Editor.
- Chapter 23, "Performing and Debugging a Data Load," describes how to load data with the Application Manager (including the kinds of data that you can load), how to choose the data source, how to set the error log file, how to start and finish a data load, and tips for loading data and debugging a data load.

Note: For optimization, see Chapter 50, "Optimizing Data Loads," and Chapter 49, "Optimizing Database Restructuring."

Chapter

18

Introducing Dynamic Dimension Building

Some Essbase dimensions, such as those related to product codes and customer numbers, contain hundreds or even thousands of members. It is more efficient to build, change, or remove these dimensions dynamically, using a data source and a rules file, than it is to add or change each member manually in Outline Editor.

- The data source can specify member names, their aliases, formulas and consolidation properties, the names of generations and levels, currency name and category, data storage properties, attributes, and UDAs (user-defined attributes).
- The rules file tells Essbase which build method to use, specifies whether the data is in sorted or random order, and tells Essbase how to transform the data before loading it. You use Data Prep Editor to create rules files. It is best to create a separate rules file for each dimension. For more information on data sources and rules files, see Chapter 20, “Introducing Data Loading.”

This chapter provides a background on dynamic dimension building.

For information on how to build dimensions dynamically, see Chapter 19, "Building Dimensions Using a Rules File."

Workflow for Creating Dimensions Using Rules Files

This section describes the process for building dimensions using a rules file. For more information on data sources and rules files, see Chapter 20, "Introducing Data Loading."

Before you create dimensions using a rules file, determine whether to use the same rules file for dimension building and data loading:

Reasons for using the same rules file	Reasons for using a different rules file
• To modify the outline based on the contents of the data source. • To load the data source and build new dimensions at the same time.	• To build an outline from scratch. • To perform different field operations. • To re-use the dimension build or data load separately. • To use data sources that contain no data values, only dimensions.

Note: You can neither merge nor separate rules files after you create them.

➤ To create dimensions using a rules file:

1. Use Data Prep Editor to create a dimension build rules file. Select View > Dimension Building Fields or click the Dimension Build button, , on the toolbar to put Data Prep Editor into dimension build mode.

   Set field and record operations in the rules file. The field and record operations determine how Essbase transforms the data before building the dimensions. For more information on how to set field and record operations, see Chapter 21, "Setting up a Rules File to Manipulate Records."

2. Select Options > Dimension Build Settings to open the Dimension Build Settings dialog box. This dialog box contains three tabs, each of which sets up a different part of the rules file:
   - Use the Dimension Definition tab to name the dimension and define its properties.
   - Use the Dimension Build Settings tab to specify the build method and indicate how to sort the members.
   - Use the Global Settings tab to specify which alias table to update, whether Essbase should select the dense/sparse configuration, and how to combine selection and rejection criteria.
3. Validate and save the rules file using the Options > Validate menu command. See "Validating and Saving Data Load Rules" on page 608.
4. Perform the dimension build. See Chapter 23, "Performing and Debugging a Data Load."

Introduction to Build Methods

The build method that you select determines the algorithm that Essbase uses to add, change, or remove dimensions, members, and aliases in the outline. The kind of build method that you select depends on your data source.

Each record in a data source defines a single member of a dimension. Essbase can read data sources in a variety of formats:

- Top-down data sources beginning with the most general information and progress to the most specific. Each record specifies the parent's name, the child's name, the children of that child, and so forth.
- Bottom-up data sources provide a complete description of the member, beginning with the most specific information and progressing to the most general. Each record specifies the name of the member, followed by the name of its parent, the name of its parent's parent, and so forth.
- Parent-child data sources specify the name of the parent and the name of the new child member, in that order, although they can specify other information as well.
- Other data sources consist of lists of new members; they do not specify where in the outline these members belong. Essbase provides different algorithms to determine where to add these members.

The following table provides guidelines to help you select the appropriate build method for your data source:

Table 27: Build Method Guidelines

Type of Data in Data Source	**Examples**	**Desired Operation**	**Build Method to Use**	**Specification**
Top-down data	Year, Quarter, Month	Modify the properties of existing dimensions and members	Generation references	The generation number for each field
Bottom-up data	Month, Quarter, Year	Create shared members that roll up into different generations Modify the properties of existing dimensions and members	Level references	The level number for each field
A parent followed by its child	Cola, Diet Cola	Create shared members that roll up into different generations Share non-leaf members Modify properties of existing dimensions and members	Parent-child references	Whether each field is the parent or child
A list of base dimension members and their attributes	Cola 16oz Can, Root Beer 14oz Bottle	Add members to the attribute dimension and associate them with members of the base dimension	Generation, level, or parent-child references, depending on the organization of the source data	The number for each field, as appropriate: • Generation number or level number of the associated member of the base dimension • Zero

Table 27: Build Method Guidelines (Continued)

Type of Data in Data Source	Examples	Desired Operation	Build Method to Use	Specification
A list of new members	Jan, Feb, Mar, April	Add them all as children of an existing (possibly a "dummy") parent	Add as child of specified parent	
	800-10, 800-20	Add them at the end of the dimension	Add as sibling of lowest level	
	800-10, 800-20	Add each new member to a dimension with similar members	Add as sibling of member with matching string	

Note: Essbase does not support concurrent attribute association with the Add as Sibling and Add as Child build methods.

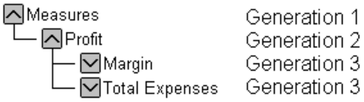
Using Generation References

Top-down data sources are organized from the highest level down. Each record begins with the most general information and progresses to the most specific. The name of the member is at the end of the record. When building a dimension with a top-down data source, use the generation references build type. Use the rules file to specify the generation number and field type for each field in the data source. For more information about creating a rules file, see Chapter 21, "Setting up a Rules File to Manipulate Records."

Essbase numbers members within a dimension according to their hierarchical position in the dimension. These are called *generation references*. Dimensions are always generation 1. All members at the same branch in a dimension are called *generations*. Generations are numbered top-down according to their position relative to the dimension.

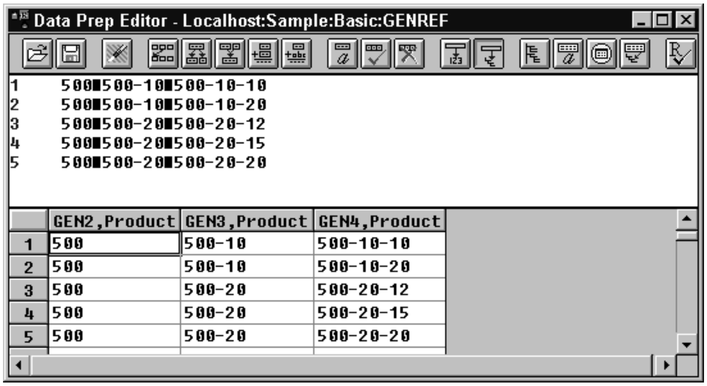
For example, the Measures dimension in the Sample Basic database is generation 1. It has a Profit member, which is generation 2. Profit has members for Margin and Total Expenses, which are generation 3. To build a dimension using the generation references build method, you must specify the generation reference number in the rules file.

Figure 223: Generations



The top half of Figure 224 shows a top-down data source used to build the Product dimension, `GENREF.TXT`. The bottom half of Figure 224 shows the rules file for this data source, `GENREF.RUL`. The rules file specifies the generation number for each field in the data source. For more information, see "Step 4a: Setting Rules File Field Types" on page 545.

Figure 224: Rules File for Generation Build

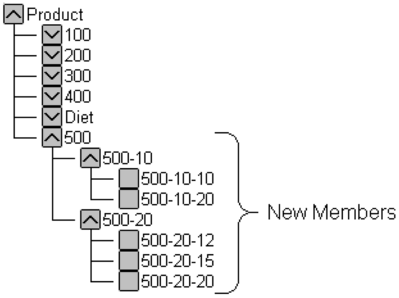


```
1     500■500-10■500-10-10
2     500■500-10■500-10-20
3     500■500-20■500-20-12
4     500■500-20■500-20-15
5     500■500-20■500-20-20
```

	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

Figure 225 shows the tree Essbase builds from this data source and rules file:

Figure 225: Generation References



Null Processing with Generation References

When you use the generation references build method, you can also choose to use null processing. When null processing is enabled, Essbase processes nulls as follows:

- If the null occurs where Essbase expects a GENERATION field, Essbase promotes the next GENERATION field in place of the missing one. For example:

```
GEN2,Products   GEN3,Products   GEN4,Products
100                             100-10a
```

  When Essbase reads the above file, it promotes the GEN4 field (100-10a) to GEN3.

- If a null occurs directly before a secondary field, Essbase ignores the secondary field. Secondary field options are alias, property, formula, duplicate generation, duplicate generation alias, currency name, currency category, attribute parent, UDA, and a name of an attribute dimension. For example:

```
GEN2,Products  ALIAS2,Products  GEN3,Products  GEN4,Products
                                100-10         100-10a
```

  When Essbase reads the above file, it ignores the ALIAS2 field and promotes the GEN3 field (100-10) to GEN2 and the GEN4 field (100-10a) to GEN3.

- If the null occurs where Essbase expects a secondary field, Essbase ignores that field and continues loading. For example:

  ```
  GEN2,Products  ALIAS2,Products  GEN3,Products  GEN4,Products
  100                             100-10         100-10a
  ```

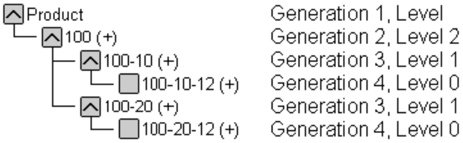
  When Essbase reads the above file, it ignores the ALIAS2 field.

Using Level References

In bottom-up data sources, each record defines a single member of a dimension. The definition begins with the most specific information about the member and provides progressively more general information. A typical record would specify the member itself, then the name of its parent, then its parent's parent, and so forth.

Levels are defined from a bottom-up hierarchical structure. In the outline in Figure 226, for example, the lowest level members are at the bottoms of the branches in the Product dimension.

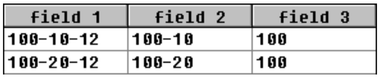
Figure 226: Generation and Level Numbers



```
Product                    Generation 1, Level
└─100 (+)                  Generation 2, Level 2
   ├─100-10 (+)            Generation 3, Level 1
   │  └─100-10-12 (+)      Generation 4, Level 0
   └─100-20 (+)            Generation 3, Level 1
      └─100-20-12 (+)      Generation 4, Level 0
```

To build the outline in Figure 226, you can use the bottom-up data source shown in Figure 227.

Figure 227: Bottom-up Data Source



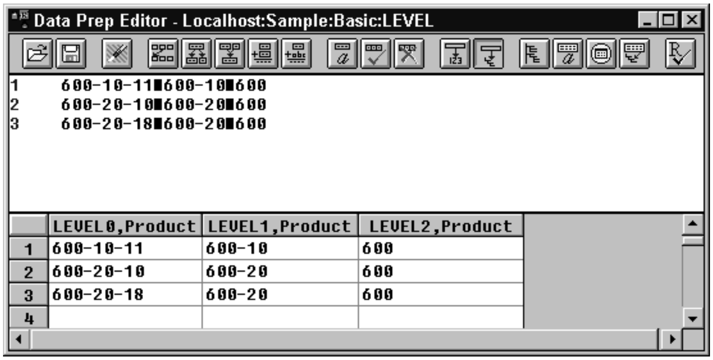
field 1	field 2	field 3
100-10-12	100-10	100
100-20-12	100-20	100

In a level reference build, the lowest level members are sequenced left to right. Level 0 members are in the first field, level 1 members are in the next field, and so on. This is the opposite of how data is specified for generation references (top-down).

The rules file in Figure 228 uses the level reference build method to add members to the Product dimension in the Sample Basic database. The first column of the data source contains new members (100-10-12, 100-10-16, 100-20-12, and 100-20-16). Subsequent columns contain their parents. (Family-18 oz., and Family-24 oz.)

The rules file specifies the level number and field type for each field in the data source. For more information, see "Step 4a: Setting Rules File Field Types" on page 545. To build the tree in Figure 229 for example, use Figure 228 to set up the data source, `LEVEL.TXT`, and rules file, `LEVEL.RUL` to match.

Figure 228: Rules File for Level Build



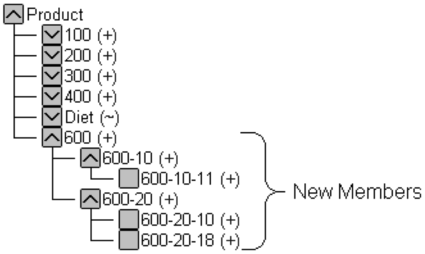
Data Prep Editor - Localhost:Sample:Basic:LEVEL

```
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
4			

Figure 229 shows the tree Essbase builds from this data source and rules file.

Figure 229: Levels



Null Processing with Level References

When you use the level references build method, you can also choose to use null processing. When null processing is enabled, Essbase processes nulls as follows:

- If the null occurs where Essbase expects a LEVEL field, Essbase promotes the next LEVEL field in place of the missing one. For example:

  ```
  LEVEL0,Products  LEVEL1,Products  LEVEL2,Products
                   100-10           100
  ```

  When Essbase reads the above file, it promotes the LEVEL1 field(100-10) to LEVEL0 and the LEVEL2 field(100) to LEVEL1.

- If a null occurs directly before a secondary field, Essbase ignores the secondary field. Secondary field options are alias, property, formula, duplicate level, duplicate level alias, currency name, currency category, attribute parent, UDA, and a name of an attribute dimension. For example:

```
LEVEL0,Products  ALIAS0,Products  LEVEL1,Products  LEVEL2,Products
                 Cola             100-10           100
```

  When Essbase reads the above file, it ignores the ALIAS0 field and promotes the LEVEL1 field(100-10) to LEVEL0, the LEVEL2 field(100) to LEVEL1.

- If the null occurs where Essbase expects a secondary field, Essbase ignores that field and continues loading. For example:

```
LEVEL0,Products  ALIAS0,Products  LEVEL1,Products  LEVEL2,Products
100-10a          100-10                            100
```

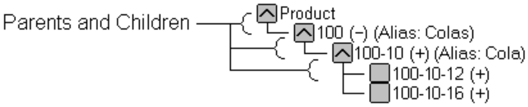
  When Essbase reads the above file, it ignores the ALIAS0 field.

Using Parent-Child References

Use the parent-child references build method with data sources in which each record specifies the name of the new member and the name of the parent to which you want to add it.

Members in a database exist in parent-child relationships to one another. Figure 230 shows part of the Product dimension with its parent and children relationships identified.

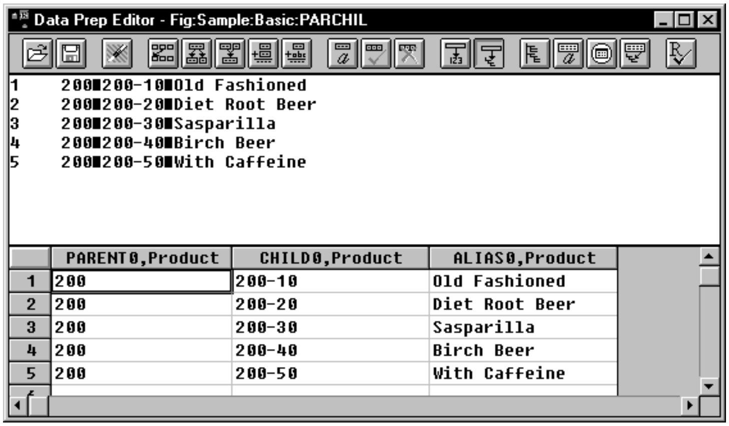
Figure 230: Parents and Children



Parent-child data sources must contain at least two columns: the parent column and the child column, in that order. They can include columns with other information as well (for example, the alias of the new member, its attributes or its properties). The data source *cannot* specify more than one parent or more than one child, and cannot reverse the order of the parent and child columns.

In a parent-child build, the rules file specifies which column is the parent and which the child. For more information, see "Step 4a: Setting Rules File Field Types" on page 545. For example, the top half of Figure 231 shows a data source, `PARCHIL.TXT`, in which each record specifies the name of a parent and the name of its child, in that order. The bottom half of the figure shows the rules file, PARCHIL.RUL, that specifies which column is the parent and which is the child. In addition to parent and child fields, this example demonstrates associating aliases with the parent field.

Figure 231: Rules Files for Parent-Child Build



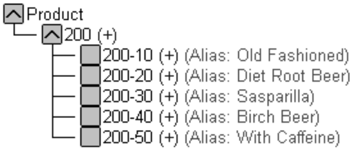
Data Prep Editor - Fig:Sample:Basic:PARCHIL

```
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

Figure 232 shows the tree that Essbase builds from this data source and rules file.

Figure 232: Parents and Children



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)

Adding a List of New Members

If a data source consists of a list of new members, without specifying their ancestors, Essbase must decide where in the outline to add the new members. Essbase provides three different build methods for this type of data source. Each uses a different algorithm to determine where to add the new members.

Depending upon which build method you specify, Essbase can:

- Add each new member as a sibling of the existing member whose text most closely matches its own.
- Add each new member as a sibling of the lowest existing member.
- Add all new members as children of a specified parent (generally a “dummy” parent).

After you select your build method, you must specify the dimension to which each field maps. See “Step 4a: Setting Rules File Field Types” on page 545.

After Essbase has added all the new members to the outline, use Outline Editor to examine the outline. If necessary, move the new members into their correct positions. See Chapter 7, “Creating and Changing Database Outlines.”

Adding Members Based Upon String Matches

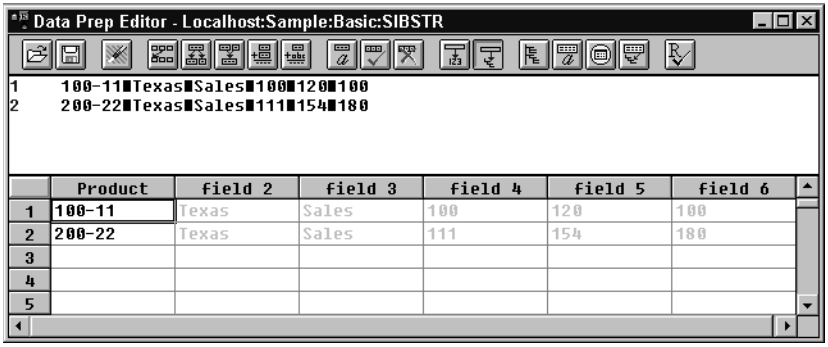
You can add new members from a data source to an existing dimension by matching strings with existing members. When Essbase encounters a new member in a data source, it scans the outline for a member name with similar text. Essbase then adds the new member as a sibling of the member with the closest string match.

For example, the data source in Figure 233, `SIBSTR.TXT`, contains two new members to add to the Product dimension in the Sample Basic database, 100-11 and 200-22. They are similar to strings in the Product dimension (they contain 3 digits, a dash, and two digits).

To add these members dynamically to the database, set the following values to match the rules file, `SIBSTR.RUL`, in Figure 233.

- For the Product column (field 1) on the Dimension Building Properties tab of the Field Properties dialog box:
  - Do not select a field type for the field.
  - Set the dimension for the field to Product. Field 1 is displayed as Product, as shown in Figure 233.
- For the other data fields that are irrelevant to this operation (fields 2 through 6), on the Global Properties tab of the Field Properties dialog box, select "Ignore field during dimension build."
- In the Dimension Build Settings dialog box, for the Product dimension, select the "Add as sibling of matching string" method.

Figure 233: Rules File Fields Set to Add Members as Siblings with String Matches

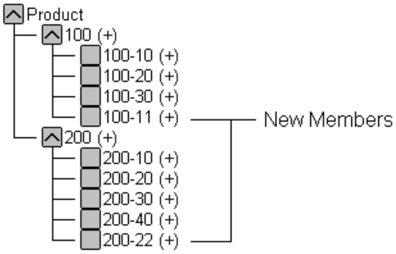


```
1    100-11■Texas■Sales■100■120■100
2    200-22■Texas■Sales■111■154■180
```

	Product	field 2	field 3	field 4	field 5	field 6
1	100-11	Texas	Sales	100	120	100
2	200-22	Texas	Sales	111	154	180
3						
4						
5						

When you load the data in Figure 233, Essbase builds the following member tree:

Figure 234: Adding Members as Siblings with String Matches Tree



Adding Members as Siblings of the Lowest Level

You can add new members from a data source as siblings of members that reside at the lowest level of a dimension, that is, the lowest branch. When Essbase encounters a new member in a data source, it scans the outline for the lowest branch of members. Essbase adds the new member as a sibling of these members.

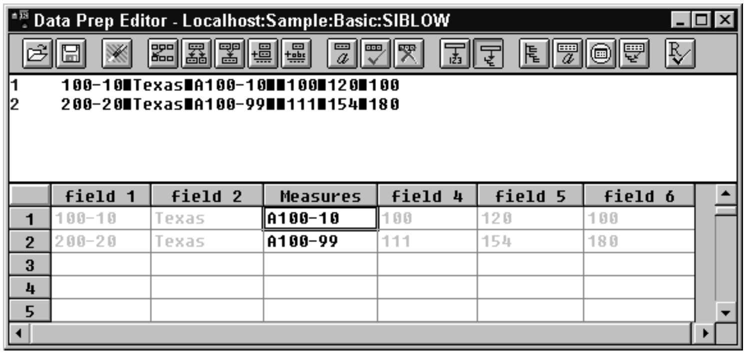
Note: If the outline contains more than one group of members at this level, Essbase adds the new member to the first group of members it encounters.

For example, the data source, `SIBLOW.TXT`, and rules file, `SIBLOW.RUL`, in Figure 235 contain new members (A100-10 and A100-99) to add to the Measures dimension in the Sample Basic database.

To add these members dynamically to the database, set the following values in the rules file:

- For the Measures column (field 3) on the Dimension Building Properties tab of the Field Properties dialog box:
  - Do not select a field type for the field.
  - Set the dimension for the field to Measures. Field 3 is displayed as Measures, as shown in Figure 235.
- For the other data fields that are irrelevant to this operation (fields 1, 2, 4, 5, and 6), on the Global Properties tab of the Field Properties dialog box, select “Ignore field during dimension build.”
- In the Dimension Build Settings dialog box, for the Measures dimension, select the “Add as sibling of lowest level” build method.

Figure 235: Rules File Fields Set to Add Members as Siblings of the Lowest Level

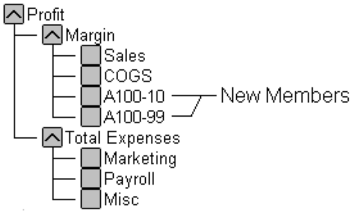


```
1     100-10■Texas■A100-10■■100■120■100
2     200-20■Texas■A100-99■■111■154■180
```

	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
3						
4						
5						

When you load the data in Figure 235, Essbase builds the following member tree:

Figure 236: Adding Members as Siblings of the Lowest Level



Adding All New Members to a Specified Parent

You can add all new members as children of a specified parent, generally a “dummy” parent. When Essbase adds all new members to the outline, review the added members and move or delete them in Outline Editor.

When Essbase encounters a new member in the data source, it adds the new member as a child of the parent defined in the Dimension Build Settings dialog box. This parent must be part of the outline before you start the dimension build.

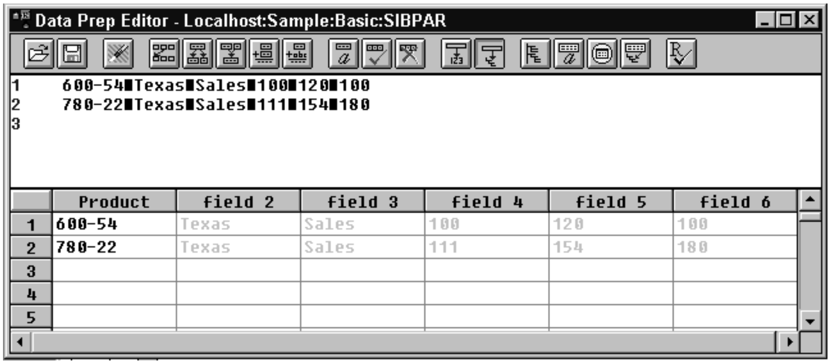
For example, the data source in Figure 237, `SIBPAR.TXT`, contains new members, 600-54 and 780-22, for the Product dimension (field 1). Assume that you have already added a new member called NewProducts to the outline under the Products dimension.

To have Essbase add these members dynamically to the database under the NewProducts member, set the following values:

- For the Product column (field 1) on the Dimension Building Properties tab of the Field Properties dialog box:
  - Do not select a field type for the field.
  - Set the dimension for the field to Product. Field 1 is displayed as Product, as shown in Figure 237.
- For the other data fields that are irrelevant to this operation (fields 2 through 6), on the Global Properties tab of the Field Properties dialog box, select “Ignore field during dimension build.”

- In the Dimension Build Settings dialog box, for the Product dimension, select the “Add as child of build” method and type `NewProducts` in the associated text box.

Figure 237: Rules File Fields Set to Add Members as a Child of a Specified Parent



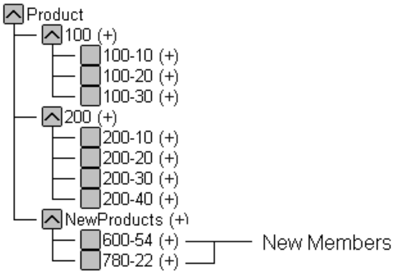
Data Prep Editor - Localhost:Sample:Basic:SIBPAR

```
1     600-54■Texas■Sales■100■120■100
2     780-22■Texas■Sales■111■154■180
3
```

	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
3						
4						
5						

When you load the data in Figure 237, Essbase builds the following member tree:

Figure 238: Adding Members as a Child of a Specified Parent



Building Attribute Dimensions and Associating Attributes

This section shows examples of rules files that are used to build attribute dimensions and to associate attributes with members of their base dimensions.

You can use rules files to build attribute dimensions dynamically, to add and delete members, and to establish or change attribute associations.

Working with attributes involves the three following operations:

- If the base dimension does not exist, you must build it.
- You must build the attribute dimension.
- You must associate members of the base dimension with members of the attribute dimension.

You can use any of three approaches to perform these operations.

When you use an all-at-once approach, you use a single rules file to build the base dimension and one or more attribute dimensions and to associate the attributes with members of the base dimension. Because this approach uses a single rules file, it can be the most convenient. Use this approach if the base dimension does not exist yet and each source data record contains all attribute information for each member of the base dimension.

If the base dimension is built in a separate step or already exists, you can build an attribute dimension and associate the attributes with the members of the base dimension in either of two ways:

- In a single step. You need only to define the attribute associations in the rules file. See "Associating Attributes" on page 505.
- In separate steps. Build the attribute dimension, and then associate the attribute members with members of the base dimension. You must use this approach when you build numeric attribute dimensions that are multilevel or that have members that represent different-sized ranges.

Building Attribute Dimensions

Before you build any attribute dimensions in a database, you must define the attribute member name formats for the outline. See “Working with Member Names in Attribute Dimensions” on page 239.

You can build attribute dimensions in either of the following two ways:

- The same way that you build standard dimensions, as described in “Workflow for Creating Dimensions Using Rules Files” on page 488.
- At the same time as you associate attributes with members of the base dimension, as described in “Associating Attributes” on page 505.

Essbase does not support concurrent attribute association with the following three build methods: “Add as sibling of mbr with matching string,” “Add as sibling of lowest level,” and “Add as child of.”

When you define the rules file for building attribute dimensions, be sure to specify the base dimension and the name of the attribute dimension file, as described in “Naming New Attribute Dimensions” on page 537.

Associating Attributes

Whether you build the attribute dimension at the same time as you associate its members with members of the base dimension or perform these tasks in separate steps, define the fields as described in this section.

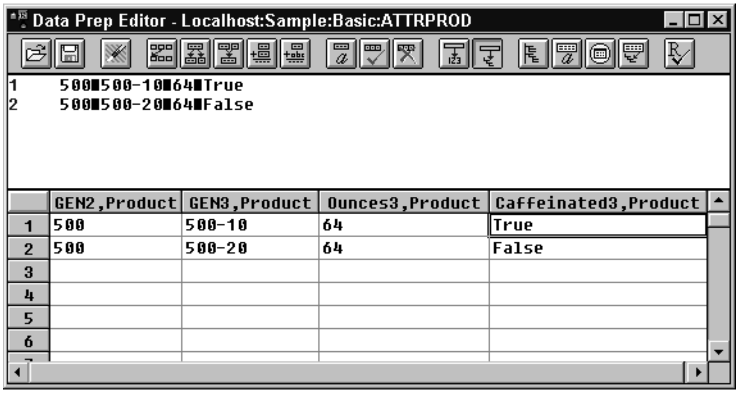
However, when you are working with a multilevel attribute dimension or with an attribute dimension of the type numeric, Boolean, or date, the rules file requires an additional field. For a complete example of a multilevel situation, see “Working with Multilevel Attribute Dimensions” on page 507.

Each record in the source data must include at least a column for the member of the base dimension and a column for the member’s attribute value. In the same source data record you can include additional columns for other attributes that you want to associate with the member of the base dimension. You must position the field for the member of the base dimension before any of the fields for the members of the attribute dimension.

Define the field type for the attribute dimension member as the name of the attribute dimension, use the generation or level number of the associated member of the base dimension, and specify the base dimension name. For example, as shown in the `ATTRPROD.RUL` file in Figure 239, the field definition

Ounces3,Product specifies that the field contains members of the Ounces attribute dimension. Each member in this field is associated with the data field that is defined as the generation 3 member of the base dimension Product. Based on this field definition, Essbase associates the attribute 64 with product 500-10.

Figure 239: Rules File for Associating Attributes



```
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
3				
4				
5				
6				

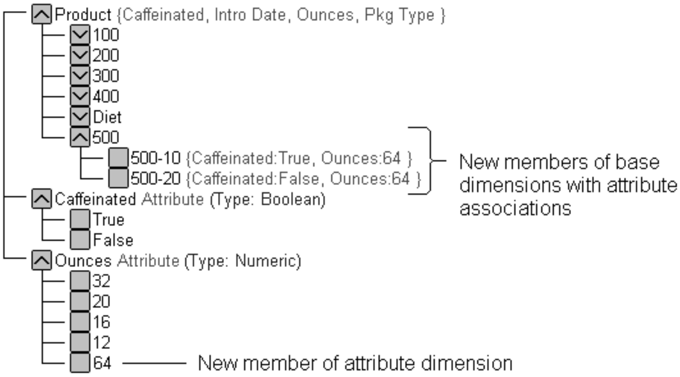
To have Essbase use the data attribute columns to build members in the attribute dimensions, on the Dimension Build Settings tab in the Dimension Build Settings dialog box, for the base dimension, leave the Do Not Create Mbrs option clear. See “Step 3: Specifying Changes to Dimensions” on page 543.

When you are working with numeric ranges, you may have to build attribute dimensions and perform associations in separate steps. See “Working With Numeric Ranges” on page 510.

The Caffeinated3,Product field in the example in Figure 239 shows how to associate attributes from additional single-level attribute dimensions. Because the base dimension is already specified, you need only to define an additional field for each attribute that you want to associate with the member of the base dimension.

The file in Figure 239 associates attributes as shown in the resulting outline in Figure 240. The members 500, 500-10, and 500-20 are new members of the base dimension, Product. The member 64 is a new member of the Ounces attribute dimension.

Figure 240: Associating Attributes


Product {Caffeinated, Intro Date, Ounces, Pkg Type }
100
200
300
400
Diet
500
500-10 {Caffeinated:True, Ounces:64 }
500-20 {Caffeinated:False, Ounces:64 }
New members of base dimensions with attribute associations
Caffeinated Attribute (Type: Boolean)
True
False
Ounces Attribute (Type: Numeric)
32
20
16
12
64
New member of attribute dimension

Updating Attribute Associations

You can also use the rules file shown in Figure 239 to change attribute associations. Make sure that you select “Allow Association Chgs” for the base dimension in the Dimension Build Settings tab of the Dimension Build Settings dialog box. See “Step 3: Specifying Changes to Dimensions” on page 543.

Working with Multilevel Attribute Dimensions

Multilevel, numeric, Boolean, or date attribute dimensions can have duplicate level 0 members. For example, associated with a Product dimension you can have a Size attribute dimension with two levels. Level 1 categorizes sizes by men or by women. The level 0 members (attributes) are the actual sizes. You can have a member named 8 under Women and member named 8 under Men.

When the attribute is part of a multilevel numeric, Boolean, or date attribute dimension, the source data must include columns for all generations or levels of the attribute dimension. In the rules file, you must make copies of all fields that comprise the levels of the attribute dimension. Define the first set of the attribute fields to build the attribute dimension. Define the second set of attribute fields to associate the attributes with the base dimension member. To ensure association with the correct attribute, to indicate the parent field for the attribute field, make a copy of the parent field and set the copy of the parent field as the field type Attribute Parent.

The position of the fields in the rules file is important.

- Place the copied attribute dimension field or fields that define the association immediately to the right of the field for the members of the base dimension.
- For a multilevel attribute dimension, place the attribute parent field immediately to the left of the field that is the child of the attribute parent.

The following procedure describes how to define the fields in the rules file to build this multilevel attribute dimension and associate its members with members of its base dimension. This example uses the level references build method.

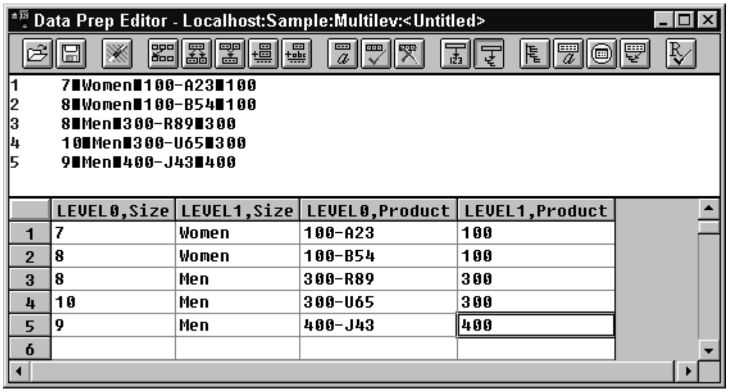
1. In the rules file, in field 1 and field 2, define the attribute dimension fields in the same way that you define standard dimensions; specify level or generation type and number and dimension name.

   Essbase uses these two fields to build the attribute dimension.

2. Define the fields for building the base dimension, in this example, the level 0 and level 1 fields for the Product dimension.

   Figure 241 shows the fields of the rules file at this stage.

Figure 241: Defining Multilevel Attribute Dimensions Before Adding the Association Fields



Data Prep Editor - Localhost:Sample:Multilev:<Untitled>

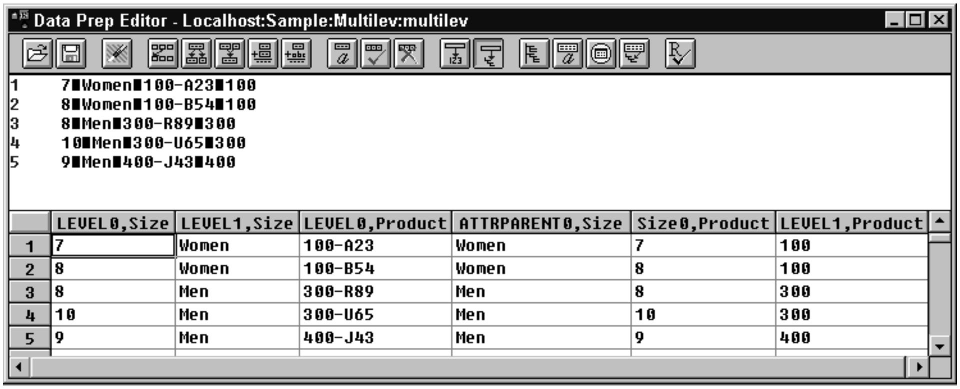
```
1     7■Women■100-A23■100
2     8■Women■100-B54■100
3     8■Men■300-R89■300
4     10■Men■300-U65■300
5     9■Men■400-J43■400
```

	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
6				

3. To define the association, select **Field > Create Using Join** to make a copy of the field that contains the level 0 attribute. In this example, make a copy of field 1.
   - Use the attribute dimension name as the field type and specify the generation or level number of the member of the base dimension with which Essbase associates the attribute; for example, Size0.
   - Specify the base dimension; for example, Product.
   - Select **Field > Move** to move the new field immediately to the right of the field for the base dimension with which Essbase associates the attribute. In this example, move it to the right of the field Level0,Product.
4. Make a copy of the field containing the parent of the attribute field; in this example, make a copy of field 2.
   - Set the field type of this new field as **Attribute Parent** and specify the generation or level number of the base member with which you want Essbase to associate the attribute; for example, ATTRPARENT0.
   - Specify the attribute dimension; for example, Size.
   - Move the ATTRPARENT field immediately to the left of the attribute association field that you created in Step 3.

As shown in Figure 242, the rules file now contains the field definitions to build the attribute dimension Size and to associate the members of Size with members of the base dimension Product.

Figure 242: Source Data and Rules File for Building a Multilevel Attribute Dimension

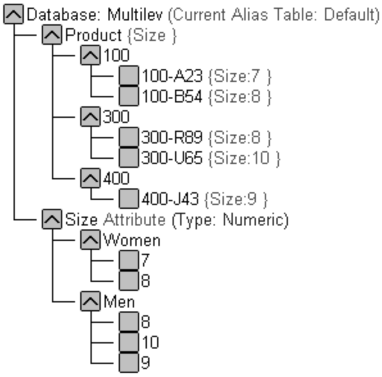


```
1      7■Women■100-A23■100
2      8■Women■100-B54■100
3      8■Men■300-R89■300
4      10■Men■300-U65■300
5      9■Men■400-J43■400
```

	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

When you run a dimension build with the data shown in Figure 242, Essbase builds the Size attribute dimension and associates its members with the members of the base dimension. Figure 243 shows the updated outline.

Figure 243: Multilevel Attribute Dimension



```
Database: Multilev (Current Alias Table: Default)
    Product {Size }
        100
            100-A23 {Size:7 }
            100-B54 {Size:8 }
        300
            300-R89 {Size:8 }
            300-U65 {Size:10 }
        400
            400-J43 {Size:9 }
    Size Attribute (Type: Numeric)
        Women
            7
            8
        Men
            8
            10
            9
```

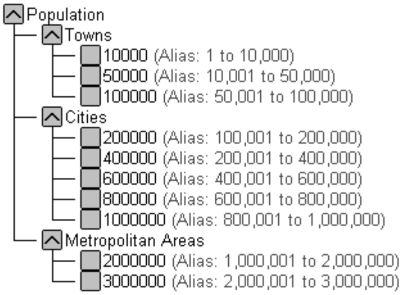
Working With Numeric Ranges

In many cases, you can use one rules file in a single dimension build operation to dynamically build attribute dimensions for numeric ranges and to associate the members of the base dimension with the ranges. However, in the following situations you must use two rules files, one to build the attribute dimension and one to associate the attributes with members of the base dimension:

- When the size of the range is different for different members. For example, you can define several shorter ranges for towns and cities with smaller populations, larger ranges for mid-sized cities, and ranges of 1,000,000 for cities with large populations.
- When the ranges are members of a multilevel attribute dimension. For example, the Population attribute dimension can have level 1 members that categorize the population ranges as Towns, Cities, and Metropolitan Areas.

The Population attribute dimension shown in Figure 244 demonstrates both situations. Population is a multilevel, numeric attribute dimension with level 0 members representing ranges of different sizes.

Figure 244: Numeric Attribute Dimension with Different-Sized Ranges



```
Population
    Towns
        10000 (Alias: 1 to 10,000)
        50000 (Alias: 10,001 to 50,000)
        100000 (Alias: 50,001 to 100,000)
    Cities
        200000 (Alias: 100,001 to 200,000)
        400000 (Alias: 200,001 to 400,000)
        600000 (Alias: 400,001 to 600,000)
        800000 (Alias: 600,001 to 800,000)
        1000000 (Alias: 800,001 to 1,000,000)
    Metropolitan Areas
        2000000 (Alias: 1,000,001 to 2,000,000)
        3000000 (Alias: 2,000,001 to 3,000,000)
```

You must use one rules file to build this Population dimension and a second rules file to associate its members as attributes of members of the base dimension.

Building the Attribute Dimension

First, create a rules file that uses the generation, level, or parent-child build method to build the attribute dimension. In this rules file, be sure to specify the following:

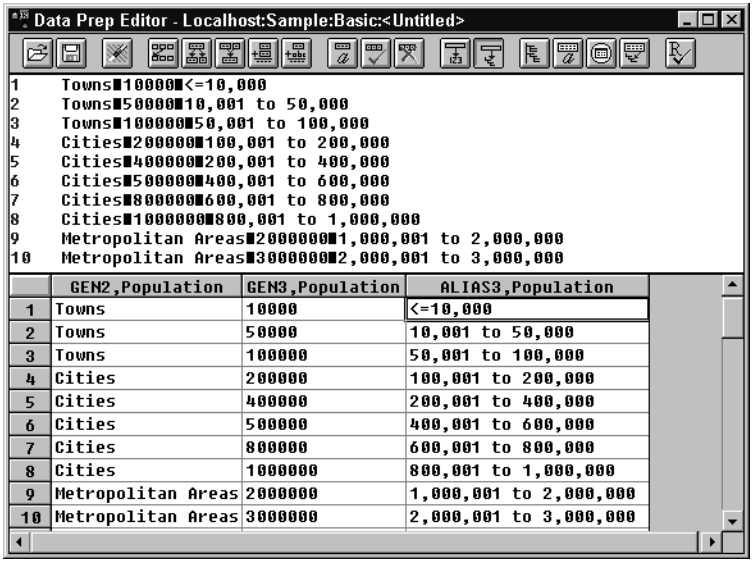
- The name of the attribute dimension and its associated base dimension. See “Naming New Attribute Dimensions” on page 537.
- The fields for building the attribute dimension.

The source data must be in attribute sequence, in ascending order. If the ranges have different sizes, the source data must include a record for every attribute range.

Note: In later builds you cannot insert attribute members between existing members.

To use the generation method to build the outline in Figure 244, you must sequence the source data in ascending sequence, based on the numeric attribute value. Define the fields in a rules file as shown in Figure 245.

Figure 245: Rules File for Building a Numeric Attribute Dimension with Ranges



```
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

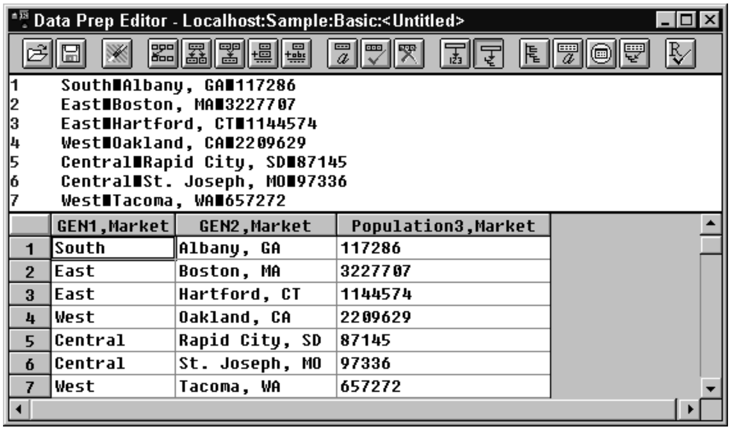
Figure 245 also shows how you can associate aliases with attributes.

Associating the Base Dimension Members with Their Range Attributes

After you build the numeric attribute dimension ranges, you need a rules file to associate the members of the base dimension with their attributes. The source data includes fields for the members of the base dimension and a field for the data values that Essbase uses to associate the appropriate Population attribute.

Define the rules file as shown in Figure 246.

Figure 246: Rules File for Associating Numeric Range Attributes



```
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

When you define the association field (for example, Population3,Market) be sure to click the Ranges button to open the Numeric Range Rules dialog box. Select “Place attribute members within a range.”

Note: Figure 246 includes a city, Boston, whose population of 3,227,707 is outside the existing ranges in the attribute dimension in Figure 244. (The ranges in Figure 244 go up only to 3,000,000.)

To allow for values in the source data that are outside existing ranges in the attribute dimension, enter a range size, such as 1000000, in the Numeric Range Rules dialog box. Essbase uses the range size to add members to the attribute dimension above the existing highest member or below the existing lowest member, as needed.

CAUTION: After you associate members of the base dimension with members of the attribute dimension, be aware that if you manually insert new members into the attribute dimension or rename existing members of the attribute dimension, you may invalidate existing attribute associations.

Consider an example where numeric range attributes are defined as “Tops of ranges” and an attribute dimension contains members 100, 200, 500, and 1000. A base dimension member with the value 556 is associated with the attribute 1000. If you rename a member of the attribute dimension from 500 to 600, the base dimension member with the value 556 now has an invalid association. This base member is still associated with the attribute 1000 when it should now be associated with the attribute 600.

If you manually insert new members or rename existing members, to ensure associations are correct, you should rerun the dimension build procedure that associates the base members with the changed attribute dimension. For example, rerunning the attribute association procedure would correctly associate the member of the base dimension with the value 556 with the new attribute 600.

Working With Ranges

To ensure the validity of the attribute associations, you must be careful to select the correct dimension building options and to perform the builds in the proper sequence.

Adding or Changing Members of the Attribute Dimension: After you associate members of a base dimension with their numeric attribute ranges, if you manually insert new members or rename existing members in the attribute dimension, you should make sure that associations between attributes and base members are correct. To ensure that the associations are correct, you can do one of the following:

- Rerun the dimension build procedure that associates the base members with the changed attribute dimension.
- Through Outline Editor, manually review and fix, as needed, the associations of all base dimensions.

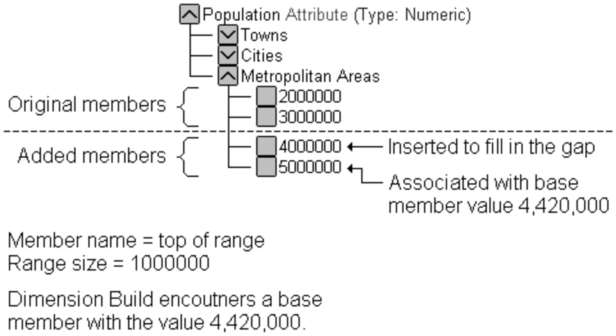
Deleting Members from the Attribute Dimension: You can delete all existing members of an attribute dimension so you can rebuild the dimension with the new data. Select “Delete all members of this attribute dimension” in the Numeric Range Rules dialog box. Essbase uses the start value and range size value to rebuild the attribute dimension. To ensure proper attribute association, on the Dimension Build Settings tab in the Dimension Build Settings dialog box, for the base dimension you must select the “Allow Association Chgs” option.

Adding Members to the Base Dimension: You can use the same rules file to add new members simultaneously to the base dimension and to associate the new members with their numeric range attributes. In the Numeric Range Rules dialog box for the attribute dimension, be sure to provide a value for the range size.

If Essbase encounters a new base dimension value that is greater than the highest attribute member by more than the range size or is lower than the lowest attribute member by more than the range size, it creates members in the attribute dimension for any intermediate ranges.

Consider the example, in Figure 244, where numeric range attributes are defined as “Tops of ranges.” The highest value member of the Population attribute dimension is 3000000. If the source data includes a record with the population 4,420,000 and the range size is 1000000, Essbase adds two members to the attribute dimension, 4000000 and 5000000, and associates the base member with the 5000000 attribute.

Figure 247: Dynamically Adding Attribute Range Members


Population Attribute (Type: Numeric)
Towns
Cities
Metropolitan Areas
Original members
2000000
3000000
Added members
4000000
5000000
Inserted to fill in the gap
Associated with base member value 4,420,000
Member name = top of range
Range size = 1000000
Dimension Build encoutners a base member with the value 4,420,000.

When you add range members and base dimension members at the same time, Essbase does not create aliases for the new members of the attribute dimension. If you want aliases that describe the range values for the new members of the attribute dimension, you must add the aliases in a separate operation.

Rules Summary for Building Attribute and Base Dimensions

The following list describes a few areas unique to defining and associating attributes through dimension build.

Getting Ready

- Before running a dimension build, you must define the attribute member name formats for the outline. See "Working with Member Names in Attribute Dimensions" on page 239.
- Defining new attribute dimensions in a rules file is different from defining new standard dimensions in a rules file. See "Naming New Attribute Dimensions" on page 537.

Defining Fields in Rules Files

Rules files that are used to build single-level attribute dimensions require fewer field types than rules files that build and associate members of multilevel attribute dimensions.

- For single-level attribute dimensions, define the field that contains the attribute values as the field to be associated with members of the base dimension. Dimension build also uses this field to add new members to the attribute dimension. See "Associating Attributes" on page 505.
- For multilevel attribute dimensions, Essbase requires fields that define each generation or level in the attribute dimension in addition to fields that define the associations. Use the new field type, Attribute Parent, to identify fields that are parent members for the attribute members being associated. See "Working with Multilevel Attribute Dimensions" on page 507.

Controlling Adding New Attribute Members

When Essbase encounters attribute data values that are not members of the attribute dimension, it automatically adds the values as new members. To prevent adding new members to attribute dimensions, you must select the Do Not Create Mbrs option for the attribute dimension in the Dimension Build Settings dialog box. See "Step 3: Specifying Changes to Dimensions" on page 543.

Controlling Associations

- Essbase does not make changes to attribute associations unless you select the Allow Association Chgs option for the attribute dimension in the Dimension Build Settings page of the Dimension Build Settings dialog box. See "Step 3: Specifying Changes to Dimensions" on page 543.
- To enable automatic association of base members with attributes that represent ranges of values, make sure you have defined the size of the range in the Advanced Numeric Rules dialog box. See "Step 4b: Setting Rules File Field Information" on page 546.
- Essbase does not support concurrent attribute association with the two Add as Sibling methods and the Add as Child build method.

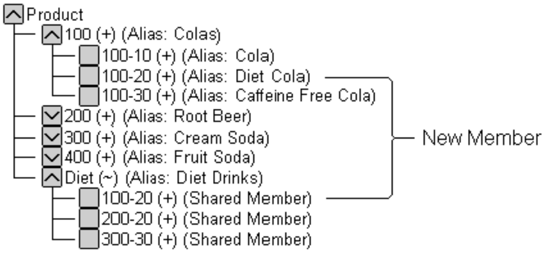
Note: Because attributes are defined only in the outline, the data load process does not affect them.

Building Shared Members Using a Rules File

The data associated with shared members comes from another real member with the same name. The shared member stores a pointer to data contained in the real member; thus the data is shared between the members and is only stored one time.

In the Sample Basic database, for example, the 100-20 (Diet Cola) member rolls up into the 100 (Cola) family and the Diet family.

Figure 248: Shared Members in the Sample Basic Database



```
Product
    100 (+) (Alias: Colas)
        100-10 (+) (Alias: Cola)
        100-20 (+) (Alias: Diet Cola)
        100-30 (+) (Alias: Caffeine Free Cola)
    200 (+) (Alias: Root Beer)
    300 (+) (Alias: Cream Soda)
    400 (+) (Alias: Fruit Soda)
    Diet (~) (Alias: Diet Drinks)
        100-20 (+) (Shared Member)
        200-20 (+) (Shared Member)
        300-30 (+) (Shared Member)
```

You can share members among as many parents as you want. Diet Cola has two parents, but you can define it to roll up into even more parents.

You can share members at multiple generations in the outline. In Figure 248, Diet Cola is shared by two members at generation 2 in the outline, but it could be shared by a member at generation 3 and a member at generation 4 as in Figure 256.

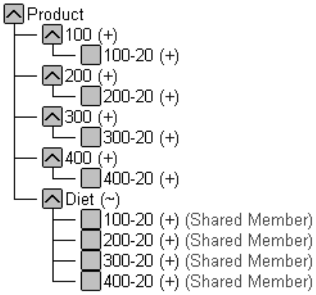
While creating shared members at different generations in the outline is easy in Outline Editor, they are a little more difficult to create using dynamic dimension building. You must pick your build method and format your data source carefully. The following sections describe how to build shared members in the outline using a data source and rules file.

Note: You should not create an outline where shared members are located before actual members in a dimension.

Sharing Members at the Same Generation

Members that are shared at the same generation roll up into the same branch. In the Sample Basic database, 100-20 (Diet Cola) is shared by two parents. Both parents roll up into the same branch, that is, the Product dimension, and both parents are at generation 2.

Figure 249: Members Shared at the Same Generation


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)

This is the simplest way to share members. You can share members at the same generation using the following build methods:

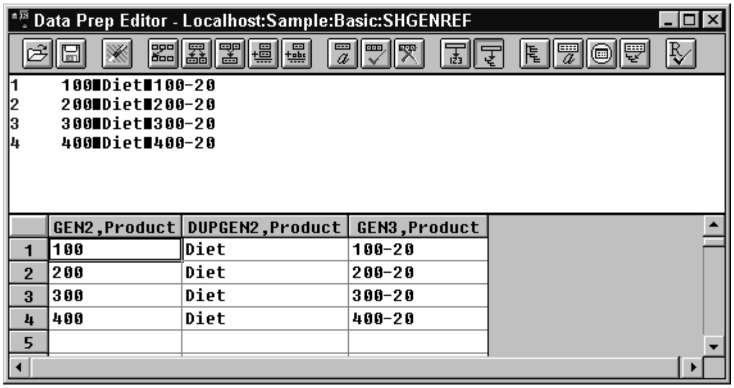
- “Creating Shared Members Using Generation References” on page 519
- “Creating Shared Members Using Level References” on page 520
- “Creating Shared Members Using Parent-Child References” on page 521

Creating Shared Members Using Generation References

To create shared member parents at the same generation using the generation references build method, define the field type for the parent of the shared member as DUPGEN. A *duplicate generation* is a generation with shared members for children. Use the same GEN number as the primary member.

For example, to create the Diet parent and share the 100-20, 200-20, 300-20, and 400-20 members, use the sample file, `SHGENREF.TXT`, and set up the rules file so that the fields look like `SHGENREF.RUL`, shown in Figure 250. Remember 100 is the Cola family, 200 is the Root Beer family, 300 is the Cream Soda family and the -20 after the family name indicates a diet version of the soda.

Figure 250: Sample Generation Shared Member Rules File



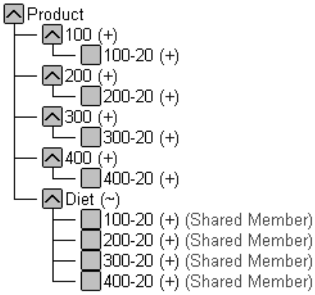
Data Prep Editor - Localhost:Sample:Basic:SHGENREF

```
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
5			

This builds the following tree:

Figure 251: Sample Generation Shared Member Rules Tree


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)

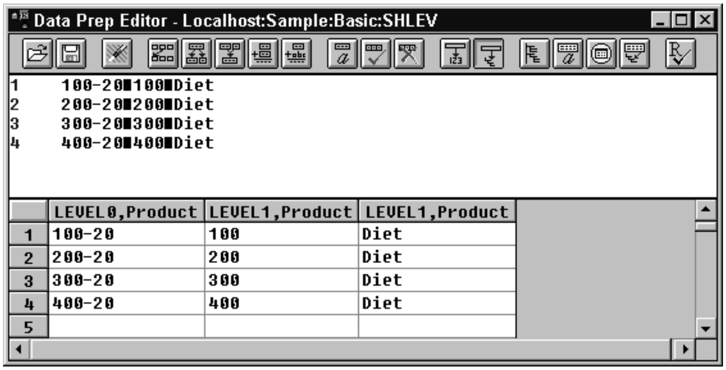
Creating Shared Members Using Level References

To create shared members at the same generation using the level references build method, first make sure that both the primary and any secondary roll-ups are specified in one record. You can specify as many secondary roll-ups as you want, as long as the roll-ups are all in one record.

Define the field type for the shared member as LEVEL. Then enter the level number. To create a shared member of the same generation, set the level number of the secondary roll-up to have the same number of levels as the primary roll-up. While processing the data source, Essbase creates a parent at the specified level and inserts the shared members under it.

For example, to create the shared 100-20 (Diet Cola), 200-20 (Diet Root Beer), 300-20 (Diet Cream Soda), and 400-20 (Fruit Soda) members in the Sample Basic database, use the sample file, `SHLEV.TXT`, and set up the rules file so that the fields look like `SHLEV.RUL` shown in Figure 252.

Figure 252: Sample Level Shared Member Rules File



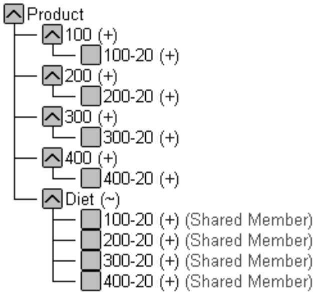
Data Prep Editor - Localhost:Sample:Basic:SHLEV

```
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
5			

This builds the following tree:

Figure 253: Sample Level Shared Member Rules Tree

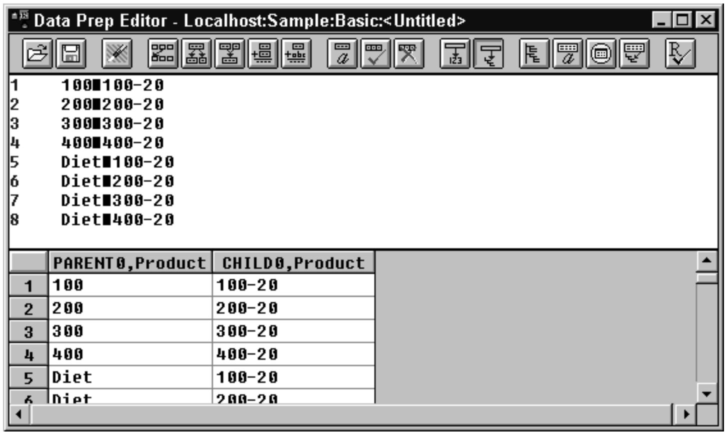


```
Product
├── 100 (+)
│   └── 100-20 (+)
├── 200 (+)
│   └── 200-20 (+)
├── 300 (+)
│   └── 300-20 (+)
├── 400 (+)
│   └── 400-20 (+)
└── Diet (~)
    ├── 100-20 (+) (Shared Member)
    ├── 200-20 (+) (Shared Member)
    ├── 300-20 (+) (Shared Member)
    └── 400-20 (+) (Shared Member)
```

Creating Shared Members Using Parent-Child References

To create shared members at the same generation using the parent-child references build method, define the PARENT and CHILD field types. Make sure that Do Not Share is cleared in the Dimension Build Settings page of the Dimension Build Settings dialog box. When Do Not Share is cleared, Essbase automatically creates duplicate members under a new parent as shared members.

Figure 254: Sample Parent-Child Shared Members Rules File

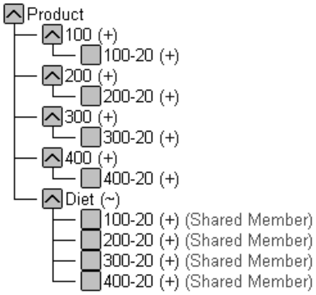


```
100■100-20
200■200-20
300■300-20
400■400-20
Diet■100-20
Diet■200-20
Diet■300-20
Diet■400-20
```

	PARENT0,Product	CHILD0,Product
1	100	100-20
2	200	200-20
3	300	300-20
4	400	400-20
5	Diet	100-20
6	Diet	200-20

This builds the following tree:

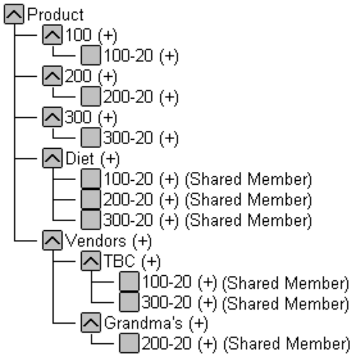
Figure 255: Sample Parent-Child Shared Member Rules Tree


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)

Sharing Members at Different Generations

Sometimes you want shared members to roll up into parents at different generations in the outline. In Figure 256, for example, the shared members roll up into parents at generation 2 and generation 3 in the outline. This outline assumes that The Beverage Company (TBC) buys some of its beverages from outside vendors. In this case, it buys 200-20 (Diet Root Beer) from a vendor named Grandma's.

Figure 256: Members Shared at Different Generations


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)

To share members across parents at different generations in the outline, use the following build methods:

- "Creating Shared Members Using Level References" on page 523
- "Creating Shared Members Using Parent-Child References" on page 524

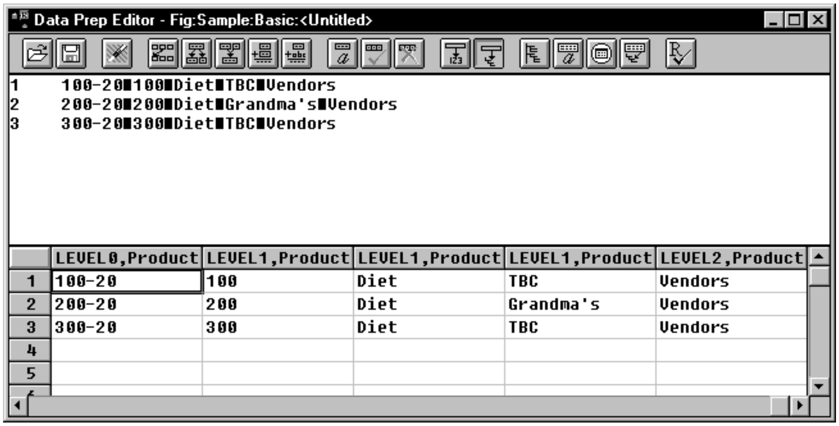
Creating Shared Members Using Level References

To create shared members at different generations using the level references build method, first make sure that both primary and secondary roll-ups are specified in one record. You can specify as many secondary roll-ups as you want, as long as the roll-ups are all in one record.

Define the field type for the shared member as LEVEL. Then enter the level number. While processing the data source, Essbase creates a parent at the specified level and inserts the shared members under it.

For example, to share the products 100-20, 200-20, and 300-20 with a parent called Diet and two parents called TBC (The Beverage Company) and Grandma's, use the sample data file and rules file in Figure 257.

Figure 257: Level References Sample Rules File for Shared Members at Different Generations



```
1     100-20■100■Diet■TBC■Vendors
2     200-20■200■Diet■Grandma's■Vendors
3     300-20■300■Diet■TBC■Vendors
```

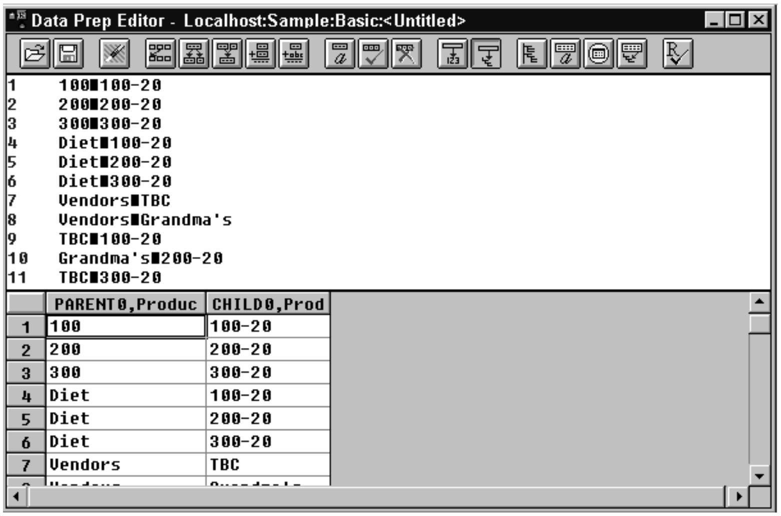
	LEVEL0,Product	LEVEL1,Product	LEVEL1,Product	LEVEL1,Product	LEVEL2,Product
1	100-20	100	Diet	TBC	Vendors
2	200-20	200	Diet	Grandma's	Vendors
3	300-20	300	Diet	TBC	Vendors
4					
5					

This builds the tree in Figure 256 on page 522.

Creating Shared Members Using Parent-Child References

To create shared members at the same generation using the parent-child references build type, define the PARENT and CHILD field types. Make sure that Do Not Share is not selected in the Dimension Build Settings page of the Dimension Build Settings dialog box so that Essbase creates duplicate members as shared when it encounters them under a new parent.

Figure 258: Parent-Child References Sample Rules File for Shared Members at Different Generations



Data Prep Editor - Localhost:Sample:Basic:<Untitled>

```
100■100-20
200■200-20
300■300-20
Diet■100-20
Diet■200-20
Diet■300-20
Vendors■TBC
Vendors■Grandma's
TBC■100-20
Grandma's■200-20
TBC■300-20
```

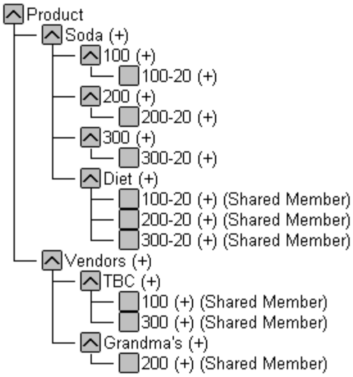
	PARENT0,Produc	CHILD0,Prod
1	100	100-20
2	200	200-20
3	300	300-20
4	Diet	100-20
5	Diet	200-20
6	Diet	300-20
7	Vendors	TBC

This builds the tree in Figure 256 on page 522.

Sharing Members with Branches

Sometimes you want to share non-leaf members (members that are not at the lowest generation). In Figure 259 for example, 100, 200, and 300 are shared by TBC and Grandma’s. This outline assumes that TBC (The Beverage Company) buys some of its product lines from outside vendors. In this case, it buys 200 (all root beer) from a vendor named Grandma’s.

Figure 259: Members with Branches Shared at Different Generations



```
Product
├── Soda (+)
│   ├── 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 (+) (Shared Member)
    │   └── 300 (+) (Shared Member)
    └── Grandma's (+)
        └── 200 (+) (Shared Member)
```

To share members in the outline with branches below them, use the following build methods:

- “Creating Shared Members Using Level References” on page 525
- “Creating Shared Members Using Parent-Child References” on page 527

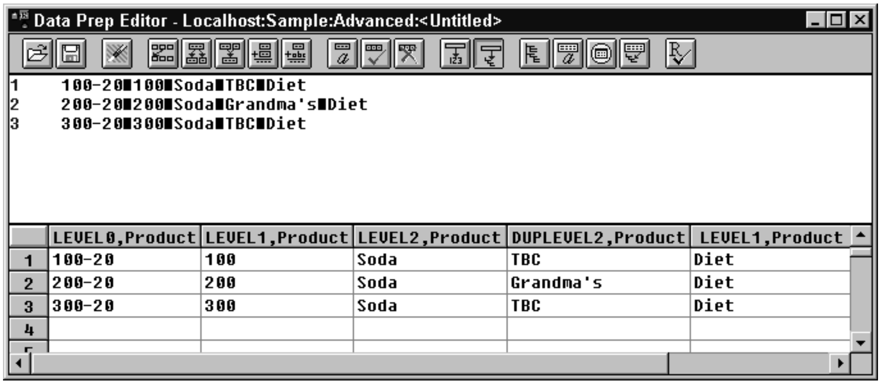
Creating Shared Members Using Level References

To create shared non-leaf members using the level references build method, first make sure that both primary and secondary roll-ups are specified in one record. You can specify as many secondary roll-ups as you want, as long as the roll-ups are all in one record.

Define the field type for the shared member’s parent as duplicate level (DUPLEVEL). Then enter the level number. To create a shared member of the same generation, set the level number of the secondary roll-up to have the same number of levels as the primary roll-up. While processing the data source, Essbase creates a parent at the specified level and inserts the shared members under it.

For example, to share the product lines 100, 200, and 300 with a parent called Soda and two parents called TBC and Grandma’s, use the sample data file and rules file shown in Figure 260. This data source and rules file only work if the Diet, TBC, and Grandma’s members exist in the outline. The DUPLEVEL field is always created as a child of the dimension (that is, at generation 2), unless the named level field already exists in the outline.

Figure 260: Level References Sample Rules File for Shared Members at Different Generations with Branches



```
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
4					

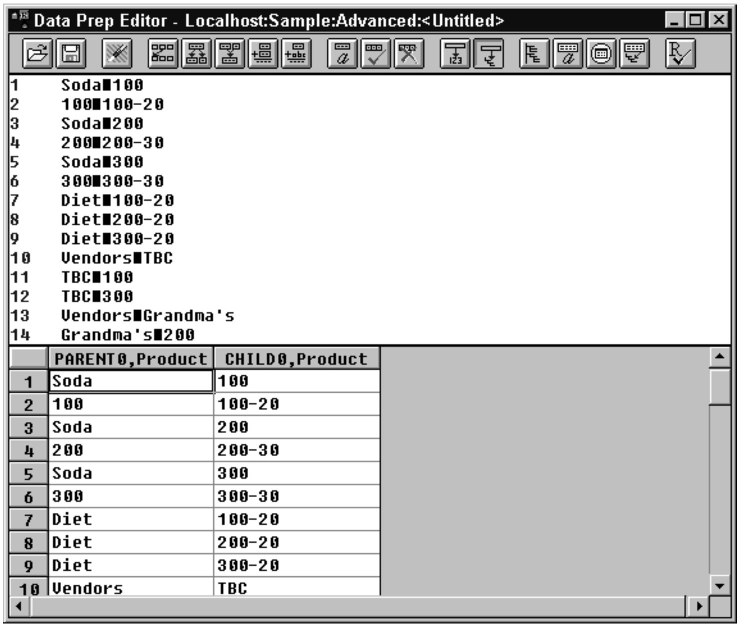
This builds the tree in Figure 259 on page 525.

Creating Shared Members Using Parent-Child References

To create shared members at the same generation using the parent-child references build type, define the PARENT and CHILD field types. Make sure that Do Not Share is not selected in the Dimension Build Settings page of the Dimension Build Settings dialog box so that Essbase creates duplicate members as shared when it encounters them under a new parent.

The parent-child references build method is the most versatile for creating shared members. It does not have any restrictions on the position of the shared members in the outline, unlike the generation references and level references build methods.

Figure 261: Parent-Child Sample Rules File for Shared Members at Different Generations with Branches



```
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
```

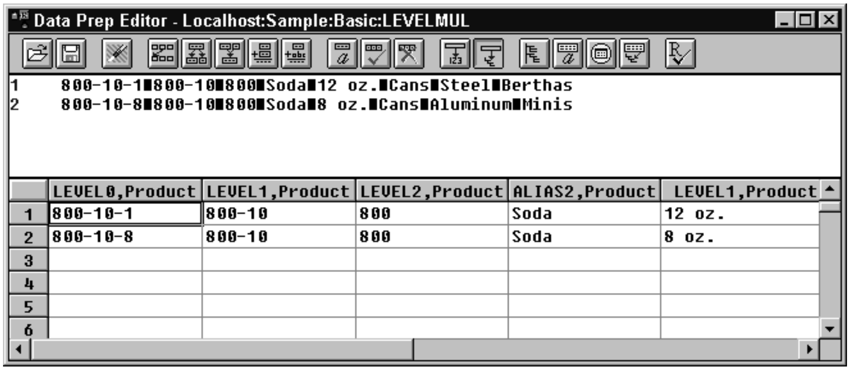
	PARENT0,Product	CHILD0,Product
1	Soda	100
2	100	100-20
3	Soda	200
4	200	200-30
5	Soda	300
6	300	300-30
7	Diet	100-20
8	Diet	200-20
9	Diet	300-20
10	Vendors	TBC

This builds the tree in Figure 259 on page 525.

Building Multiple Roll-Ups Using Level References

To enable retrieving totals from multiple perspectives, you can also put shared members at different levels in the outline. Use the level references build method. The rules file, `LEVELMUL.RUL`, in Figure 262 specifies an example of build instructions for levels in the Product dimension.

Figure 262: Rules File Fields Set to Build Multiple Roll-Ups Using Level References

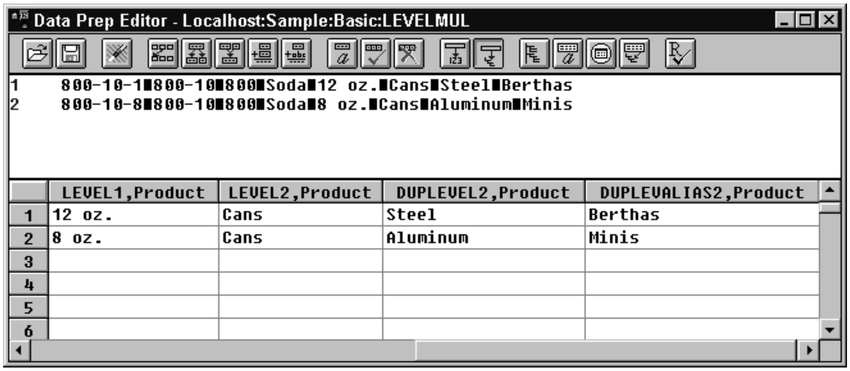


```
1     800-10-1■800-10■800■Soda■12 oz.■Cans■Steel■Berthas
2     800-10-8■800-10■800■Soda■8 oz.■Cans■Aluminum■Minis
```

	LEVEL0,Product	LEVEL1,Product	LEVEL2,Product	ALIAS2,Product	LEVEL1,Product
1	800-10-1	800-10	800	Soda	12 oz.
2	800-10-8	800-10	800	Soda	8 oz.
3					
4					
5					
6					

Because the record is so long, this second graphic shows the rules file after it has been scrolled to the right to show the extra members:

Figure 263: Scrolled Window

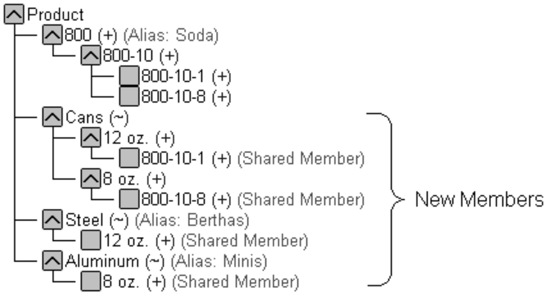


```
1     800-10-1■800-10■800■Soda■12 oz.■Cans■Steel■Berthas
2     800-10-8■800-10■800■Soda■8 oz.■Cans■Aluminum■Minis
```

	LEVEL1,Product	LEVEL2,Product	DUPLEVEL2,Product	DUPLEVALIAS2,Product
1	12 oz.	Cans	Steel	Berthas
2	8 oz.	Cans	Aluminum	Minis
3				
4				
5				
6				

When you run the dimension build using the data in Figure 262 on page 528, Essbase builds the following member tree:

Figure 264: Multiple Roll-Ups


Product
800 (+) (Alias: Soda)
800-10 (+)
800-10-1 (+)
800-10-8 (+)
Cans (~)
12 oz. (+)
800-10-1 (+) (Shared Member)
8 oz. (+)
800-10-8 (+) (Shared Member)
Steel (~) (Alias: Berthas)
12 oz. (+) (Shared Member)
Aluminum (~) (Alias: Minis)
8 oz. (+) (Shared Member)
New Members

This example enables analysis not only by package type (Cans), but also by packaging material; for example, analysis comparing sales of aluminum and steel cans.

Since Product is a sparse dimension, you can use an alternative outline design to enable retrieval of the same information. Consider creating a multilevel attribute dimension for package type with Steel and Aluminum as level 0 members under Can. See “Analyzing Database Design” on page 110.

Creating Shared Roll-Ups from Multiple Data Sources

In many situations, the data for a dimension is in two or more data sources. If you are building dimensions from more than one data source and want to create multiple roll-ups, use this procedure:

1. Load the first data source using the most appropriate build method.
2. Load all other data sources using the parent-child references build method. Make sure that “Do Not Share” is not selected so that Essbase creates duplicate members as shared when it encounters them under a new parent.

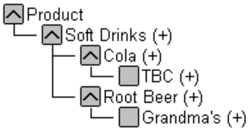
For example, assume this Product data:

Figure 265: Soft Drinks Data Source

```
"Soft Drinks"   Cola
"Soft Drinks"   "Root Beer"
Cola            TBC
"Root Beer"     Grandma's
```

Essbase takes the data from Figure 265 and builds this tree:

Figure 266: Soft Drinks Tree



```
Product
└─ Soft Drinks (+)
   ├─ Cola (+)
   │  └─ TBC (+)
   └─ Root Beer (+)
      └─ Grandma's (+)
```

A second data source relates the products to the vendors:

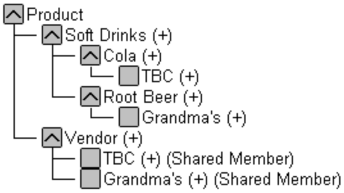
Figure 267: Second Shared Roll-Ups Data Source

```
Vendor   TBC
Vendor   Grandma's
```

To create a member tree where the products are shared by the Vendor member, load the second data source using the parent-child build method. Make sure that "Do Not Share" is not selected so that Essbase creates duplicate members as shared when it encounters them under a new parent.

Essbase builds the following tree:

Figure 268: Shared Roll-Ups Tree



```
Product
├─ Soft Drinks (+)
│  ├─ Cola (+)
│  │  └─ TBC (+)
│  └─ Root Beer (+)
│     └─ Grandma's (+)
└─ Vendor (+)
   ├─ TBC (+) (Shared Member)
   └─ Grandma's (+) (Shared Member)
```

Security and Multiple-User Considerations

Essbase supports concurrent multiple users reading and updating the database. This means that users can use the database while you are dynamically building dimensions, loading data, or calculating the database.

- Security Issues: The security system prevents unauthorized users from changing the database. Only users with write access to a database can add dimensions to the database. Write access can be provided as global write access or by using filters.
- Multiple User Issues: You cannot build dimensions while other users are reading or writing to the database. After you build dimensions, Essbase restructures the outline and locks the database for the duration of the restructure operation.

Note: For information on how to see which user has a lock on a particular block, see Chapter 15, "Managing Security for Users and Applications."

Chapter 19

Building Dimensions Using a Rules File

This chapter describes how to build dimensions using a rules file in Essbase. For background information on dynamic dimension building, see Chapter 18, “Introducing Dynamic Dimension Building.”

About Dimensions and Rules Files

Before defining dimensions, you must associate the rules file with the outline.

You can build dimensions dynamically in the following ways:

- Create a rules file using the Dimension Build Settings dialog box in Data Prep Editor. This process includes defining new dimensions, specifying changes to existing dimensions, setting global options, and validating the dimension build rules.
- Create a dynamic reference in a rules file to a record in the data source that defines each field. This method enables you to use a single rules file with several different source files. To use this method, you must manipulate the date source to include this header record.

Validate the rules file and perform the dimension build. If you have problems validating the rules file or using it to build dimensions, this chapter also discusses debugging tips.

Tip: You can dynamically build dimensions outside Application Manager, using ESSCMD commands BUILDDIM or INCBUILDDIMI. For more information, see the *Technical Reference* in the `docs` directory.

Step 1: Defining Dimensions

➤ To define a dimension:

1. Name new dimensions and specify whether a standard dimension comes from the outline or a rules file.
2. Set the build type and the properties for new dimensions or change the properties of existing dimensions.

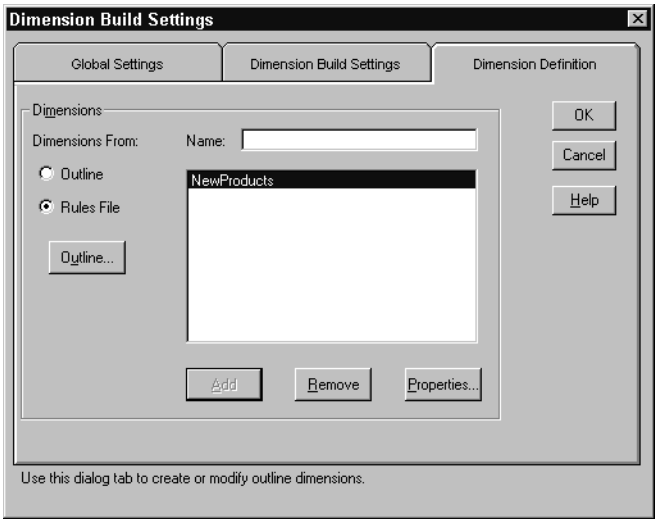
Naming New Dimensions

The processes for naming new standard dimensions and new attribute dimensions are different.

➤ To name a new dimension:

1. Select the application and database in the Application Desktop window in Essbase Application Manager.
2. Click the Data Load Rules button, .
3. Click New to open Data Prep Editor with a new rules file or Open to open an existing rules file.
4. Select View > Dimension Building fields or click the Dimension Build button, , to make sure that Data Prep Editor displays dimension building fields and not data load fields.
5. Select Options > Dimension Build Settings to open the Dimension Build Settings dialog box. Select the Dimension Definition tab.

Figure 269: Dimension Definition Page



Dimension Build Settings

Dimensions From: ○ Outline ● Rules File

Name:

NewProducts

Use this dialog tab to create or modify outline dimensions.

When Rules File is selected, the large list box displays the names of all new standard dimensions defined in the rules file. To see new attributes dimensions, you must follow the instructions in "Naming New Attribute Dimensions" on page 537.

When Outline is selected, the large list box displays the names of all dimensions in the existing outline.

6. If the list of dimensions is empty, make sure that the rules file is associated with an outline. Click the Outline button, then open the outline file.

   Note: If the outline is empty, the list of dimensions remains empty, even when properly associated with the outline.

Naming a New Standard Dimension

If you are naming a new attribute dimension to be associated with an existing standard dimension, proceed to "Naming New Attribute Dimensions" on page 537.

➤ To name a new standard dimension:

1. Select Rules File to indicate that the dimension is defined in the rules file.
2. Enter the name of the new dimension, such as NewProducts. See "Rules for Naming Applications and Databases" on page 154.
3. Click Add to add it to the end of the outline.

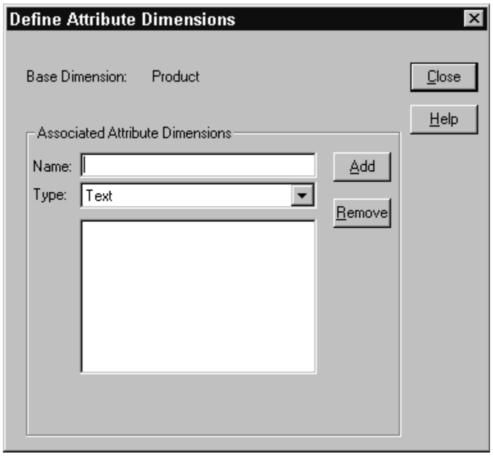
   If you are not also defining associated attribute dimensions, continue with setting the dimension properties. See "Setting Dimension Properties" on page 538.

Naming New Attribute Dimensions

➤ To create a new attribute dimension, the base dimension must already be defined in either the outline or the rules file. The base dimension must be a sparse dimension.

1. Select the base dimension in the list box and click Properties to open the Dimension Properties dialog box.
   - If the base dimension is defined in the outline, on the Dimension Definition page of the Dimension Build Settings dialog box, select Outline to display the base dimension name in the list box.
   - If the base dimension is defined in the rules file, on the Dimension Definition page of the Dimension Build Settings dialog box, select Rules to display the base dimension name in the list box
2. On the Dimension Properties page, click Attribute Dimensions to display the Define Attribute Dimensions dialog box.

Figure 270: Defining Attribute Dimensions



Define Attribute Dimensions

Base Dimension: Product

Associated Attribute Dimensions

Name:

Type: Text

The large list box lists the names of new attribute dimensions associated with the base dimension in the current rules file.

3. For each new attribute dimension that you define to associate with the base dimension:
   - Type the attribute dimension name; for example, Population.
   - Select the attribute dimension type; for example, Numeric.
   - Click Add.
4. To remove an attribute dimension from the list box:
   - Select the attribute dimension name.
   - Click Remove.
5. After adding all new attribute dimension names and specifying their types, click OK to close the dialog box and return to the Dimension Properties dialog box.
6. Click OK to close the Dimension Properties dialog box.

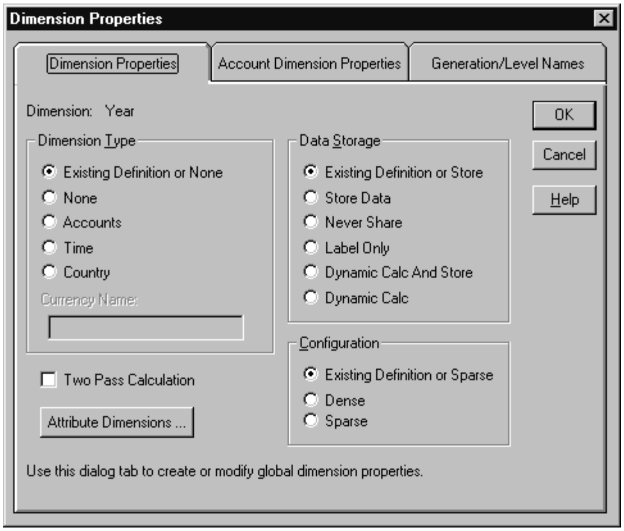
Setting Dimension Properties

➤ To set the properties of a standard dimension:

1. If the Dimension Build Settings dialog box is not open, select Options > Dimension Build Settings to open it.
   - Select the Dimension Definition page.
   - If the dimension exists in the outline, click the Outline option to display the names of the dimensions in the outline. If the list box is empty, click the Outline button to associate the dimension build rules file with an outline and display the dimensions in the list box.
   - If the dimension is new, click Rules File. The list box displays the new dimensions that you named.
2. Click the dimension name in the list box.

3. Click Properties to open the Dimension Properties dialog box.

Figure 271: Dimension Properties Page



Dimension: Year

Dimension Type
- ● Existing Definition or None
- ○ None
- ○ Accounts
- ○ Time
- ○ Country

Currency Name:

□ Two Pass Calculation

Data Storage
- ● Existing Definition or Store
- ○ Store Data
- ○ Never Share
- ○ Label Only
- ○ Dynamic Calc And Store
- ○ Dynamic Calc

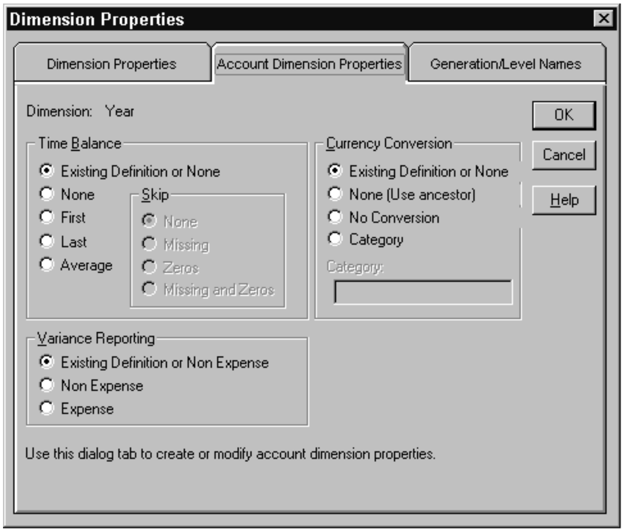
Configuration
- ● Existing Definition or Sparse
- ○ Dense
- ○ Sparse

Use this dialog tab to create or modify global dimension properties.

- Set the Dimension Type.
- Select the Data Storage property.
- Select a Dense or Sparse configuration. Base dimensions must be set as sparse.
- If you are not sure what settings to use, click Help.

4. For an accounts dimension, click the Account Dimension Properties tab.
   - Set the accounts dimension properties and click OK.
   - If you are not sure what settings to use, click Help.

Figure 272: Account Dimension Properties Page



Dimension Properties

Dimension Properties | Account Dimension Properties | Generation/Level Names

Dimension: Year

Time Balance
● Existing Definition or None
○ None
○ First
○ Last
○ Average

Skip
● None
○ Missing
○ Zeros
○ Missing and Zeros

Currency Conversion
● Existing Definition or None
○ None (Use ancestor)
○ No Conversion
○ Category

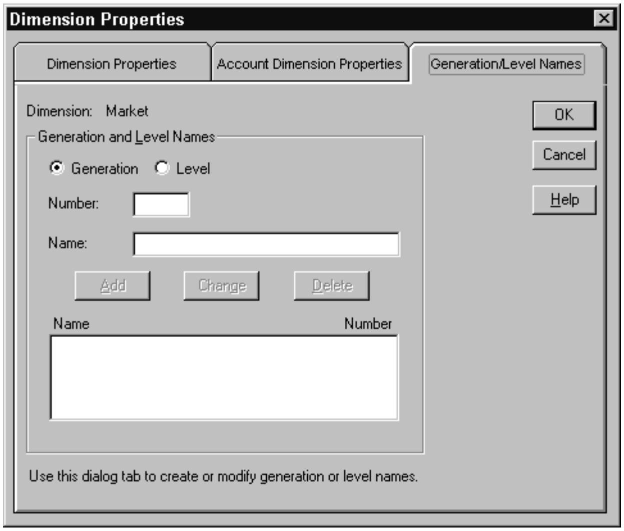
Category:

Variance Reporting
● Existing Definition or Non Expense
○ Non Expense
○ Expense

Use this dialog tab to create or modify account dimension properties.

5. To name the generations and levels in the current dimension, select the Generation/Level Names tab.
   - Set the generation/level names and click OK.
   - If you are not sure which settings to use, click Help.

Figure 273: Generation/Level Names Page



Dimension: Market

◉ Generation ○ Level

Use this dialog tab to create or modify generation or level names.

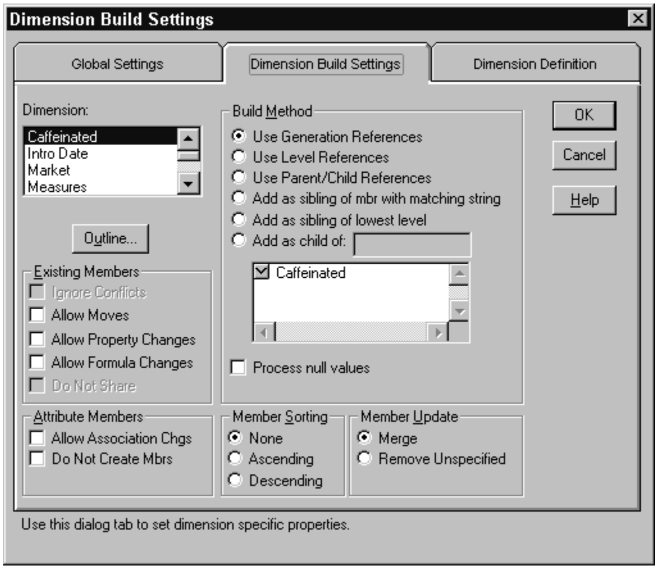
Note: If you define a name for a generation or level that already exists in the outline, the new name overwrites the existing name.

Step 2: Choosing the Build Method

➤ To specify the build method for the rules file:

1. If it is not showing, click the Dimension Build Settings tab of the Dimension Build Settings dialog box.

Figure 274: Dimension Build Settings Page



Dimension Build Settings

Global Settings | Dimension Build Settings | Dimension Definition

Dimension:
Caffeinated
Intro Date
Market
Measures

Outline...

Existing Members
□ Ignore Conflicts
□ Allow Moves
□ Allow Property Changes
□ Allow Formula Changes
□ Do Not Share

Attribute Members
□ Allow Association Chgs
□ Do Not Create Mbrs

Build Method
● Use Generation References
○ Use Level References
○ Use Parent/Child References
○ Add as sibling of mbr with matching string
○ Add as sibling of lowest level
○ Add as child of:
☑ Caffeinated
□ Process null values

Member Sorting
● None
○ Ascending
○ Descending

Member Update
● Merge
○ Remove Unspecified

Use this dialog tab to set dimension specific properties.

2. Select the dimension from the Dimension list. If the list is empty, click Outline to associate the dimension build rules file with an outline. The list includes new standard dimensions defined in the rules file.

3. Select the build method from the Build Method box. If you are not sure which method to use, see "Introduction to Build Methods" on page 489.

The following build methods require that you specify additional information:

Build Method	What to Specify
Use generation references	Whether to use null processing. See "Null Processing with Generation References" on page 493.
Use level references	Whether to use null processing. See "Null Processing with Level References" on page 496.
Add as child of	The parent to which the new members are added.

Step 3: Specifying Changes to Dimensions

➤ To specify the changes that you want Essbase to make to dimensions in the outline:

1. On the Dimension Build Settings page of the Dimension Build Settings dialog box, select the dimension from the Dimension list.
2. Select the types of changes you will allow to existing members of the selected dimension in the outline from Figure 28 on page 544.
3. To sort the members of a dimension after building it, select either Ascending (A to Z) or Descending order (Z to A). To leave the members unsorted, select None.
4. By default, Essbase merges new members found in the data source into the dimension. To remove existing members if Essbase does not encounter them in the data source, select Remove Unspecified.
5. If the rules file works with attribute source data, select the attribute dimension and select the types of changes to allow from Figure 29 on page 545.
6. Click OK.

Table 28: Existing Member Changes Allowed

Type of Change	Option to Select
Ignore member names that already exist in the outline under different dimensions.	Ignore Conflicts (valid only with the Add as... build methods)
Allow a member and its descendants to be moved to a new parent in the same dimension. Essbase cannot move the member to itself or to another member below it in the tree. Use Allow Moves to reorganize primary members in the outline to match the data source. To reorganize shared members, use Outline Editor.	Allow Moves
Allow changes to the properties, UDAs, and aliases of existing members.	Allow Property Changes
Allow changes to the formulas of existing members. To include quotation marks in a formula, precede the marks with a backslash. For example, `\"Other Variable\" + Tax`.	Allow Formula Changes
Reject records that specify a new parent for an existing member. If you do not select this box, each time a member is repeated with another parent, it is created as a shared member.	Do Not Share (valid only with the parent/child references build method)

Table 29: Attribute Dimensions Changes Allowed

Type of Change	Option to Select
Allow an existing association to be changed. For example, if the source data shows the Ounces attribute for product 100-10 as 8 and the existing outline shows the attribute as 12, Essbase associates product 100-10 with the attribute 8.	Allow Association Chgs
Prevent creation of new members of the attribute dimension. For example, if the source data shows the Ounces attribute for product 100-10 as 8 and the Ounces attribute dimension does not include the member 8, Essbase does not add the member 8 to the Ounces dimension.	Do Not Create Mbrs

Step 4a: Setting Rules File Field Types

Each field defines a source data column that becomes a member in the outline, a property of a member, or information that helps to define an association; for example, associating an alias with a member or an attribute with a base dimension member.

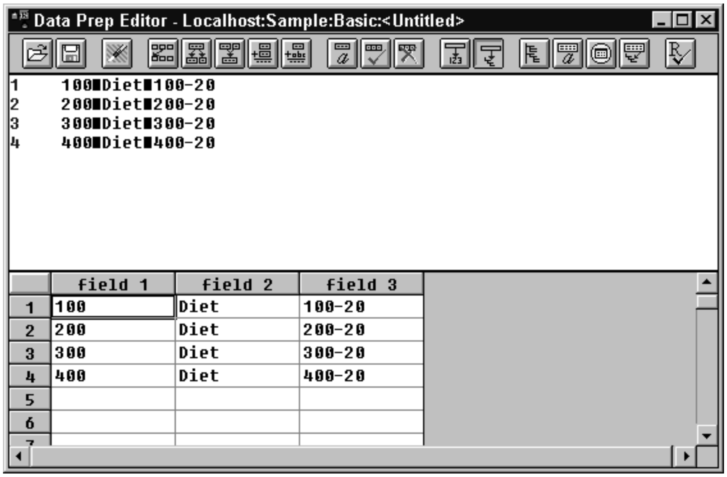
Setting the field type tells Essbase what kind of field to expect, such as a generation field or an alias field. At the same time, you specify the dimension for the member and its generation or level number.

To set the field types in Data Prep Editor, you must set Data Prep Editor to dimension building mode and you must open the data source.

- Select View > Dimension Building Fields or click the Dimension Build button, , to ensure that Data Prep Editor is in dimension building mode.
- Select File > Open Data File or File > Open SQL, whichever is appropriate, to specify the source location and open the data source. For more details, see "Opening a Data Source" on page 592.

As shown in Figure 275, Data Prep Editor displays the data source in the upper half of the window and rules fields in the lower half of the window.

Figure 275: Data Prep Editor



```
1     100■Diet■100-20
2     200■Diet■200-20
3     300■Diet■300-20
4     400■Diet■400-20
```

	field 1	field 2	field 3
1	100	Diet	100-20
2	200	Diet	200-20
3	300	Diet	300-20
4	400	Diet	400-20
5			
6			
7			

If desired, you can customize Data Prep Editor. For example, you can hide the raw data or maximize the window, to set a better view of the rules fields. See “Customizing the Data Prep Editor” on page 592.

Step 4b: Setting Rules File Field Information

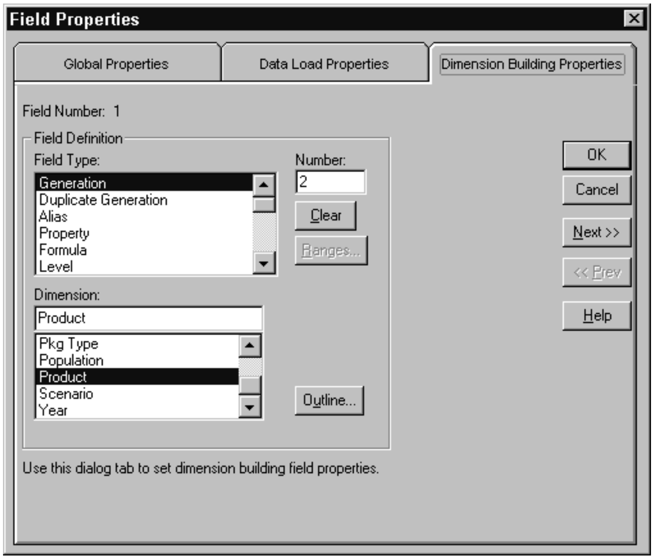
This section describes how to define the field type for each field in the rules file. Many of the details for each field depend on the nature of the data source and the build method to be used. See Chapter 18, “Introducing Dynamic Dimension Building,” for explanations and examples of how the fields should be defined to build or modify outlines for various situations.

To map the rules file fields to the source data, you may need to manipulate how the data is used. For example, you may need to create a member name from two source data fields, or you may need to ignore a source data field. For information on mapping and manipulating fields, see “Manipulating Fields Using a Rules File” on page 613.

➤ To set field types:

1. In Data Prep Editor window, select the field for which you want the field type set.
2. Select Field > Properties or click the Field Properties button, , to open the Field Properties dialog box.
3. Select the Dimension Building Properties tab.

Figure 276: Dimension Building Properties Page



Field Number: 1

Field Definition

Field Type: Generation

Duplicate Generation
Alias
Property
Formula
Level

Number: 2

Dimension: Product

Pkg Type
Population
Product
Scenario
Year

Use this dialog tab to set dimension building field properties.

4. If the Dimension list is empty, click the Outline button to associate the rules file with an outline or check the rules file to be sure that at least one dimension is defined.
5. Select the field type from the Field Type list box. Table 30 on page 551 lists valid field types for each build method from.
6. Enter the generation or level number in the Number text box.
   - If Field Type is the name of an attribute dimension, the generation or level number must correspond to the generation or level of the associated base member in the outline. For example, the 3 in OUNCES3,PRODUCT

shows that the data values in the field are members of the Ounces attribute dimension associated with the third generation member of the Product dimension in the same source data record.

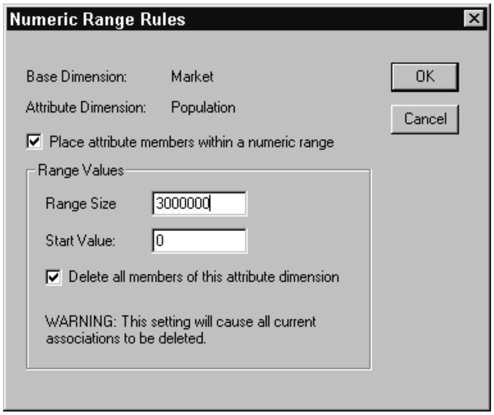
- If Field Type is parent or child, enter 0 (zero) in the Number text box.
- If Field Type is not the name of an attribute dimension nor parent nor child, the generation or level number must correspond to the generation or level of the member in the outline for which the field provides values. For example, the 3 in GEN3,PRODUCT shows that the data values in the field are third generation members of the Product dimension. The 2 in ALIAS2,POPULATION shows that the data values in the field are associated with the second generation or level member of the Population dimension.

As described in Table 31 on page 552, how you assign and sequence fields in the rules file can vary depending if the number is for a level or generation build.

7. In the Dimension box, enter the dimension for which the field provides values, or select the dimension name from the Dimension list. For attribute associations, select the base dimension from the Dimension list.
8. To define ranges for members of the numeric attribute dimension specified in the Field Type list box, click Ranges. Otherwise, proceed to Step 14.

   Data Prep editor displays the Numeric Range Rules dialog box.

Figure 277: Defining Range Size for Numeric Attribute Dimensions



Numeric Range Rules

Base Dimension: Market

Attribute Dimension: Population

☑ Place attribute members within a numeric range

Range Values

Range Size: 3000000

Start Value: 0

☑ Delete all members of this attribute dimension

WARNING: This setting will cause all current associations to be deleted.

For information about using attribute dimension members to represent ranges of base member values, see "Assigning Member Names to Ranges of Values" on page 247 and "Working With Numeric Ranges" on page 510.

9. To enable automatic range building for the specified numeric attribute dimension members, click Place attribute members within a member range.

10. In the Range Size text box, enter the numeric value of the range size for each member; for example, 3000000.

11. In the Start Value text box, enter a numeric value for the name of one member of the numeric attribute dimension.

    This member becomes the pivot point upon which Essbase uses the range size to create other range members above and below the specified start value. Essbase uses the start value only when it builds an attribute dimension and associates its members to members of a base dimension in the same build operation.

    Assume, for example, that you set the range size as 10 and the start value as 5. As it processes the data source, Essbase may build the following members of the numeric attribute dimension: 5-, 5, 15, 25, and so on.

    Note: Although you enter the names of negative numeric attributes with the minus sign before the number, for example -5, Essbase creates the member name with the minus sign after the number, for example 5-.

    The start value can be a positive or negative whole number, zero, or a decimal value. To enter a negative number, type the minus sign in front of the number; for example, -15.

    Tip: If you want a numeric range member named 0, specify 0 as the start value. For example, if the range size is 3000000, Essbase builds the following members of the numeric dimension: 0, 3000000, 6000000, and so on.

12. To remove and re-create all members of the specified attribute dimension and reassociate them with the members of the base dimension, select “Delete all members of this attribute dimension.”

CAUTION: All base member associations with the attribute dimension are lost. To enable the dimension build to re-create the associations, the source data must include all members of the base dimension that you want to be associated with members of this attribute dimension.

13. Click OK to close the Numeric Range Rules dialog box.
14. To move to the next field, click Next.
15. When you are finished setting the field types, click OK.

    If needed, move the fields to the required locations. The required location of fields depends on the build method and what you want to achieve through the dimension build operation. For specific requirements, see the sixth step under “Step 4b: Setting Rules File Field Information” on page 546 and “Working with Multilevel Attribute Dimensions” on page 507.

Table 30: Defining Field Types in Rules Files

Field Type	What the Field Contains	Valid Build Methods
Alias	An alias	Generation, level, and parent/child references
Property	A member property	
Formula	A formula	
Currency name	A currency name	
Currency category	A currency category	
UDA	A UDA (user-defined attribute)	
Attribute Parent	In an attribute dimension, the name of the parent member for the attribute member in the next field	
The name of a specific attribute dimension	A member of the specified attribute dimension. This member will be associated with the specified generation or level of the selected base dimension.	
Generation	Name of a member in a generation	Generation references
Duplicate generation	A member that is shared by more than one parent	
Duplicate generation alias	Alias for the new parent of the shared member as the member is created	
Level	Name of member in a level	Level references
Duplicate level	Name of member that has duplicate parents; that is, a member that is shared by more than one parent	
Duplicate level alias	Alias for the new parent of the shared member as the member is created	
Parent	Name of a parent	Parent/child reference
Child	Name of a child	

Table 31: Rules Files Fields and Levels or Generations

Type of Number	Rules for Assigning
Level	• Put DUPLEVEL fields immediately after LEVEL fields. • Put DUPLEVELALIAS fields immediately after the DUPLEVEL fields. • Each record must contain a level 0 member. If a level 0 member is repeated on a new record with a different parent, Essbase rejects the record unless you select Allow Moves. See "Step 3: Specifying Changes to Dimensions" on page 543. • Group level fields sequentially within a dimension. • Put the fields for each roll-up in sequential order. • Use a single record to describe the primary and secondary roll-ups. • Put attribute association fields after the base field with which they are associated and specify the level number of the associated base dimension member; for example: `LEVEL3, PRODUCT   OUNCES3,PRODUCT   LEVEL2,PRODUCT`

Generation	• If GEN numbers don't start at 2, the first member in the specified generation must exist in the outline.
	• GEN numbers must form a contiguous range. For example, if GEN 3 and GEN 5 exist, you must also define GEN 4.
	• Put DUPGEN fields immediately after GEN fields.
	• Put DUPGENALIAS fields immediately after DUPGEN fields.
	• Group GEN fields sequentially within a dimension; for example:
	`GEN2,PRODUCT  GEN3,PRODUCT  GEN4,PRODUCT`
	• Put attribute association fields after the base field with which they are associated and specify the generation number of the associated base dimension member; for example:
	`GEN2, PRODUCT   GEN3,PRODUCT   OUNCES3,PRODUCT`

Step 5: Setting Global Options

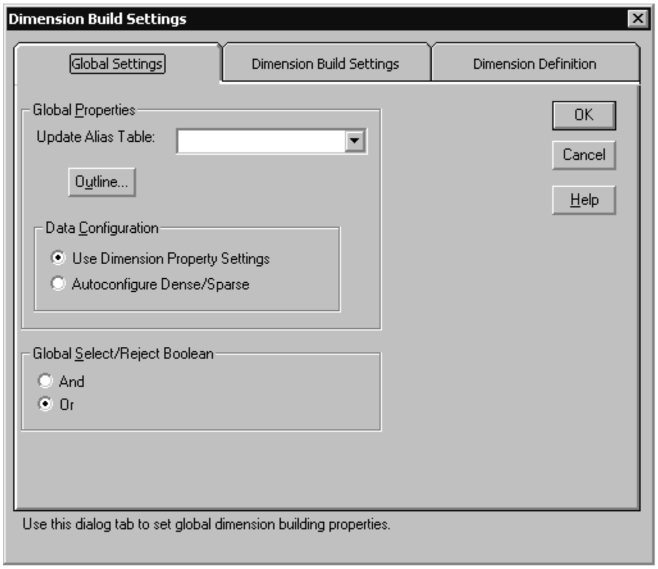
Global options affect *all* dimensions in the rules file. Generally, you build one dimension per rules file. The global build options include:

- Whether to configure dimensions as dense or sparse automatically or to use the dense/sparse configuration defined in the outline or rules file
- Which alias table to update
- How to combine field select or reject criteria between fields

➤ To set global build options:

1. Click the Global Settings tab of the Dimension Build Settings dialog box.

Figure 278: Global Settings Page



Dimension Build Settings

Global Properties

Update Alias Table:

Data Configuration

● Use Dimension Property Settings
○ Autoconfigure Dense/Sparse

Global Select/Reject Boolean

○ And
● Or

Use this dialog tab to set global dimension building properties.

2. Select which alias table to update with new aliases from the data source. If you do not specify an alias table, Essbase updates the Default table.

3. Select either:
   - The Use Dimension Property Settings to keep using the current data configuration or use the one specified in the rules file.
   - The Autoconfigure Dense/Sparse to let Essbase determine the data configuration automatically. For more information on dense and sparse dimensions, see "Sparse and Dense Dimensions" on page 80.

   When you change the configuration settings, Essbase restructures the database.
4. Select either And or Or to determine how Essbase combines select and reject criteria. For more information, see "Defining Multiple Select and Reject Criteria" on page 605.
5. Click **OK** to save the changes.

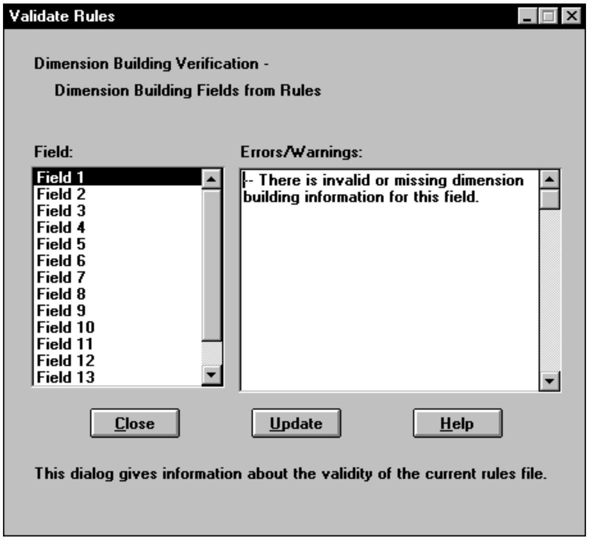
Step 6: Validating Dimension Build Rules

➤ To validate a rules file, make sure that the rules file is open and associated with an outline. If you're building dimensions by altering the data source (using dynamic references), open the data source.

1. Select View > Dimension Building Fields or click the Dimension Build button, , to make sure that Data Prep Editor is in dimension building mode.

2. Select Options > Validate or click the Validate Rules button, , to validate the rules file against the outline. When Essbase finishes the validation, the Validate Rules dialog box displays.

Figure 279: Validate Rules Dialog Box



Dimension Building Verification -
Dimension Building Fields from Rules

Field 1
Field 2
Field 3
Field 4
Field 5
Field 6
Field 7
Field 8
Field 9
Field 10
Field 11
Field 12
Field 13

-- There is invalid or missing dimension building information for this field.

This dialog gives information about the validity of the current rules file.

If the rules file is correct, you can use it perform a dimension build. For more information, see “Performing Dimension Builds” on page 559.

3. If the rules file is not valid, fix it before using it to build dimensions. Go to the invalid fields listed in the Validate Rules dialog. In Figure 279, for example, Field 1 is invalid.

   Make sure that the field name is valid.

   - Are the reference numbers sequential?
   - Are there repeated generations?
   - Is the field type valid for the build method?
   - Are the fields in correct order?
   - Does the child field have a parent field?
   - Do all dimension names exist in the outline or the rules file?

4. Validate the file again. Return to Step 1.

Manipulating the Data Source

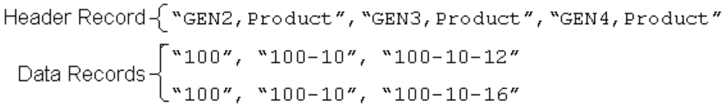
You can also build dimensions or change the properties of existing members in a dimension by adding information to the data source. You can add:

- Header information to specify the dimension and field types
- Member codes that set member properties

Using Dynamic References

You can dynamically build dimensions by adding header information to the top of the data source and specifying the location of the header information in the rules file as a dynamic reference. Figure 280 contains an example of a header record.

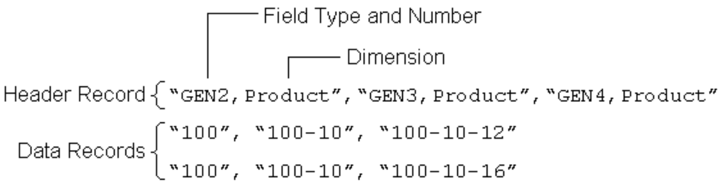
Figure 280: Header Record



```
"GEN2,Product","GEN3,Product","GEN4,Product"
"100", "100-10", "100-10-12"
"100", "100-10", "100-10-16"
```

The header record lists field definitions for each field. The field definition includes the field type and number and the dimension name into which to load the fields. The header record must be in the following format:

Figure 281: Header Record with Three Field Definitions



```
"GEN2,Product","GEN3,Product","GEN4,Product"
"100", "100-10", "100-10-12"
"100", "100-10", "100-10-16"
```

If the file delimiter is a comma, enclose each field definition in quotation marks (" ").

Make sure that the data source contains the same number of columns, and in the same order, as specified in the header information. Otherwise, the dimension build may not work as expected. If some columns are missing from the data source, replace each missing column with a comma. The comma tells Essbase that the column is there, but has no values.

After you set the header record in the data source, you must use a dynamic reference to specify the location of the header record in the rules file. If a rules file contains a dynamic reference, Essbase uses the information in the header record—rather than that in the rules file itself—to determine field types and dimensions.

Valid field types must be in capital letters:

- GEN, DUPGEN, and DUPGENALIAS
- LEVEL, DUPLEVEL, and DUPLEVELALIAS
- PARENT, CHILD
- PROPERTY
- ALIAS
- FORMULA
- CURNAME
- CURCAT
- UDA
- ATTRPARENT
- The name of an attribute dimension.

For each field type that you set, you must also enter a field number. When the field type is the name of an attribute dimension, the field number cannot be greater than 9. For more information on field numbers, see “Step 4a: Setting Rules File Field Types” on page 545.

Setting Member Properties

You can modify the properties of both new and existing members during a dimension build by setting the properties directly in the data source. Put properties in the field directly following the field they modify. For example, to specify that the Margin% member not roll up into its parent and not be shared, use the following data source:

```
Margin%  Margin%  Sales  ~ N
```

Set the field type for the properties field to Property. To set the field type to property, see “Manipulating the Data Source” on page 556.

19

The following table lists all member codes used to assign properties to members in the data source.

Code	Description
%	Express as a percentage of the current total in a consolidation
*	Multiply by the current total in a consolidation
+	Add to the current total in a consolidation
-	Subtract from the current total in a consolidation
/	Divide by the current total in a consolidation
~	Exclude from the consolidation
A	Average time balance item (applies to accounts dimensions only)
B	Exclude data values of zero or `#MISSING` in the time balance (applies to accounts dimensions only)
E	Expense item (applies to accounts dimensions only)
F	First time balance item (applies to accounts dimensions only)
L	Last time balance item (applies to accounts dimensions only)
M	Exclude data values of `#MISSING` from the time balance (applies to accounts dimensions only)
N	Never allow data sharing
O	Label only (store no data)
T	Require a two-pass calculation (applies to accounts dimensions only)
V	Create as Dynamic Calc And Store
X	Create as Dynamic Calc
Z	Exclude data values of zero from the time balance (applies to accounts dimensions only)

Performing Dimension Builds

When you have a valid dimension build rules file, you can create and update dimensions in the database in the following ways:

- Using the Data Load dialog box. For more information on loading data, see Chapter 23, “Performing and Debugging a Data Load.”
- Using Outline Editor. For more information, see “Updating Dimensions in Outline Editor” on page 559.

To create new dimensions, you must define them in the rules file.

Tip: You can perform a dimension build outside Application Manager, using the ESSCMD command BUILDDIM. For more information, see the *Technical Reference* in the `docs` directory.

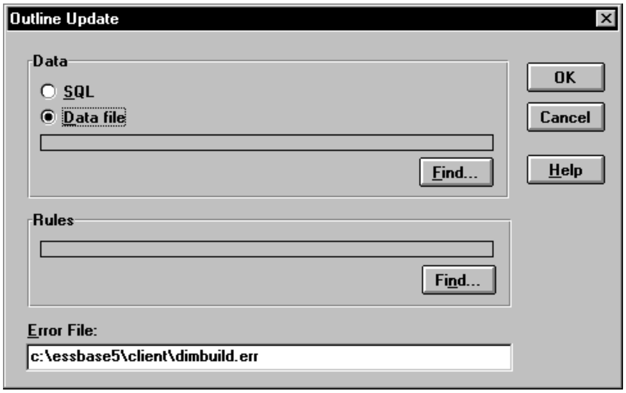
Updating Dimensions in Outline Editor

You can start the dimension build from within Outline Editor.

➤ To update dimensions using Outline Editor:

1. Open the outline in Outline Editor. See “Opening Outlines” on page 168.
2. Select File > Update Outline to open the Outline Update dialog box.

Figure 282: Outline Update Dialog Box



Outline Update

Data

○ SQL

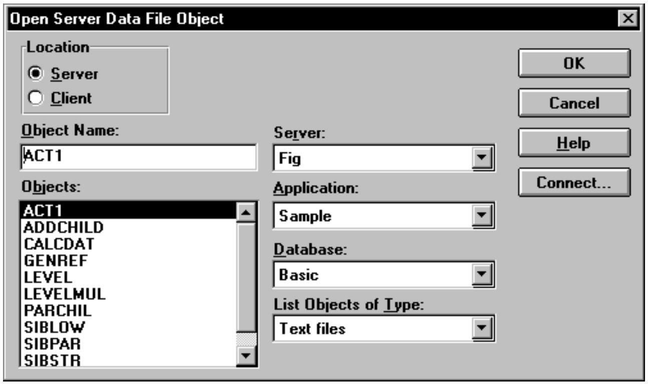
● Data file

Rules

Error File: c:\essbase5\client\dimbuild.err

3. Select the kind of data source by choosing SQL or Data File.
4. If you chose SQL, all of the information you need is stored in the rules file. Skip to Step 7.
5. If you select Data File, click Find to select the data source. The Open Server Data File Object dialog box displays:

Figure 283: Open Server Data File Object Dialog Box



Open Server Data File Object

Location
- ● Server
- ○ Client

Object Name: ACT1

Objects:
ACT1
ADDCHILD
CALCDAT
GENREF
LEVEL
LEVELMUL
PARCHIL
SIBLOW
SIBPAR
SIBSTR

Server: Fig

Application: Sample

Database: Basic

List Objects of Type: Text files

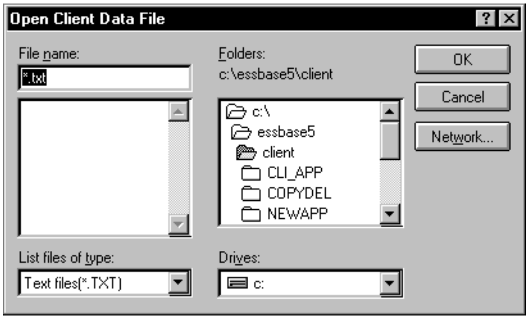
6. Make sure the appropriate server, application, and database are selected from their respective lists.

   If you select Server, the data source must reside in the database directory under `\essbase\app\`*`application_name`*`\`*`database_name`*, where *application_name* and *database_name* represent the name of your application and database. Type the name of the data source in the Object Name text box or select it from the Objects list box.

If you select Client, the data source may reside in either the application or database directory under `\ESSBASE\CLIENT` or on the drives accessible from the client file system. Click File System to select a data source from a standard Open Client Data Files dialog box. Select the data source to open.

Note: The `\essbase\app` and `\essbase\client` are the default directories specified during installation. You may have set these directories differently.

Figure 284: Open Client Data File Dialog Box



7. Select the dimension build rules file to load by clicking Find and then selecting the rules file in the Open Server Data File Object or Open Client Data File dialog box as described in Step 5.
8. Click OK. Essbase adds the dimensions in the data source to the outline.

Building Dimensions Without Connecting to the Server

You can build dimensions dynamically without connecting to the server. This might be the case if, for example, you want to do a dynamic dimension build at home and could not connect to a server from there.

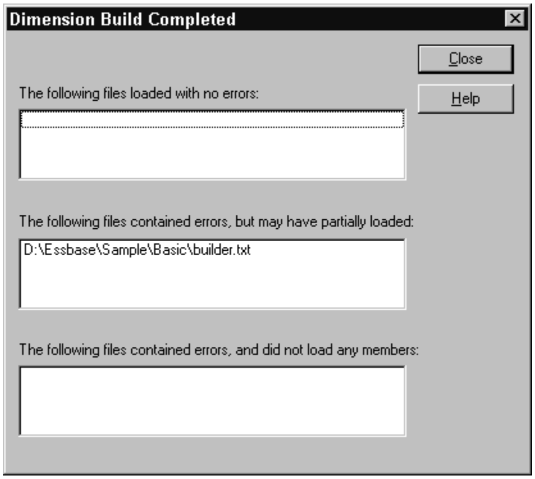
➤ To build dimensions without a connection to the server:

1. Move the outline and the data source to the client machine using standard Windows tools.
2. Perform the dimension build using Outline Editor.
3. Move the updated outline back to the server using standard Windows tools.

Debugging Dimension Builds

Essbase displays the results of dimension builds in the Dimension Build Completed dialog box as shown in Figure 285.

Figure 285: Dimension Build Completed Dialog Box



Dimension Build Completed

The following files loaded with no errors:

The following files contained errors, but may have partially loaded:

D:\Essbase\Sample\Basic\builder.txt

The following files contained errors, and did not load any members:

The Dimension Build dialog box displays the results in three different windows.

- The top window lists files that loaded completely.
- The middle window lists files that may have partially loaded.
- The bottom window list files that did not load at all.

If errors occurred during dimension building, Essbase logs them in the `\essbase\client\dimbuild.err` file. To find data errors and correct the data source, use the strategies outline in "Debugging a Data Load" on page 658. When you fix the problem, see "Loading Dimension Build or Data Load Error Logs" on page 1245 to reload the failed records.

Understanding How Essbase Builds Dimensions

Sometimes, you can track down problems with dimension builds by understanding how Essbase initializes the rules file and processes the data source. Essbase performs the following steps to initialize a rules file:

1. Validates the rules file against the associated outline.
2. Validates the dimensions. This includes ensuring that the build method and field types are compatible and that each dimension name is unique. Member names must also be unique or shared.
3. Adds new dimensions defined in the rules file to the outline.
4. Reads header records specified as dynamic references.

Then Essbase performs the following operations on each record in the data source:

1. Sets the file delimiters.
2. Applies field operations to the data, including joins, moves, splits, and creating fields using text and joins.
3. Performs all replace operations.
4. Applies select and reject criteria.
5. Adds members or member information, or both, to the outline.

Chapter 20

Introducing Data Loading

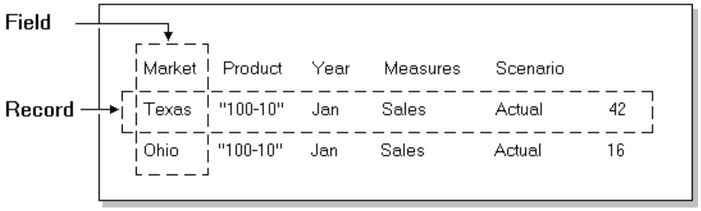
Data loading is the process of copying data from external data sources, such as spreadsheets or SQL databases, into an Essbase database. After you load the data sources into an Essbase database, you can view and analyze the data quickly. This chapter describes the various components involved in loading data, such as rules files, data sources, and free-form data source. This chapter contains the following sections:

- “Introduction to Data Sources” on page 565
- “Introduction to Rules Files” on page 572
- “Rules for Rules File Data Sources” on page 576
- “Rules for Free-Form Data Sources” on page 582
- “Security and Multi-User Considerations” on page 588

Introduction to Data Sources

As illustrated in Figure 286, a data source is composed of records and fields. A *record* is a row of fields that is read as a unit. A *field* is a vertical list of values.

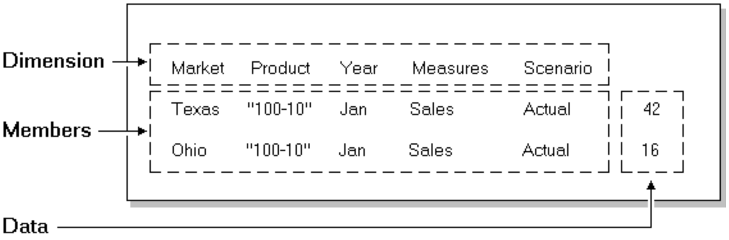
Figure 286: Records and Fields



Market	Product	Year	Measures	Scenario	
Texas	"100-10"	Jan	Sales	Actual	42
Ohio	"100-10"	Jan	Sales	Actual	16

As illustrated in Figure 287, data sources can contain dimension fields, member fields, and data fields. *Dimension fields* identify the dimensions in the database. Although you can set dimension fields in the data source, usually you define them in the rules file. *Member fields* identify members of the dimensions in the database. *Data fields* contain the data that is stored in the database.

Figure 287: Kinds of Fields



How Does Essbase Read a Data Source?

Essbase reads data sources starting at the top and proceeding from left to right. To load a data value successfully, Essbase must encounter one member from each dimension before encountering the data value. For example, in Figure 287, Essbase loads the data value `42` into the database with the members Texas, 100-10, Jan, Sales, and Actual. If Essbase encounters a data value before all members are specified, it stops loading the data source.

The data source can contain only dimension names, member names, alias names or data values; it cannot contain miscellaneous text. Not only must the data source contain enough information, the information must be in an order Essbase understands. Data sources, therefore, must be complete and correctly formatted.

Before you load data or build dimensions, you must format your data source so that it maps to the multidimensional database you are loading it into. You can format your data source in the following ways:

- By transforming the data with a rules file during loading. The original data source is not changed. Rules files perform operations on the data as the data source is loaded, such as rejecting invalid records, scaling data values or dynamically building new dimensions. You can re-use one rules file with multiple data sources. For information on rules files, see "Introduction to Rules Files" on page 572.

- By altering your data source. If the data source contains all the information necessary to map the data values in the data source to the database, you can do a free-form data load. For information on formatting requirements for free-form data sources, see the "Rules for Free-Form Data Sources" on page 582.

  Note: You must use a rules file to load SQL data and to build dimensions and members dynamically.

When Essbase loads data from external sources:

1. Essbase reads the external data source. You must format the external data source carefully.
2. If you are using a rules file, Essbase transforms the data to match the Essbase database during loading without changing the original data source. You must use a rules file if:
   - You are loading data from SQL databases.
   - Your data source contains dimensions that are not in the outline.
   - The data source is not correctly formatted.
3. Essbase stores the data in the multidimensional database.

If you are loading data into a transparent partition, follow the same steps as for loading data into a local database.

Valid Data Fields

A *data field* is a specific kind of field in a record. Data fields contain the data for their intersection in the database. In Figure 287, for example, 42 is a data field. It is the dollar sales of 100-10 (Cola) sold in Texas in January.

Essbase accepts only the following kinds of data fields:

Data Field Types	Examples
Numbers and their modifiers with no spaces or separators between them:	
• Numbers (0–9)	12
• Dollar sign ($)	$ 12 is not a valid value because of the space between the dollar sign and the 12. $12 is a valid value.
• Euro currency symbol (€)	(€)12
• Numbers in parentheses (to indicate a negative number)	(12)
• Minus sign before numbers. Minus signs after numbers are *not* valid.	-12
• Decimal point	12.3
Large numbers with or without commas	Both 1,345,218 and 1345218 are valid values.
`#MI` or `#MISSING` to represent missing or unknown values	You must insert `#MI` or `#MISSING` into a data field that has no value. If you don't, the data source may not load correctly. To replace a blank field with `#MI` or `#MISSING`, see "Replacing an Empty Field with Text" on page 627.

Valid Member Fields

A *member field* contains the name of a member or alias in a dimension. In Figure 287, for example, Texas and Ohio are members of the Market dimension. Member fields must be formatted as follows:

Data Field Types	Examples
Member fields must map to member names or aliases in the database.	A member field called Texas maps to the Texas member in the Sample Basic database. A member field called Salesperson does not map to any member in the Sample Basic database and is, therefore, invalid.
You can only load data into members that are pre-calculated, that is, you cannot load data into Dynamic Calc or Dynamic Calc And Store members.	If Year is a Dynamic Calc member, you cannot load data into it. Instead, load data into Qtr1, Qtr2, Qtr3, and Qtr4, which are not Dynamic Calc members.
A member name must be enclosed in quotes if it contains any of the following: • White space • Numeric characters (0–9) • Dashes (minus signs, hyphens) • Plus signs • & (ampersands)	For files that free form load into the Sample Basic database, the 100-10 product member name must be in quotes: "100-10" For rules on naming dimensions and members, see “Rules for Naming Dimensions and Members” on page 179.
If a member field maps to an alias, Essbase uses the current alias table.	Default is the name of the default alias table.

Invalid Member or Data Fields

When Essbase encounters an invalid member or data field, it stops the data load. Essbase loads any fields read before the invalid field into the database, resulting in a partial load of the data.

In the following file, for example, Essbase stops the data load when it encounters the 15- data value. Essbase loads the Jan and Feb Sales records, but not the Mar and Apr Sales records.

Figure 288: Invalid Data Field

```
East Cola   Actual
Sales       Jan     $10
            Feb     $21
            Mar     $15-
            Apr     $16
```

For information on continuing the load, see “Loading Dimension Build or Data Load Error Logs” on page 1245.

Setting File Delimiters

You must separate fields from each other with delimiters. Delimiters can be any combination of the following:

- Spaces
- Tabs
- New lines
- Carriage returns

Note: You cannot use commas as delimiters in free-form data sources, although you can use them in data sources you are loading using a rules file.

The delimiter you use can vary between fields. Essbase ignores excess delimiters in free-form data sources.

In Figure 289, for example, the fields are separated by spaces. Essbase ignores the extra spaces between East and Cola in the first record.

Figure 289: File Delimiters

```
East    Cola    Actual    Jan    Sales    10
East    Cola    Actual    Feb    Sales    21
East    Cola    Actual    Mar    Sales    30
```

For more information, see “Setting File Delimiters” on page 597.

Ignored Characters

Some characters are in the data source for formatting reasons only. For that reason, Essbase ignores the following characters:

== Two or more equal signs, such as for double underlining

-- Two or more minus signs, such as for single underlining

__ Two or more underscores

== Two or more IBM PC graphic double underlines (ASCII character 205)

__ Two or more IBM PC graphic single underlines (ASCII character 196)

Ignored fields do not affect the data load.

For example, Essbase ignores the equal signs in Figure 290, but loads the other fields normally.

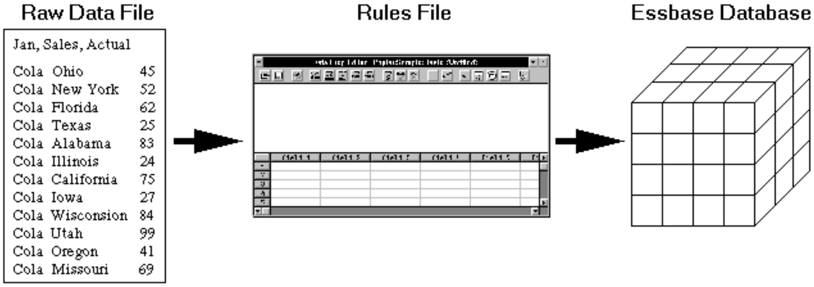
Figure 290: Ignoring Formatting Characters During Loading

```
East Actual "100-10"
        Sales    Marketing
        =====    =========
Jan     10       8
Feb     21       16
```

Introduction to Rules Files

Data load rules are a set of operations that Essbase performs on data when it loads the associated data source into the database, such as rejecting invalid records in the data source. Data sources are external sources of data such as spreadsheet files, text files, or SQL data sources. Applying data load rules to data sources makes it possible to map external data values to an Essbase database during loading.

Figure 291: Loading Data Sources through Rules Files



```
Jan, Sales, Actual

Cola  Ohio         45
Cola  New York     52
Cola  Florida      62
Cola  Texas        25
Cola  Alabama      83
Cola  Illinois     24
Cola  California   75
Cola  Iowa         27
Cola  Wisconsion   84
Cola  Utah         99
Cola  Oregon       41
Cola  Missouri     69
```

Data load rules are stored in rules files. Essbase loads the data in the data source into the database through the rules file without changing the data source. You can re-use a rules file with any data source that requires the same set of data loading rules.

You also use rules files in dimension build operations to add or change members and dimensions in outlines. For information about creating rules files and defining them for dimension build operations, see Chapter 18, “Introducing Dynamic Dimension Building” and Chapter 19, “Building Dimensions Using a Rules File.”

When to Use Data Load Rules

Use data load rules when the data load should:

- Ignore fields or strings in the data source.
- Change the order of fields by moving, joining, splitting, or creating them.
- Map the data in the data source to the database by changing strings.
- Change the data values in the data source by scaling data values or adding them to existing values in the data source.
- Set header records for missing values.

- Reject an invalid record and continue loading data.
- Add new dimensions and members to the database.
- Change existing dimensions and members in the database.

See Chapter 21, “Setting up a Rules File to Manipulate Records,” and Chapter 22, “Manipulating Fields Using a Rules File,” for information about manipulating fields and records. See Chapter 18, “Introducing Dynamic Dimension Building,” for information about changing or adding members and dimensions.

How to Create Data Load Rules

You create data load rules using the following process:

1. Select the data source in the Data Prep Editor. For example, you can select an SQL data source, a spreadsheet, or a text file. See “Selecting the Data Source” on page 589.
2. Set the file delimiter for your data source. For example, you can select tabs, commas, or spaces. See “Setting File Delimiters” on page 597.
3. Perform operations on records. For example, you can set up header records, and define select and reject operations. See “Setting up a Rules File to Manipulate Records” on page 589.
4. Perform operations on fields. For example, you can split them, join them, and create new ones. See “Manipulating Fields Using a Rules File” on page 613.
5. Map fields in the data source to dimensions and members in the database. See “Mapping Fields to Member Names” on page 624.
6. Save and validate the data load rules. For more information, see “Validating and Saving Data Load Rules” on page 608.

How Does Essbase Execute Operations in a Rules File?

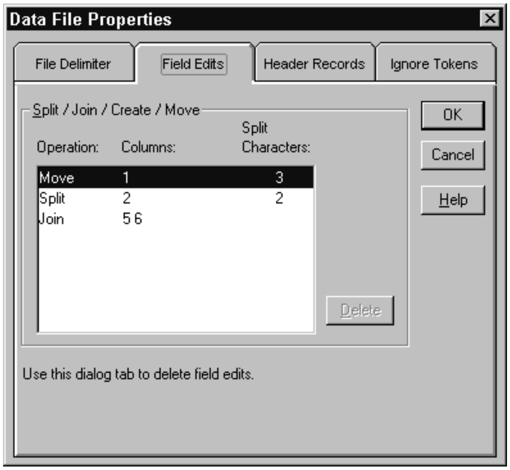
When Essbase loads data using a rules file, it executes the operations in the rules file in the following order:

1. Essbase sets all file delimiters, including fixed-width columns. For more information, see “Setting File Delimiters” on page 597.
2. Essbase performs all field operations in the order they are defined in the rules file. Field operations alter the position or number of fields and include moves, splits, joins, create using text, and create using join operations. For more information, see “Ordering Fields” on page 617.

   If you’re not sure in what order the field operations were defined, select Options > Data File Properties and click the Field Edits tab. The Data File Properties dialog box appears, listing all the field operations.

   The rules file in Figure 292, for example, contains move, split, and join operations.

   Figure 292: Field Operations



Data File Properties

Split / Join / Create / Move

Operation:	Columns:	Split Characters:
Move	1	3
Split	2	2
Join	5 6	

Use this dialog tab to delete field edits.

3. Essbase applies all properties for each field, applying all of the properties to field1 before proceeding to field2. Essbase applies field properties in the following order:

   a. Ignores fields set to ignore during data load.

   b. Ignores fields set to ignore during outline update.

c. Flags the data field.

d. Applies field names.

e. Applies field generations.

f. Performs all replaces in the order they are defined in the rules file. If you're not sure what order the replace operations are in, select Field > Properties and click the Global Properties tab. The Field Properties dialog box appears, listing all replace operations.

g. Drops leading and trailing white spaces.

h. Coverts spaces to underscores.

i. Applies suffix and prefix operations.

j. Scales data values.

k. Converts text to lowercase.

l. Converts text to uppercase.

For more information, see Chapter 22, "Manipulating Fields Using a Rules File."

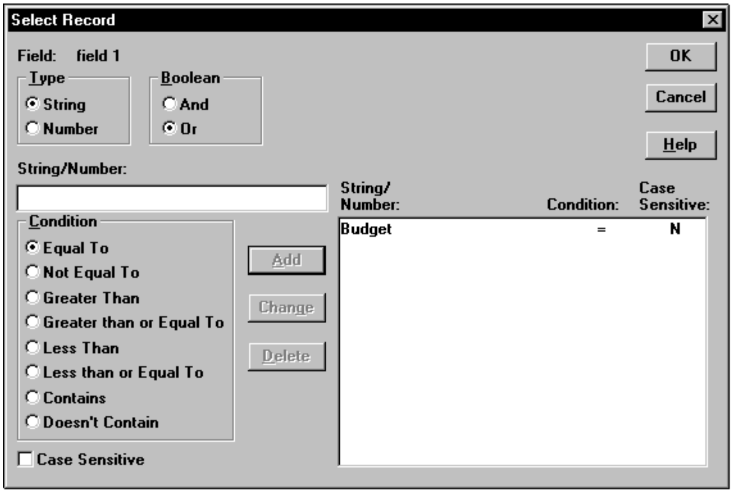
4. If you choose to skip lines, Essbase skips the number of lines that you specified, otherwise Essbase proceeds to the first record. For more information, see "Defining Header Information in the Rules File" on page 598.

5. Essbase performs selection or rejection criteria in the order that they are defined in the rules file. Essbase loads or doesn't load individual records in the data source based on these criteria specified in the rules file.

   If you're not sure in what order the selection or rejection criteria are defined, select Record > Select or Record > Reject. The Select Record or Reject Record dialog box displays, listing all the selection or rejection operations.

The rules file in Figure 293, for example, contains a selection criterion.

Figure 293: Selecting Records



Select Record

Field: field 1

Type
● String
○ Number

Boolean
○ And
● Or

String/Number:

Condition
● Equal To
○ Not Equal To
○ Greater Than
○ Greater than or Equal To
○ Less Than
○ Less than or Equal To
○ Contains
○ Doesn't Contain

□ Case Sensitive

String/ Number:	Condition:	Case Sensitive:
Budget	=	N

Rules for Rules File Data Sources

This section describes rules you must follow when formatting data sources that are loaded using rules files.

Rules for Dimension Fields

Essbase must be able to identify each dimension in the database using information in the data source or the rules file. The field values in a dimension field must contain members for that dimension. For example, a field defined as Year has members such as Jan, Feb, and Mar. A field defined as Product has members such as Cola and Root Beer.

If the data source does not identify each dimension in the database, you must identify the missing dimensions in a header record. For example, the Sample Basic database has a dimension for Year. If several data sources arrive with monthly numbers from different regions, the month itself might not be specified in the data sources. You must set header information to specify the month. For information on setting header records, see "Using Header Information" on page 598.

If a member value is missing for a dimension field, the value from the last valid record is used. For example, when you load Figure 294 to the Sample Basic database, Essbase maps the Ohio member field into the Market dimension for all records, including the records that have Root Beer and Diet Cola in the Product dimension.

Figure 294: Valid Missing Members

```
Jan, Sales, Actual
Ohio       Cola          25
           "Root Beer"   50
           "Diet Cola"   19
```

Essbase stops the data load if no prior records contain a value for the missing member field. If you tried to load Figure 295 into the Sample Basic database, for example, the data load would stop while trying to process the first record, because the Market dimension (Ohio, in Figure 294) is not specified.

Figure 295: Invalid Missing Members

```
Jan, Sales, Actual
           Cola          25
           "Root Beer"   50
           "Diet Cola"   19
```

For information on restarting the load, see "Loading Dimension Build or Data Load Error Logs" on page 1245.

Rules for Member Fields

After Essbase identifies all dimensions, it maps the member fields into the appropriate members in the outline. A member field can map to a single member name, such as Jan (which is a member of the Year dimension), or a member combination, such as Jan, Actual (which are members of the Year and Scenario dimensions).

If the data source contains member fields, the data source or rules file must identify the dimensions they map to in the database. For example, the following file contains member fields for Jan, Cola, East, Sales, and Actual. The rules file must identify the dimensions that those members map to, in this case Year, Product, Market, Measures, and Scenario.

Figure 296: All Dimensions Specified

```
Jan    Cola    East    Sales    Actual    100
Feb    Cola    East    Sales    Actual    200
```

Quoting Member Names

You must use double quotes around member names that contain the same character as the file delimiter. File delimiters are the character(s) that separate fields in the data file.

Note: You do not have to double quote any member names that come from SQL data sources, because the fields in SQL data sources are not delimited by characters.

For example, if your data source is delimited by spaces, use quotes around member names with embedded spaces. Figure 297, for example, quotes New York, because it has a space in it:

Figure 297: Quoted Member Names

```
Cola    Jan    "New York"   Actual    Sales   50
Cola    Jan    Ohio         Actual    Sales   78
```

Unknown Member Names

If a member field contains an unknown member name, Essbase rejects the entire record during the data load. If there was a prior record with a member name for the missing data load field value, Essbase continues to the next record. If there are no prior records, the data load stops.

For example, when you load Figure 298 into the Sample Basic database, Essbase rejects the record containing Ginger Ale because it is not a valid member name. Essbase loads the records containing Cola, Root Beer, and Cream Soda. If Ginger Ale were in the first record, however, the data load would stop.

Figure 298: Unknown Members

```
Jan, Sales, Actual
Ohio    Cola          2
        "Root Beer"   12
        "Ginger Ale"  15
        "Cream Soda"  11
```

Note: Instead of rejecting the record, you can add the new members encountered to the database using the dimension build feature. See Chapter 18, "Introducing Dynamic Dimension Building."

For information on restarting the load, see "Loading Dimension Build or Data Load Error Logs" on page 1245.

Rules for Data Fields

After Essbase identifies all dimensions and maps the member fields into the appropriate members in the outline, it loads the data fields to the Essbase database. The data source or rules file must contain enough information for Essbase to determine where to put the data. To Essbase, data are the numbers stored for each intersection in the database. In Figure 287, for example, 42 is the data stored in the database as the actual quantity of 100-10 (Cola) sold in Texas in Jan (January).

If the data source contains a member field for every dimension and only one data column, you must set the data column as a data field. To read Figure 299 into the Sample Basic database, for example, identify the last column as a data field.

Figure 299: Setting Data Fields

```
Jan    Cola    East    Sales    Actual    100
Feb    Cola    East    Sales    Actual    200
```

To identify a column as a data field, see "Defining a Column as a Data Field" on page 633.

Assigning All Members

The field name you assign to a data field must be a dimension, a member, or a member combination from the database. For example, the data field in the following file specifies each member so Essbase knows where to put the data.

Figure 300: Assigning Data Fields

```
             Jan, Actual
Cola         East    Sales   100
"Root Beer"  East    Sales   200
```

The only exception to this rule is a data source where each record contains a data load field for every dimension and one data column, such as Figure 299, where each record specifies each dimension (for example, Jan, Cola, East, Sales, and Actual) and the final column is a data field (for example, 100).

Empty Data Fields

If there is no value in the data field (or the value is #MISSING), Essbase does not change the existing data value in the database. Essbase won't replace current values with empty values.

Note: If the data source contains blank fields for data values, replace them with #MI or #MISSING. Otherwise, the data may not load correctly. To replace a blank field with #MI or #MISSING, see "Replacing an Empty Field with Text" on page 627.

Rules for Extra Fields

If the data source contains fields that you don't want to load into the database, you can tell Essbase to ignore those fields. For example, the Sample Basic database has five standard dimensions into which you would load data: Year, Product, Market, Measures, and Scenario. If the data source had an extra field, such as Salesperson, that isn't a member of any dimension, tell Essbase to ignore the Salesperson field during the data load.

No Blank Fields

If a rules file has blank fields, the data source won't load. So, for example, if your rules file has extra fields at the end, it won't work. Join the empty fields with the field next to them.

For more information, see "Joining Fields" on page 618.

Each Record Must Have the Same Number of Fields

Each record must have the same number of fields. If fields are missing, the data loads incorrectly. For example, the file in Figure 301, is invalid, because there is no value under Apr. To fix the file, insert `#MISSING` or `#MI` into the missing field.

Figure 301: Missing Fields

```
Actual Ohio Sales Cola
Jan     Feb    Mar    Apr
10      15     20
```

Figure 302 is valid because #MI was inserted to replace the missing field.

Figure 302: Valid Missing Fields

```
Actual Ohio Sales Cola
Jan     Feb    Mar    Apr
10      15     20     #MI
```

Rules for File Delimiters

A data source cannot have extra file delimiters if you are using a rules file. The rules file reads the extra delimiters as empty fields. For example, if you tried to load the file in Figure 303 into the Sample Basic database using a rules file, it would fail. Essbase reads the extra comma between East and Cola in the first record as an extra field. Essbase then puts Cola into Field 3. In the next record, however, Cola is in Field 2. Essbase expects Cola to be in Field 3 and stops the data load.

Note: You cannot use commas as delimiters in free-form data sources, although you can use them in data sources you are loading using a rules file.

Figure 303: File Delimiters

```
East,,Cola,Actual,Jan,Sales,10
East,Cola,Actual,Feb,Sales,21
East,Cola,Actual,Mar,Sales,30
```

To solve the problem, delete the extra delimiter from the data source.

Rules for Free-Form Data Sources

If a data source contains enough information to load into the database, you can load the data source directly. This kind of load is called a free-form data load.

This section describes how free-form data sources must be formatted. If your data source is not correctly formatted, it will not load. You can edit your data source directly to fix the problem. If you find that you must perform many edits (such as moving several fields and records), it might be easier to load the data source using a rules file. See "Introduction to Rules Files" on page 572.

Note: If the data source contains blank fields for data values, replace them with `#MI` or `#MISSING`. Otherwise, the data may not load correctly. To replace a blank field with `#MI` or `#MISSING` using a rules file, see "Replacing an Empty Field with Text" on page 627.

As a free-form data source, you can use a file previously created through the Application Manager's export feature. Such a file is already formatted properly.

Use the LOADDATA command in ESSCMD to load data free form. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Formatting Ranges of Member Fields

You can express member names as ranges within a dimension. For example, Sales and Profit form a range in the Measures dimension. Ranges of member names can handle a series of consecutive values.

A data source can contain ranges from more than one dimension at a time.

In Figure 304, for example, Jan and Feb form a range in the Year dimension and Sales and Profit form a range in the Measures dimension.

Figure 304: Multiple Ranges of Member Names

```
Texas             Sales        Profit
                  Jan   Feb    Jan   Feb
Actual  "100-10"  98    89     26    19
        "100-20"  87    78     23    32
```

In Figure 304, Sales is defined for the first two columns and Profit for the last two.

Ranges Set Automatically

When Essbase encounters two or more members from the same dimension with no intervening data fields, it sets up a range for that dimension. The range stays in effect until Essbase encounters another member name from the same dimension, at which point Essbase replaces the range with the new member or new member range.

Figure 305, for example, contains a range of Jan to Feb in the Year dimension. It remains in effect until Essbase encounters another member name, such as Mar. If Essbase encounters Mar, the range changes to Jan, Feb, Mar.

Figure 305: Ranges of Member Names

```
Texas Sales
                  Jan   Feb    Mar
Actual  "100-10"  98    89     58
        "100-20"  87    78     115
```

Data Values Out of Range

When Essbase encounters a member range, it assumes that there is a corresponding range of data values. If the data values are not in the member range, the data load stops. Essbase loads any data fields read before the invalid field into the database, resulting in a partial load of the data.

Figure 306, for example, contains more data fields than the defined range of members. The data load stops when it reaches the `10` data field. Essbase loads the `100` and `120` data fields into the database.

Figure 306: Extra Data Values

```
Cola Actual East
         Jan    Feb
Sales    100    120    10
COGS     30     34     32
```

For information on restarting the load, see "Loading Dimension Build or Data Load Error Logs" on page 1245.

Duplicate Members in a Range

Be sure to structure ranges in the source data so that Essbase interprets them correctly. If the same member appears more than once in a range, Essbase ignores the duplicate members.

The file in Figure 307 contains two ranges: Actual to Budget and Sales to COGS. It also contains duplicate members.

Figure 307: Duplicate Members in a Range

```
Cola East
        Actual    Budget    Actual    Budget
        Sales     Sales     COGS      COGS
Jan     108       110       49        50
Feb     102       120       57        60
```

Essbase ignores the duplicate members. The members that Essbase ignores have a line through them in the following example:

Figure 308: Ignored Duplicate Members

```
Cola East
        Actual    Budget    ~~Actual~~    ~~Budget~~
        Sales     ~~Sales~~     COGS      ~~COGS~~
Jan     108       110       49        50
Feb     102       120       57        60
```

For Actual, the first member in the first range, Essbase maps data values to each member in the second range. Essbase then proceeds to the next value in the first range, Budget, similarly mapping values to each member in the second range. As a result, Essbase Essbase interprets the file as shown in Figure 309.

Figure 309: How Essbase Interprets the File in Figure 307

```
Cola East
           Actual              Budget
           Sales     COGS    Sales     COGS
Jan        108       110     49        50
Feb        102       120     57        60
```

How Essbase Reads Multiple Ranges

As Essbase scans a file, it processes the most recently encountered range first when identifying a range of data values. In Figure 309, for example, there are two ranges: Actual and Budget and Sales and COGS. While reading the file from left to right and top to bottom, Essbase encounters the Actual and Budget range first and the Sales and COGS range last. Because the Sales and COGS range is encountered last, Essbase puts data fields in that part of the database first.

Formatting Columns

Files can contain columns of fields. Columns can be symmetric or asymmetric. Symmetric columns have the same number of members under them. Asymmetric columns have different numbers of members under them. Essbase supports loading data from both types of columns.

Symmetric Columns

Symmetric columns have the same number of members under them. In Figure 310, for example, each dimension column has one column of members under it. For example, Product has 100-10 under it.

Figure 310: Symmetric Columns

```
Product     Measures   Market    Year    Scenario    *data*
"100-10"    Sales      Texas     Jan     Actual      112
"100-10"    Sales      Ohio      Jan     Actual      145
```

The columns in the following file are also symmetric, because Jan and Feb have the same number of members under them:

Figure 311: Groups of Symmetric Columns

```
                              Jan            Feb
                        Actual  Budget  Actual  Budget
"100-10"  Sales  Texas  112     110     243     215
"100-10"  Sales  Ohio   145     120     81      102
```

Asymmetric Columns

Columns can also be asymmetric. In Figure 312, the Jan and Feb columns are asymmetric because Jan has two columns under it (Actual and Budget) and Feb has only one column under it (Budget):

Figure 312: Valid Groups of Asymmetric Columns

```
                 Jan     Jan     Feb
                 Actual  Budget  Budget
"100-10"  Sales  Texas   112     110      243
"100-10"  Sales  Ohio    145     120      81
```

If a file contains more than one asymmetric group of member columns, you must label each column with the appropriate member name.

The file in Figure 313, for example, is not valid because the column labels are incomplete. The Jan label must appear over both the Actual and Budget columns.

Figure 313: Invalid Asymmetric Columns

```
                         Jan             Feb
                         Actual  Budget  Budget
"100-10"  Sales  Texas   112     110     243
"100-10"  Sales  Ohio    145     120     81
```

This file in Figure 314 is valid because the Jan label is now over both Actual and Budget. It is clear to Essbase that both of those columns map to Jan.

Figure 314: Valid Asymmetric Columns

```
                         Jan     Jan     Feb
                         Actual  Budget  Budget
"100-10"  Sales  Texas   112     110      243
"100-10"  Sales  Ohio    145     120      81
```

Security and Multi-User Considerations

Essbase supports concurrent multiple users reading and updating the database. This means that users can use the database while you are dynamically building dimensions, loading data, or calculating the database. In a multiple-user environment, Essbase protects your data using the security system described in Chapter 15, “Managing Security for Users and Applications.”

- Security Issues: The data load process works with the security system to prevent unauthorized users from changing the database. Only users with Write access to a database can load data into the database. Write access can be provided globally or by using filters.
- Multiple User Issues: You can load data while multiple users are connected to a database. Essbase uses a block locking scheme for handling multi-user issues. When you load data, Essbase does the following:
  - Locks the block it is loading into so that no one else can write to it. The settings on the Transaction page of the Database Settings dialog box determine whether other users get Read-Only access to the locked block, how long Essbase waits for a locked block to be released, and other transaction handling parameters. See Chapter 40, “Ensuring Data Integrity,” for more information.

    For information on how to see which user has a lock on a particular block, see Chapter 15, “Managing Security for Users and Applications.”
  - Updates the block. Essbase either unlocks the block at that time, or waits for the entire data load to complete before unlocking the block, depending on your transaction settings. See Chapter 40, “Ensuring Data Integrity,” for more information.

Chapter 21

Setting up a Rules File to Manipulate Records

This chapter describe how to create a data load or dimension build rules file that performs operations on records using Essbase Application Manager. For information about performing operations on fields within a record, see Chapter 22, “Manipulating Fields Using a Rules File.” For information about loading data using rules files, including prerequisites, see Chapter 23, “Performing and Debugging a Data Load.”

This chapter contains the following sections:

Selecting the Data Source

This section describes how to select your data source:

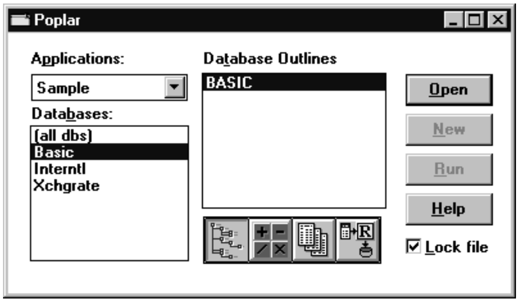
Connecting to the Server

Before you can create data load rules for a data source, you may want to connect to the server. You are not required to connect to the server before defining data load rules, because you can create the rules file on your client machine.

Viewing the Application and Database

Before you create data load rules for a data source, you may want to view the application and database. You are not required to view the application or database before defining data load rules. To view rules files, you must open Data Prep Editor which is available through the Application Desktop window. The Application Desktop window appears after you connect to the server.

Figure 315: Application Desktop Window



1. Select the application to view from the Applications list box; for example, the Sample application.
2. Select the database from the Databases list box; for example, the Basic database.

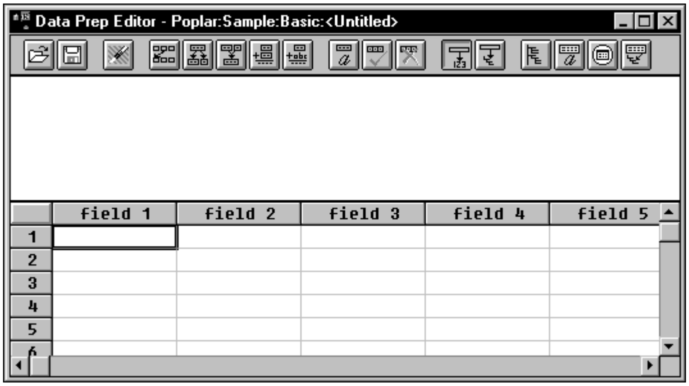
Opening the Data Prep Editor

➤ To define a rules file, you must use the Data Prep Editor. To open the Data Prep Editor:

1. In the Application Desktop window, click the Data Load Rules button, . Then click New to open the Data Prep Editor with a new rules file or Open to open an existing rules file.

2. Select View > Data Load Fields or click the Data Load button, , to make sure that the Data Prep Editor is in data load mode.

Figure 316: Data Prep Editor



Data Prep Editor - Poplar:Sample:Basic:<Untitled>

	field 1	field 2	field 3	field 4	field 5
1					
2					
3					
4					
5					
6					

The Data Prep Editor contains two windows. The top window provides a view of the data source, called the raw data source. The bottom window contains a grid showing the appearance of records after rules are applied, that is, as they will be loaded into the database. Any rules you define do not modify the content of the raw data source.

Viewing Data Load Fields or Dimension Build Fields

The Data Prep Editor can display two different kinds of fields: data load fields and dimension build fields. To determine which fields are currently displayed, pull down the View menu.

- Select View > Data Load or click the Data Load button, , to view data load fields. Fields set to be ignored during the data load turn gray.
- Select View > Dimension Building Fields or click the Dimension Build button, , to view dimension build fields. Fields set to be ignored during the dimension build turn gray. See Chapter 18, "Introducing Dynamic Dimension Building," for more information on dimension building.

Customizing the Data Prep Editor

You can customize your view of the Data Prep Editor:

- To hide the gridlines, select View > Gridlines. The check mark disappears next to the Gridlines command in the View menu and the gridlines disappear from the Data Prep Editor. To show the gridlines, select View > Gridlines.
- To hide the raw data, select View > Raw Data. The check mark disappears next to the Raw Data command in the View menu and the raw data window disappears from the Data Prep Editor. To show the raw data, select View > Raw Data.
- To hide the toolbar, select View > Toolbar. The check mark disappears next to the Toolbar command in the View menu and the toolbar disappears from the Data Prep Editor. To show the toolbar, select View > Toolbar.

Opening a Data Source

After you open the Data Prep Editor, you can open data sources, such as text files, spreadsheet files, and SQL data sources. This section describes how to open the following kinds of data sources:

- "Opening a Text File" on page 593
- "Opening a Spreadsheet File" on page 594
- "Opening an SQL Data Source" on page 595

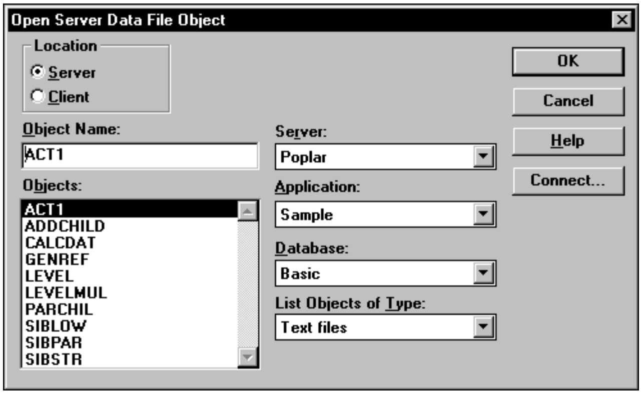
Opening a Text File

After you open the Data Prep Editor, you can open text files in it.

➤ To open a text file:

1. Select File > Open Data File to view a list of text files. The Open Server Data File Object dialog box contains values for the last data source you opened for this rules file. In this example, the last file opened was the Act1 file in the Sample Basic database.

Figure 317: Open Server Data File Object Dialog Box



Open Server Data File Object

Location: ● Server ○ Client

Object Name: ACT1

Objects:
ACT1
ADDCHILD
CALCDAT
GENREF
LEVEL
LEVELMUL
PARCHIL
SIBLOW
SIBPAR
SIBSTR

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Text files

2. If “Text files” is not selected in the List Objects of Type list box, select it.

Note: Text files must end in `.txt` on the file system. If they do not, rename them.

If you select Server, the text file to load must reside in the database directory under `\essbase\app\`*`application_name`*`\`*`database_name`*, where *application_name* and *database_name* represent the name of your application and database. Type the name of the text file in the Object Name text box or select it from the Objects list box. For example, ACT1.

If you select Client, the text file may reside in either the application or database directory under `\essbase\client` or on the drives accessible from the client file system. Click File System to select a text file from a standard Open Client Data Files dialog box.

Note: The `\essbase\app` and `\essbase\client` are the default directories specified during installation. You may have set these directories differently.

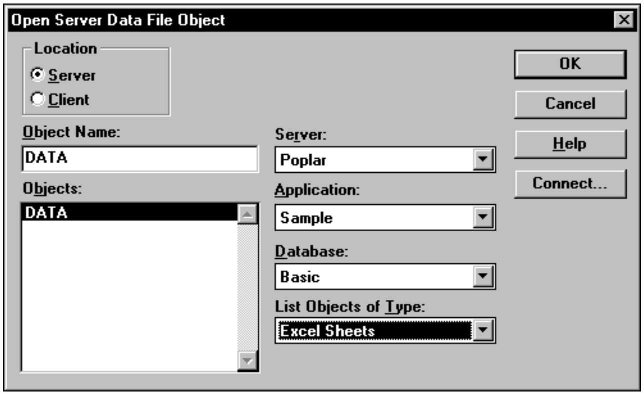
3. When you find the file to open, click OK. The contents of the file appear in the top window of the Data Prep Editor. The `act1.txt` file is tab delimited, which is the default setting in the Data Prep Editor.

Opening a Spreadsheet File

➤ To open a spreadsheet file:

1. Select File > Open Data File to open the Open Server Data File Object dialog box.

Figure 318: Open Server Data File Object Dialog Box: Spreadsheet Files



Open Server Data File Object

Location: ● Server ○ Client

Object Name: DATA

Objects: DATA

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Excel Sheets

2. Select the kind of spreadsheet to open from the List Objects of Type list box. For example, you would select Excel Sheets to open an Excel spreadsheet file.
3. Select the spreadsheet from the Objects list box.

   If you select Server, the spreadsheet to load must reside in the database directory under `\essbase\app\`*`application_name`\`database_name`* where *application_name* and *database_name* represent the name of your application and database. Type the name of the spreadsheet in the Object Name text box or select it from the Objects list box. For example, DATA.

If you select Client, the spreadsheet may reside in either the application or database directory under `\essbase\client` or on the drives accessible from the client file system. Click File System to select a spreadsheet from a standard Open Client Data Files dialog box. The `asymm.xls` spreadsheet, for example, is in the `\essbase\client\sample\` directory.

Note: `essbase is` the default directory specified during installation. You may have specified a different default directory.

4. When you find the file to open, click OK. The contents of the file appear in the top window of the Data Prep Editor.

Opening an SQL Data Source

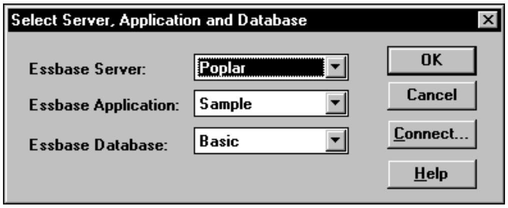
When you open the Data Prep Editor, you can open an SQL data source if you've licensed Essbase SQL Interface. The *Essbase SQL Interface Guide* provides information on supported environments, installation, and connection to supported data sources. Contact your Essbase administrator for more information.

Note: When you open an SQL data source, the rules fields default to the SQL data source column names. If these names are the same as your Essbase dimensions, you don't have to perform any field mapping.

➤ To open an SQL data source:

1. Select File > Open SQL to open the Select Server, Application and Database dialog box. If you are connected to a server, the dialog box contains the values for that server.

Figure 319: Select Server, Application and Database Dialog Box



Select Server, Application and Database

Essbase Server: Poplar

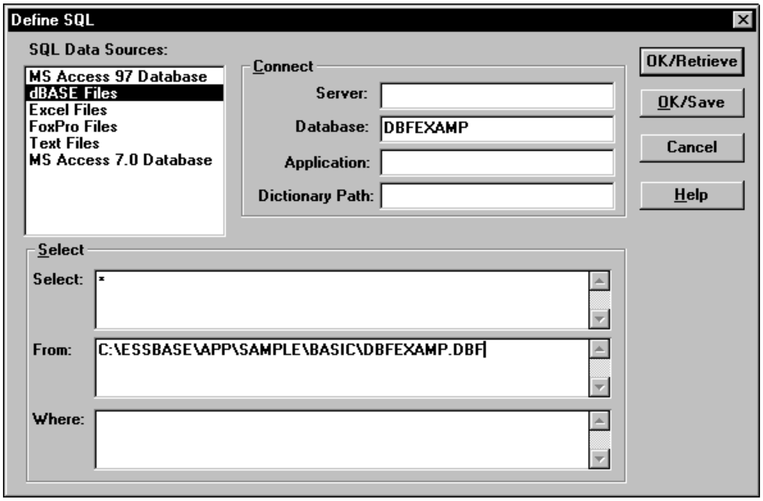
Essbase Application: Sample

Essbase Database: Basic

2. Select the server, application, and database to open. If you are already connected to the correct server, application, and database, click OK. This server, application, and database act as the client for SQL access.

3. Click OK. The Define SQL dialog box appears:

Figure 320: Define SQL Dialog Box



Server:

Database: DBFEXAMP

Application:

Dictionary Path:

Select: *

From: C:\ESSBASE\APP\SAMPLE\BASIC\DBFEXAMP.DBF

Where:

4. Select the SQL data source to use from the SQL Data Source list box; for example, dBASE files.
5. Enter the name of the database in the Database text box; for example, `dbfexamp`.
6. Enter the location of the SQL data source in the From list box; for example:

```
c:\essbase\app\sample\basic\dbfexamp.dbf
```

7. Enter any additional information that is required to connect to your SQL data source, such as the server, application, user ID, or password. To connect to a Sybase SQL Server, for example, you would enter the user ID, password, database, server, and application.
8. Define any select statements in the Select and Where list boxes. By default, the select statement is * (which selects all rows in the table).
9. Click OK/Retrieve to retrieve the SQL data source file or OK/Save to save your settings. The contents of the data source appear in the top window of the Data Prep Editor.

Setting File Delimiters

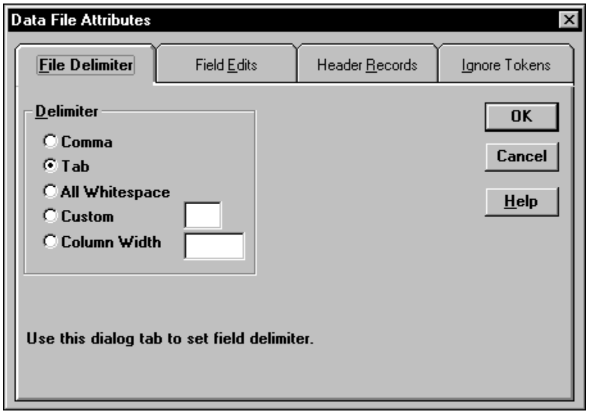
File delimiters are the characters that separate fields in the data source. By default, the rules file separates fields with tabs. You can set the file delimiter to be a comma, tab, whitespace, a fixed-width column, or a custom value. Usually, setting the file delimiters is the first thing you do after opening a data source.

Note: You do not need to set file delimiters for SQL data.

➤ To set file delimiters:

1. Select Options > Data File Properties or click the Data File Properties button, , to open the Data File Properties dialog box. Click the File Delimiter tab.

Figure 321: File Delimiter Page



Delimiter

○ Comma
● Tab
○ All Whitespace
○ Custom
○ Column Width

Use this dialog tab to set field delimiter.

2. Select the type of delimiter to use in your file:
   - Comma
   - Tab
   - All Whitespace
   - Custom—In the text box, enter a single character as the custom delimiter.
   - Column Width—In the text box, enter the width of the column. The column width must be a five digit or smaller number. Use column width for data sources with fixed-width columns.
3. Click OK.

Using Header Information

Data sources can contain data records and header records. *Data records* contain data: member fields and data fields. *Header records* describe the contents of the data source and how to load data values in the data source to the database.

Rules files contain records that translate the data in the data source to map it to the database. As part of that information, rules files can also contain header records.

You can create header records using the following methods:

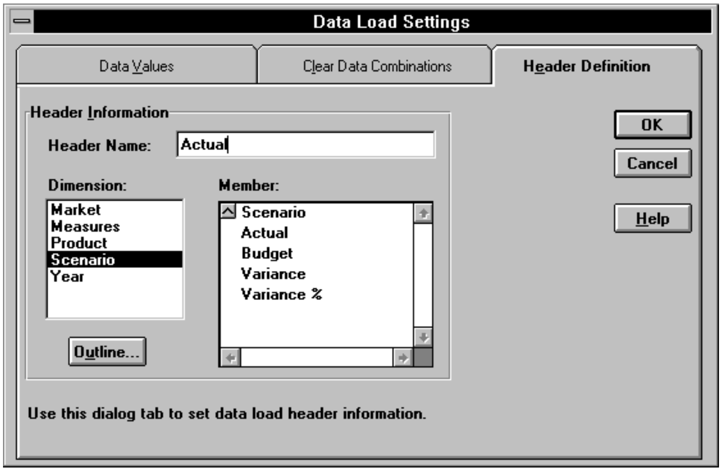
- In the rules file using the Data Prep Editor.
- In the data source using a text editor or spreadsheet; then set them as header records in the rules file.

Defining Header Information in the Rules File

You can define header information in the rules file. These headers are only used during data loading or dimension building and do not change the data source. Header information in a rules file is not used if there is also a dynamic reference in the rules file pointing to a header record in the data source.

1. Select Options > Data Load Settings or click the Global Data Load Properties button, , to open the Data Load Settings dialog box. Click the Header Definition tab.

Figure 322: Header Definition Page



Header Name: Actual

Dimension: Market, Measures, Product, Scenario (selected), Year

Member: Scenario, Actual, Budget, Variance, Variance %

Use this dialog tab to set data load header information.

2. Enter one or more member or member combinations in the Header Name text box or select the dimensions and members from the Dimension and Member lists. For example, to specify the Year dimension as March, enter `Mar`. Enter the `Mar, Budget` member combination to specify both the Year and Scenario dimensions. You must separate dimensions and members with a comma.

Note: Only one member per dimension is allowed in the header. For example, `Feb,Mar` is an invalid header because it contains two members of the Year dimension.

3. If the rules file is not associated with an outline, the Dimension and Member lists are empty. Click Outline to associate the rules file with an outline.

4. Click OK.

Defining Header Information in the Data Source for a Data Load

You can define header information directly in the data source. Placing header information in the data source makes it possible to use the same rules file for multiple data sources with different formats, because the data source format is specified in the data source header and not the rules file.

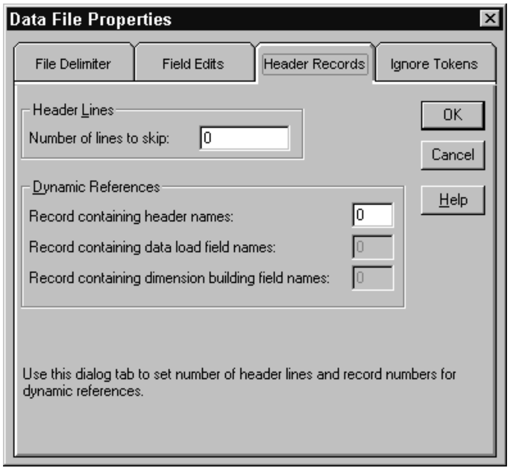
When you add one or more headers to the data source, you must also specify the location of the headers in the data source using dynamic references. *Dynamic references* are set in the rules file and tell Essbase to read the header information as a header record and not a data record. When setting dynamic references, you can also specify which type of header information is in which header record.

Header information defined in the data source takes precedence over header information defined in the rules file.

➤ To define header information in the data source:

1. Add the header information to the data source using a text editor or spreadsheet. Member combinations must be separated by a comma.
2. Select Options > Data File Properties or click the Data File Properties button, , to open the Data File Properties dialog box. Click the Header Records tab.

Figure 323: Header Records Page



Header Lines

Number of lines to skip: 0

Dynamic References

Record containing header names: 0

Record containing data load field names: 0

Record containing dimension building field names: 0

Use this dialog tab to set number of header lines and record numbers for dynamic references.

3. Enter the number of lines to skip in the data source in the “Number of lines to skip” text box. Skipping lines means that Essbase does not process the records in those lines as data records. Essbase still processes those records as header records. You should tell Essbase to skip all header records.
4. Enter the number of the record in the data source that contains the header names in the “Record containing header names” text box. Header names, for example, could be Jan, Actual.

   Note: Each dynamic reference record number must be unique and cannot be larger than 4000.

5. Enter the number of the record that contains the data load field names in the “Record containing data load field names” text box. In this case, the data load field names record contains information to map the members in the data source to dimensions in the database. A header record in a data source to load into the Sample Basic database could contain dimension names such as Product, Market, Scenario, Measures, and Year or any valid field name.
6. Enter the number of the record that contains the dimension building field names in the “Record containing dimension building field names” text box. See Chapter 18, “Introducing Dynamic Dimension Building,” for more information.
7. Click OK.

Selecting Records

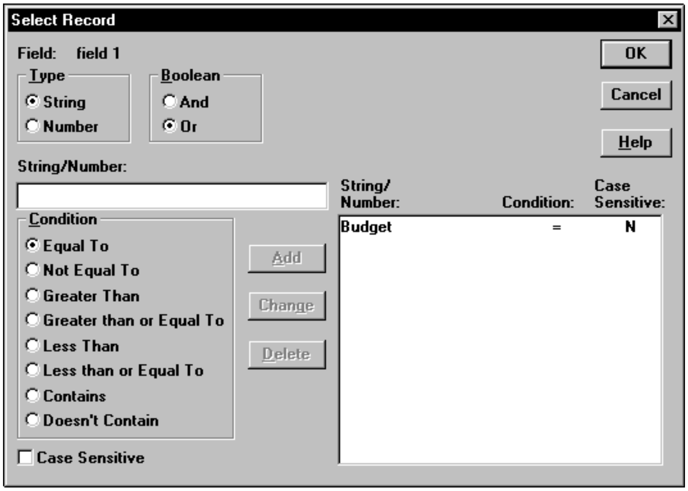
You can specify which records Essbase loads into the database or uses to build dimensions by setting selection criteria. *Selection criteria* are string and number conditions that must be met by one or more fields before Essbase loads the record. If a field or fields in the record does not meet the selection criteria, Essbase doesn't load the record. For example, to load only the Budget data from a data source, you could create a selection criterion to load only records where the first field was Budget.

➤ To define the selection criteria:

1. Select the field to which to apply the criteria. For example, field 1.

2. Select Record > Select or click the Record Selection button, ☑, in the Data Prep Editor toolbar to open the Select Record dialog box.

Figure 324: Select Record Dialog Box



Field: field 1

Type: ● String ○ Number

Boolean: ○ And ● Or

String/Number:

Condition: ● Equal To ○ Not Equal To ○ Greater Than ○ Greater than or Equal To ○ Less Than ○ Less than or Equal To ○ Contains ○ Doesn't Contain

□ Case Sensitive

String/ Number:	Condition:	Case Sensitive:
Budget	=	N

3. Set the field type as string or number by choosing the String or Number option. If the field is a string, you can define criteria that are case-sensitive by selecting Case Sensitive.

4. Enter the string or number upon which the criterion is based in the String/Number text box; for example, Budget.
5. Select how to evaluate the field by clicking one of the condition options in the Condition box; for example, Equal To.
6. Add the selection criterion to the list by clicking Add. To create multiple selection criteria for a single field, repeat this process for each selection criterion. To change or delete a criterion from the list, select the criterion to change or delete and click Change or Delete.
7. If you define multiple selection criteria on a single field, you can specify how Essbase combines the criteria, that is whether they should be connected logically with AND or OR. If you select And from the Boolean group, the field must match all the selection criteria. If you select Or from the Boolean group, the field must only match one of the selection criteria.

   Note: If you define selection criteria on more than one field, you can specify how Essbase combines the criteria. See "Defining Multiple Select and Reject Criteria" on page 605.

8. Click OK.

Rejecting Records

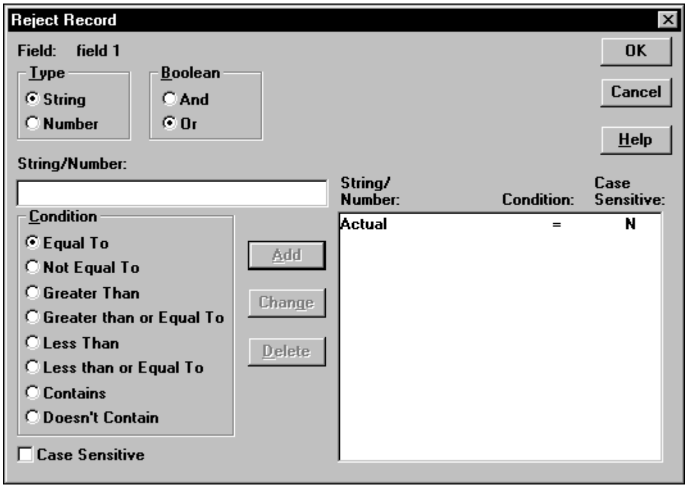
You can specify which records Essbase does not load into the database or use to build dimensions by setting rejection criteria. *Rejection criteria* are string and number conditions that must be met by one or more fields to cause Essbase to reject the record. If a field in the record does not meet the rejection criteria, Essbase loads the record. For example, to reject the Actual data from a data source and load only the Budget data, you could set a rejection criterion to reject records where the first field was Actual.

➤ To define rejection criteria:

1. Select the field to which to apply the criterion.

2. Select Record > Reject or click the Record Rejection button, , to open the Reject Record dialog box.

Figure 325: Reject Record Dialog Box



Reject Record

Field: field 1

Type

● String

○ Number

Boolean

○ And

● Or

String/Number:

Condition

● Equal To

○ Not Equal To

○ Greater Than

○ Greater than or Equal To

○ Less Than

○ Less than or Equal To

○ Contains

○ Doesn't Contain

□ Case Sensitive

String/ Number:	Condition:	Case Sensitive:
Actual	=	N

3. Set the field type as string or number by choosing the String or Number option. If the field is a string, you can define criteria that are case-sensitive by selecting the Case Sensitive box.

4. Enter the string or number upon which the criterion is based in the String/Number text box; for example, Actual.

5. Select how to evaluate the field by clicking one of the condition options in the Condition box; for example, Equal To.

6. To create multiple rejection criteria for a single field, add the first one by clicking Add and repeat this process for each rejection criterion. To change or delete a criterion from the list, select the criterion to change or delete and click Change or Delete.

7. If you define multiple rejection criteria on a single field, you can specify how Essbase combines the criteria, that is whether they should be connected logically with AND or OR. If you select And from the Boolean group, the field must match all the rejection criteria. If you select Or from the Boolean group, the field must only match one of the rejection criteria.

   Note: If you define rejection criteria on more than one field, you can specify how Essbase should combine the criteria. See “Defining Multiple Select and Reject Criteria” on page 605.

8. Click OK.

Defining Multiple Select and Reject Criteria

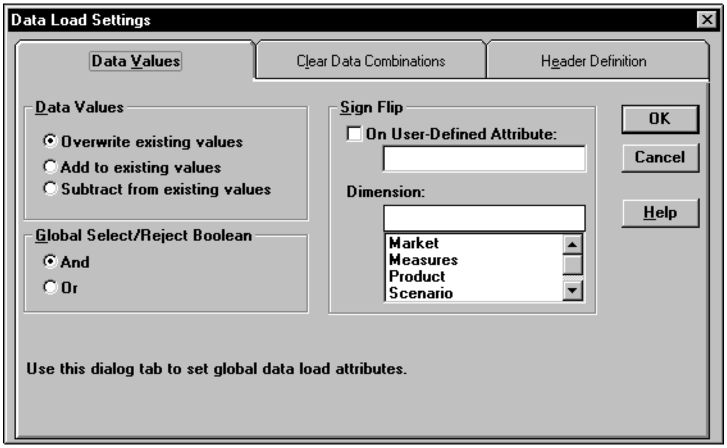
When you define select and reject criteria on multiple fields, you can specify how Essbase combines the rules across fields, that is, whether the criteria are connected logically with AND or OR. If you select And from the Boolean group, the fields must match all the selection or rejection criteria. If you select Or from the Boolean group, the fields must only match one of the selection or rejection criteria. Setting the global Boolean applies it to all select or reject operations in the rules file, for both data load and dimension-building fields.

Note: If your selection and rejection criteria apply to the same record (that is, you try to select and reject the same record), the record is rejected.

For example, to load only the New York, Actual data in a data source containing data for other markets and scenarios, create select criteria to select records that contain New York and Actual in the specified fields.

1. Select Options > Data Load Settings or click the Global Data Load Properties button, , to open the Data Load Settings dialog box. Click the Data Values tab.

Figure 326: Data Values Page: Global Select/Reject Boolean



Data Load Settings

Data Values | Clear Data Combinations | Header Definition

Data Values
● Overwrite existing values
○ Add to existing values
○ Subtract from existing values

Global Select/Reject Boolean
● And
○ Or

Sign Flip
□ On User-Defined Attribute:

Dimension:
Market
Measures
Product
Scenario

Use this dialog tab to set global data load attributes.

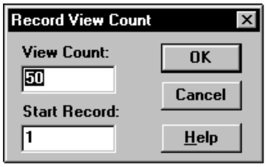
2. Select the Global Select/Reject Boolean operator to use, either And or Or.
3. Click OK.

Setting the Records Displayed

➤ To specify how many records Essbase displays in the Data Prep Editor and the first record to display:

1. Select Record > Record View Count to open the Record View Count dialog box.

Figure 327: Record View Count Dialog Box



Record View Count

View Count: 50

Start Record: 1

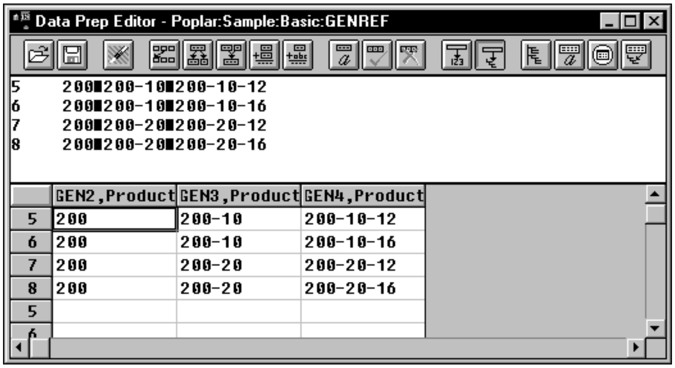
2. Enter the number of records to view in the View Count text box. The editor displays 50 records by default, but it can display anywhere from 1 to 200 records.
3. Enter the first record to view in the Start Record text box. Essbase skips the other records in the data source and displays the number of the record that you chose first in the Data Prep Editor. For example, if you enter `5` as the starting record, Essbase does not display records 1 through 4.

   Note: Essbase treats header records the same as data records when counting the records to skip.

   The highest record you can set as the first record is 38,527.

   Figure 328, for example, displays records 5 through 8.

   Figure 328: Displaying Record 5 through 8



```
5     200■200-10■200-10-12
6     200■200-10■200-10-16
7     200■200-20■200-20-12
8     200■200-20■200-20-16
```

	GEN2,Product	GEN3,Product	GEN4,Product
5	200	200-10	200-10-12
6	200	200-10	200-10-16
7	200	200-20	200-20-12
8	200	200-20	200-20-16
5			
6			

4. Click OK.

Validating and Saving Data Load Rules

This section describes how to validate data load rules to make sure they are correct, and save them so you can reuse them.

Before you validate a data load rules file, you must associate the rules file with an outline. This is usually the outline of the database into which to load the data. The rules file is not permanently associated with that outline, and you can associate it with any other outline in the future. See "Associating a Rules File with an Outline" on page 608.

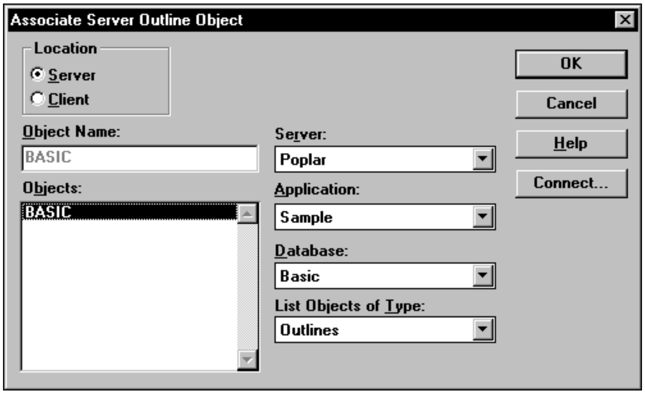
Rules files are validated to make sure the member and dimension mapping defined in the rules file maps to the outline. See Chapter 22, "Manipulating Fields Using a Rules File," to learn how to map fields to members. Validation cannot ensure that the data source will load properly.

Associating a Rules File with an Outline

➤ To associate a rules file with an outline:

1. Select Options > Associate Outline or click the Outline button, , to open the Associate Server Outline Object dialog box.

Figure 329: Associate Server Outline Object Dialog Box



Location: ● Server ○ Client

Object Name: BASIC

Objects: BASIC

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Outlines

2. Make sure the values for Object Name, Server, Application, and Database are correct. By default, Essbase assigns the values of the last outline associated with this rules file.

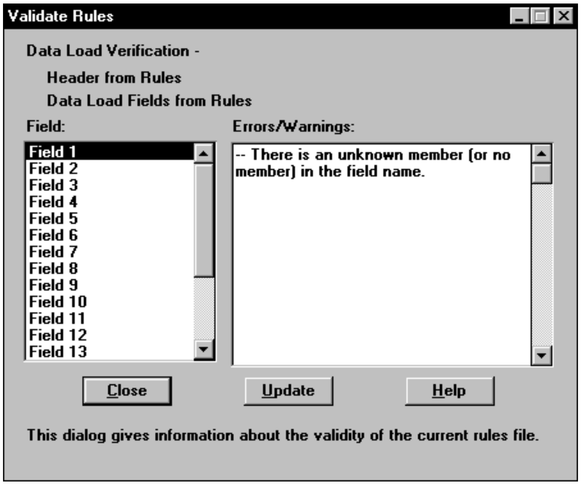
3. Select the outline to open in the Objects list box.
4. Click OK.

Validating a Rules File

To validate a rules file, make sure the rules file is open. If you are building dimensions by altering the data source (using dynamic references), make sure the data source is open as well.

1. Select Options > Validate or click the Validate Rules button, ✓, to validate the rules file against the outline. When Essbase finishes the validation, the Validate Rules dialog box appears. It contains information about the validation process, including which fields did not map to the outline.

Figure 330: Validate Rules Dialog Box



Validate Rules

Data Load Verification -

Header from Rules

Data Load Fields from Rules

Field:

Field 1
Field 2
Field 3
Field 4
Field 5
Field 6
Field 7
Field 8
Field 9
Field 10
Field 11
Field 12
Field 13

Errors/Warnings:

-- There is an unknown member (or no member) in the field name.

This dialog gives information about the validity of the current rules file.

2. If the rules file validates with no problems, you can use it to load data. See Chapter 20, “Introducing Data Loading,” for information on loading data.
3. If the rules file is not correct, you must fix it. Go to the field specified in the Validate Rules dialog that did not pass validation. In Figure 330, for example, Field 1 is invalid.

4. Make sure that the field name is valid. Check the following:

   - Is the field name spelled correctly?
   - Are the file delimiters correctly placed?
   - Is there a member in the field name?
   - Is the dimension name used in another field name or the header?
   - Are you using a member as a member combination in one place and a single member in another?
   - Is more than one field defined as a data field?
   - Is the dimension used for sign flipping in the associated outline?
   - Is the rules file associated with the correct outline?

   Note: Chapter 50, "Optimizing Data Loads," contains a complete list of data load rules file validation errors and possible causes.

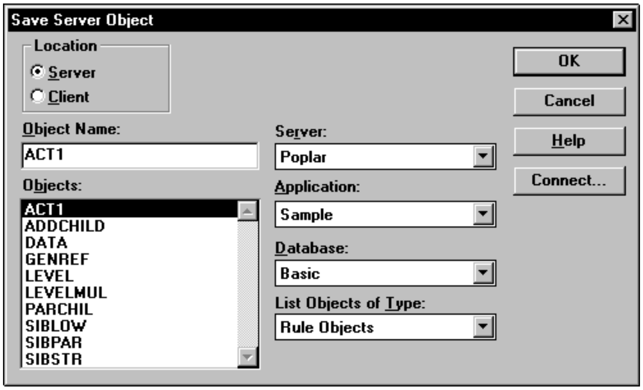
5. Validate the file again. Return to Step 1.

Saving Rules Files

➤ To save a rules file:

1. Select File > Save or click the Save button, 💾. If you haven't saved the rules file before, the Save Server Object dialog box appears.

Figure 331: Save Server Object Dialog Box



Location
● Server
○ Client

Object Name: ACT1

Objects:
ACT1
ADDCHILD
DATA
GENREF
LEVEL
LEVELMUL
PARCHIL
SIBLOW
SIBPAR
SIBSTR

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Rule Objects

2. Set the Server, Application, and Database to save the object in. You can also specify whether to save the object on the server or the client by choosing Server or Client in the Location group.
3. Enter the rules file name in the Object Name text box. The name must be a valid name in your operating system. In addition, the rules file name you specify must be eight or fewer alphanumeric characters. Essbase automatically adds an extension of `.RUL`. For example, you can name the rules file `ACT1.RUL` by entering `ACT1` in the Object Name text box.
4. Click OK.

Saving Under a Different Name

To save a rules file under a different name, select File > Save As to open the Save Server Object dialog box. Follow the same steps as for saving a rules file, but enter a different name.

Chapter

22

Manipulating Fields Using a Rules File

This chapter describes how to manipulate fields during a data load or dimension build using a rules file using Application Manager. Before you can manipulate fields, you must open the data source and set the file delimiters. After you set up the rules file, you must save and validate it. For more information on these topics, see Chapter 21, "Setting up a Rules File to Manipulate Records."

For more information about loading data using rules files, including prerequisites, see Chapter 23, "Performing and Debugging a Data Load."

This chapter contains the following sections:

Selecting Multiple Fields

You can select multiple fields and then set the properties for them. Essbase grays out any menu items and controls you cannot use when more than one field is selected.

➤ To select continuous fields:

1. Click the first field.
2. Shift-click the last field.

To select discontinuous fields, use one of the following methods:

- Select the fields in the first region, hold down Ctrl and drag the mouse along the fields in the second region.
- Click the first field, then hold down Ctrl and click the other fields.

Ignoring Fields

You can ignore all the fields in a column in your data source that do not map to the database. The fields still exist in the data source, but they are not loaded into the Essbase database. For example, you could have a column containing comment fields and you could choose to ignore the fields in that column for each record in the data source.

You can also ignore individual fields in your data source that match a string called a *token*. When you ignore fields based on string values, these fields are ignored everywhere they appear in the data source, not just in a single column.

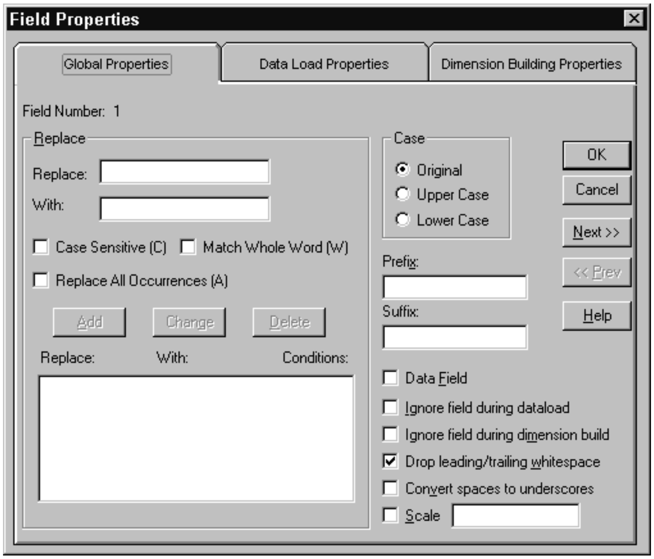
Ignoring All Fields in a Column

You can ignore an entire column, that is, ignore the field in a specified column for each record.

➤ To ignore all fields in a column:

1. Select the columns to ignore.
2. Choose Field > Properties or click the Field Properties button, , to open the Field Properties dialog box. Click the Global Properties tab.

Figure 332: Global Properties Page: Ignore Check Boxes



Field Properties

Global Properties | Data Load Properties | Dimension Building Properties

Field Number: 1

Replace
Replace:
With:
☐ Case Sensitive (C) ☐ Match Whole Word (W)
☐ Replace All Occurrences (A)
Replace: With: Conditions:

Case
● Original
○ Upper Case
○ Lower Case

Prefix:
Suffix:

☐ Data Field
☐ Ignore field during dataload
☐ Ignore field during dimension build
☑ Drop leading/trailing whitespace
☐ Convert spaces to underscores
☐ Scale

3. Check “Ignore field during data load” to ignore the field during a data load. Check “Ignore field during dimension build” to ignore the field during a dimension build. To ignore the field for both, check both boxes.
4. Click OK. The values in that field now appear grayed out in the Data Prep Editor to indicate that the field is ignored.
5. To hide ignored fields, choose View > Ignored Fields.

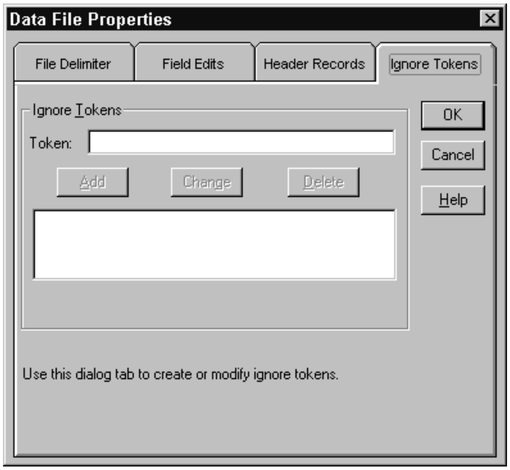
Ignoring Fields Based on String Matching

You can also ignore fields in your data source that match a string called a *token*. When you ignore fields based on string values, these fields are ignored everywhere they appear in the data source, not just in a single column.

➤ To ignore all fields in a data source that match a certain string:

1. Select Options > Data File Properties to bring up the Data File Properties dialog box. Click the Ignore Tokens tab. Token is another word for string.

   Figure 333: Ignore Tokens Page



Use this dialog tab to create or modify ignore tokens.

2. Enter the token to ignore in the Token entry field.
3. Click Add to add the token to the list.
4. Repeat steps 2 and 3 for each token to ignore.
5. To change or delete the tokens to ignore, select the token in the list and click Change or Delete.
6. Click OK.

Ordering Fields

You can change the order of fields in a data source by specifying their new position in the rules file. The data source is unchanged. The following sections describe:

- "Moving Fields" on page 617
- "Joining Fields" on page 618
- "Creating a New Field While Leaving Existing Fields Intact" on page 619
- "Splitting Fields" on page 621
- "Creating Additional Text Fields" on page 622
- "Undoing Field Ordering" on page 623

Note: Whenever you want to undo a single operation, choose Edit > Undo. To undo multiple field operations, see "Undoing Field Ordering" on page 623.

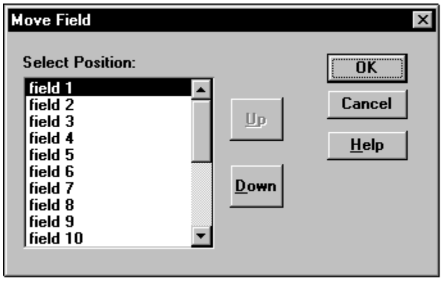
Moving Fields

➤ To move one or more fields:

1. Select the field to move. If you do not select a field, the currently active field is selected by default.

2. Choose Field > Move or click the Move button, , to open the Move Field dialog box.

Figure 334: Move Field Dialog Box



3. Click Up or Down to move the field to the proper position.
4. If you need to move another field, select the field and repeat step 3.
5. Click OK to close the dialog box.

Note: To undo a move, see "Undoing Field Ordering" on page 623.

CAUTION: In some instances, moved fields may appear to merge. This may occur if you have a data file similar to this structure:

```
1<tab>2<tab>3
1<tab>2<tab>(null)
```

If you move a field with empty cells and it is the last field in the data file, the field may merge with the field to its left when moved.

To prevent this, add a tab where the empty cell is.

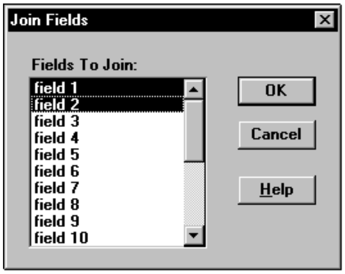
Joining Fields

You can join multiple fields into one field. For example, if you receive a data source with separate fields for the product number (100) and product family (-10), you must join those fields (100-10) to load them into the Sample Basic database.

1. Move the fields to join into the order in which you want to join them. If you do not know how to move fields, see "Moving Fields" on page 617.
2. Select the fields to join, for example 100 and -10. If you do not know how to select multiple fields, see "Selecting Multiple Fields" on page 614.

3. Choose Field > Join or click the Join button, , to open the Join Fields dialog box.

Figure 335: Join Fields Dialog Box



Join Fields

Fields To Join:

field 1
field 2
field 3
field 4
field 5
field 6
field 7
field 8
field 9
field 10

4. If you did not select the fields to join before opening the Join Fields dialog box, select the fields in the Fields to Join list box. The order in which fields are joined is determined by the order in which they appear in the list box.
5. Click OK.

   The selected fields merge into one. The new field is named after the first field in the join. The original fields are part of the joined field.

Note: To undo a join, see "Undoing Field Ordering" on page 623.

Creating a New Field While Leaving Existing Fields Intact

You can create a copy of a field or you can join two or more fields by putting the joined fields into a brand new field. This leaves the existing fields intact.

Creating a New Field By Joining Fields

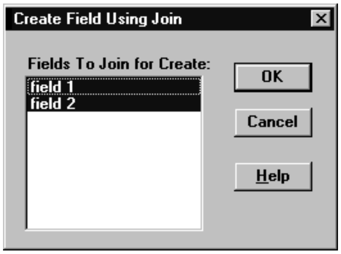
You may need to concatenate fields from your source data to create the member you want to define in your rules file.

For example, if you receive a data source with separate fields for the product number (100) and product family (-10), you must join those fields (100-10) to load them into the Sample Basic database. But suppose that you want to leave the 100 and -10 fields in the data source after the join, that is, the data source would contain three fields: 100, -10, and 100-10. To do this, create the new field using a join.

1. Select the fields to join, for example, 100 and -10. If you do not know how to select multiple fields, see "Selecting Multiple Fields" on page 614.

2. Choose Field > Create Using Join or click the Create Using Join button, , to open the Create Field Using Join dialog box.

Figure 336: Create Field Using Join Dialog Box

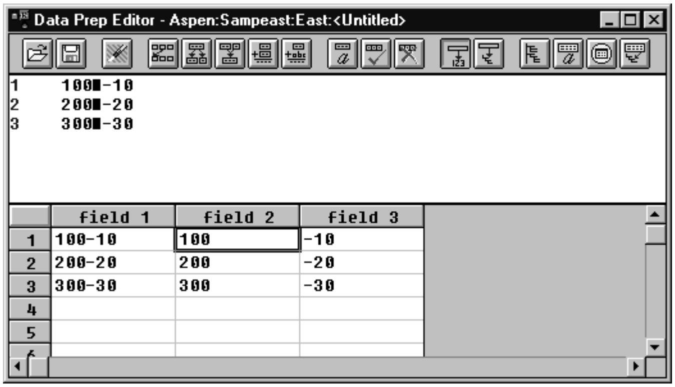


3. If you did not select the fields to join, select the fields in the Create Field Using Join dialog box.

4. Click OK.

   The new field displays to the left of the first field containing joined information. For example, in Figure 337, a new 100-10 field displays to the left of the existing 100 and -10 fields.

Figure 337: New 100-10 Field



```
1     100■-10
2     200■-20
3     300■-30
```

	field 1	field 2	field 3
1	100-10	100	-10
2	200-20	200	-20
3	300-30	300	-30
4			
5			

Copying Fields

You may need to create a new field in the rules file that is a copy of an existing one; for example, when you define a multilevel attribute dimension and associate attributes to members of a base dimension during the same dimension build.

1. Select the field to copy, for example, 12.
2. Choose Field > Create Using Join or click the Create Using Join button, , to open the Create Field Using Join dialog box.
3. Make sure the only field selected in the Create Field Using Join dialog box is the field you want to duplicate.
4. Click OK.

   The new field displays to the left of the field you selected to copy. You may need to move the field to another location. See "Moving Fields" on page 617.

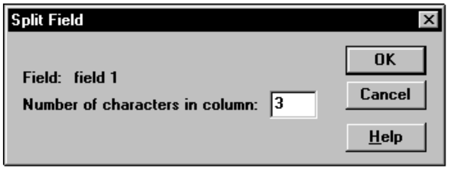
Splitting Fields

You can split a field into two fields. For example, if a data source for the Sample Basic database has a field containing UPC100-10-1, you could split the UPC out of the field and ignore it. To ignore a field, see "Ignoring All Fields in a Column" on page 615. Then 100-10-1, that is, the product number, is loaded.

➤ To split a field:

1. Select the field to split in the Data Prep Editor.
2. Choose Field > Split or click the Split button, , to open the Split Field dialog box.

   Figure 338: Split Field Dialog Box



Split Field

Field: field 1

Number of characters in column: 3

3. Enter the number of characters to split out.

   For example, to split the UPC out of the UPC100-10-1 field, split away the first three digits. Set the character position to `3`.

4. The field you split out displays to the left of the original field.

Note: To undo a split, see “Undoing Field Ordering” on page 623.

Creating Additional Text Fields

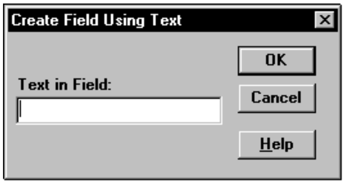
You can create a text field between two existing fields. You might do this to insert text between fields that are to be joined. For example, if you had two fields containing 100 and 10-1, you could insert a text field between them with a dash and then join them to create the 100-10-1 member of the Product dimension.

➤ To create a new text field:

1. Select the field to put the new field in front of, for example, 10-1.

2. Choose Field > Create Using Text or click the Create Using Text button, , to open the Create Field Using Text dialog box.

   Figure 339: Create Field Using Text Dialog Box



3. Enter the text to put into the new field, for example, a hyphen (-).

4. Click OK. The new field displays to the left of the selected field.

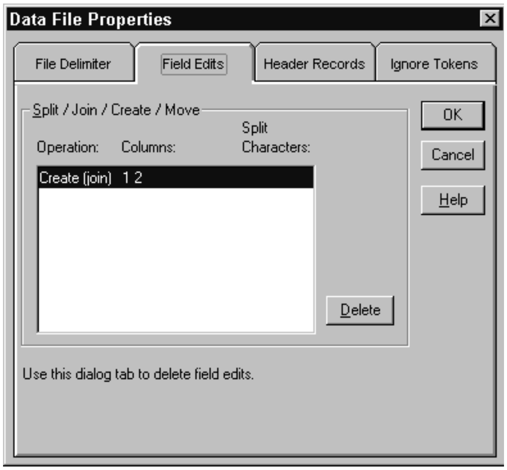
Note: To undo a field you created using text, see “Undoing Field Ordering” on page 623.

Undoing Field Ordering

You can undo the last field operation you performed such as move, join, split, or create using text, using the Edit > Undo command. You can also undo field operations even if you have performed other actions. Undoing field operations is sequential; you must undo them from the last operation to the first.

1. Select the field or fields. If you do not know how to select multiple fields, see “Selecting Multiple Fields” on page 614.
2. Choose Options > Data File Properties or click the Data File Properties button, , to open the Data File Properties dialog box. Click the Field Edits tab.

Figure 340: Field Edits Page



Create (join) 1 2

Use this dialog tab to delete field edits.

3. Select the operation to delete in the Operation list and click Delete. Operations must be deleted in reverse order, that is, by deleting the last operation first.
4. When you are finished, click OK.

Mapping Fields to Member Names

To load a data source, you must specify how the fields in the data source map to the dimensions in your database. Rules files can translate fields in the data source to match member names each time the data source is loaded without changing the data source. The rules file does the following:

- Maps member fields in the data source to members in the database
- Maps data fields in the data source to member names or member combinations (such as Jan, Actual) in the database

You can define rules to:

- Name, or translate, the fields in the data source to match members in the database. See "Naming Fields" on page 624.
- Replace strings, or translate, the fields in the data source to match members in the database. See "Replacing Text Strings" on page 626.

Naming Fields

Use a rules file to name data source fields to match Essbase dimension names during a data load. The data source is not changed.

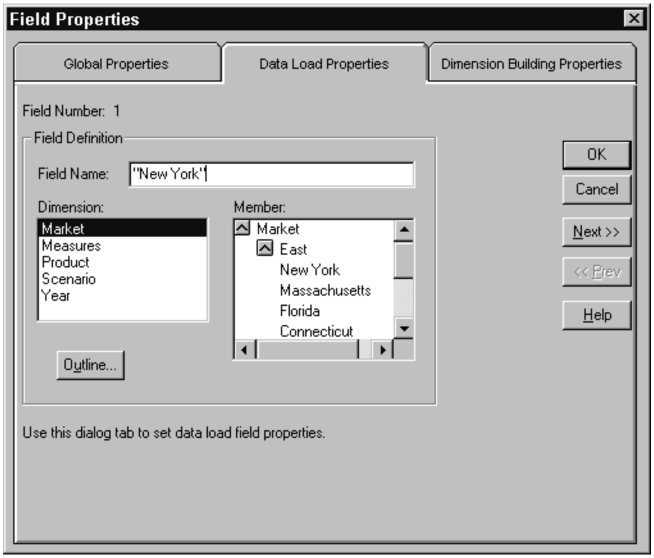
Note: When you open an SQL data source, the fields default to the SQL data source column names. If these names are the same as your Essbase dimensions, you do not have to perform any field mapping.

➤ To name a field:

1. Select the field to name in the Data Prep Editor.

2. Choose Field > Properties or click the Field Properties button, , to open the Field Properties dialog box. Click the Data Load Properties tab.

Figure 341: Data Load Properties Page



Field Number: 1

Field Name: "New York"

Dimension: Market, Measures, Product, Scenario, Year

Member: Market, East, New York, Massachusetts, Florida, Connecticut

Use this dialog tab to set data load field properties.

3. Type the member name into the Field Name text box or paste it by clicking on the appropriate member in the Dimension and Member lists. If the Dimension and Member lists are empty, click the Outline button to select a database outline.

   Note: If you enter a member name with a space in it, such as New York, be sure to put quotation marks around the member name. If you click the member name in the Member list box, Essbase automatically puts quotation marks around member names with spaces in them.

4. To name the next field in the Data Prep Editor, click the Next button. To change the previous field, click the Prev button. This saves all changes before going to the next field.

5. When you are finished, click OK.

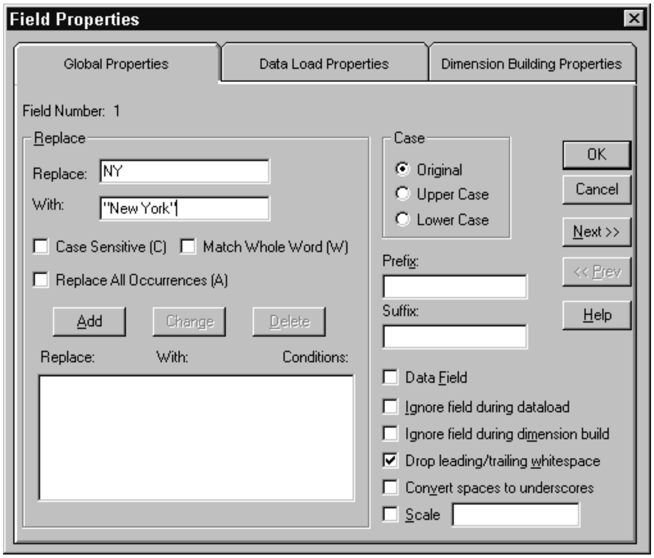
Replacing Text Strings

Use a rules file to replace text strings so that the fields map to Essbase member names during a data load. The data source is not changed. For example, if the data source abbreviates New York to NY, you could have the rules file replace each NY with New York while loading the data.

1. Select the field or fields containing the text string you want to change.

2. Choose Field > Properties or click the Field Properties button, , to open the Field Properties dialog box. Click the Global Properties tab.

Figure 342: Global Properties Page: Replace Box



Field Properties

Global Properties | Data Load Properties | Dimension Building Properties

Field Number: 1

Replace

Replace: NY

With: "New York"

□ Case Sensitive (C) □ Match Whole Word (W)

□ Replace All Occurrences (A)

Case

● Original

○ Upper Case

○ Lower Case

Prefix:

Suffix:

□ Data Field

□ Ignore field during dataload

□ Ignore field during dimension build

☑ Drop leading/trailing whitespace

□ Convert spaces to underscores

□ Scale

3. Enter the text string you want to replace in the Replace text box. For example, NY. You must enter a text string in the Replace text box. You cannot replace blank fields with text directly. To replace blank fields with text, see “Replacing an Empty Field with Text” on page 627.

4. Enter the text to replace it with in the With text box; for example, New York. You can leave the With text box empty, because you can replace a text string with an empty string.

5. Specify if the replacement operation should:
   - Be case-sensitive; that is, only replace text strings that match the capitalization of the string in the Replace text box.
   - Replace the text string only when it occurs as a whole word. For example, to replace the 10 in the string 100 10 1 with an A, setting the Match Whole Word option changes the string to 100 A 1. Not setting the Match Whole Word option changes the string to A0 10 1.
   - Replace all occurrences of the string. For example, if you replaced all occurrences of 10 in the string 100 10 1 with an A, the string changes to A0 A 1. By default, Essbase only changes the first occurrence.
6. Click Add to add the replacement operation to the list. To change an existing operation in the list, select the operation and click Change. To delete an operation from the list, select the operation and click Delete.
7. To move to the next column, click the Next button. To move to the previous column, click the Prev button.

   Note: The Next and Prev buttons only work if a single field is selected.
8. Click OK.

Replacing an Empty Field with Text

You may want to replace empty fields in a column with text. If, for example, empty fields in the column represent default values, you could insert the default values to replace the empty ones or insert `#MI` to represent missing values.

Replacing an empty field with text requires several steps. To replace an empty field with text:

1. Select the column containing the empty fields you want to replace.
2. Choose Field > Create Using Text or click the Create Using Text button, .
3. Enter the text to put in the new field. Enter a dummy string that is not in the selected column, such as “temp.” Click OK.
4. Join the new field with the column containing the fields to replace. Select both columns, choose Field > Join, and click OK. The previously blank fields now contain the dummy string you entered in step 3; for example, temp.

5. Select the column containing the dummy string and choose Field > Properties or click the Field Properties button, . Click the Global Properties tab.
6. Enter the text string you want to replace in the Replace text box. This should be the dummy string you entered in step 3; for example, temp. Now enter the text to replace it with in the With text box. This should be the final value you want to put in the fields, for example, default. Select the Match Whole Word.
7. Click Add and then click OK.
8. Now replace the extra dummy strings in the column with nothing. Choose Field > Properties or click the Field Properties button, . Click the Global Properties tab. Enter the dummy string in the Replace text box; for example, temp. Enter nothing in the With text box. Make sure the Match Whole Word option is not checked. Click Add and then click OK.

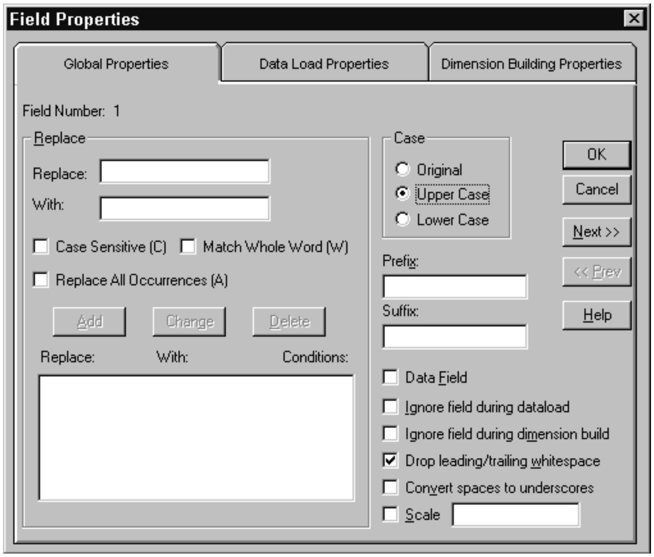
Changing the Case of Fields

Use a rules file to change the case of a field so the field maps to Essbase member names during a data load. The data source is not changed. For example, if the data source capitalizes a field that is in lower case in the database, you could change the field to lower case; for example, from JAN to jan.

1. Select the field or fields containing the text string you want to change.

2. Choose Field > Properties or click the Field Properties button, , to open the Field Properties dialog box. Click the Global Properties tab.

Figure 343: Global Properties Page: Case Box



Field Number: 1

Replace:

With:

☐ Case Sensitive (C) ☐ Match Whole Word (W)

☐ Replace All Occurrences (A)

○ Original
● Upper Case
○ Lower Case

Prefix:

Suffix:

☐ Data Field

☐ Ignore field during dataload

☐ Ignore field during dimension build

☑ Drop leading/trailing whitespace

☐ Convert spaces to underscores

☐ Scale

3. Choose the case to change the field to from the Case box, either Original, Upper Case, or Lower Case.

4. To move to the next column, click the Next button. To move to the previous column, click the Prev button.

   Note: The Next and Prev buttons only work if a single field is selected.

5. Click OK.

Dropping Leading and Trailing White Space

You can drop leading or trailing white space from around fields in your data source. A field value containing leading or trailing white spaces does not map to a member name, even if the name within the white spaces is an exact match.

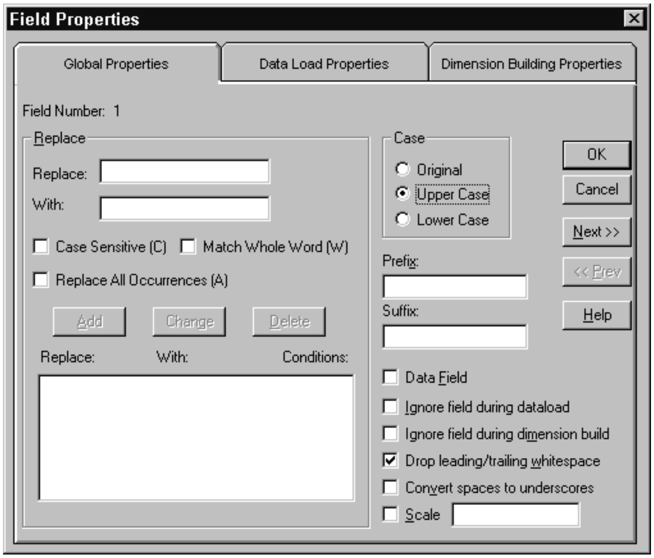
By default, Essbase drops leading and trailing white space.

➤ To drop leading or trailing white space:

1. Select the field or fields. If you do not know how to select multiple fields, see “Selecting Multiple Fields” on page 614.

2. Choose Field > Properties or click the Field Properties button, , to open the Field Properties dialog box. Click the Global Properties tab.

Figure 344: Global Properties Page: Drop Leading/Trailing Whitespace Check Box



Field Properties

Field Number: 1

Replace

Replace:

With:

□ Case Sensitive (C) □ Match Whole Word (W)

□ Replace All Occurrences (A)

Replace: With: Conditions:

Case

○ Original

● Upper Case

○ Lower Case

Prefix:

Suffix:

□ Data Field

□ Ignore field during dataload

□ Ignore field during dimension build

☑ Drop leading/trailing whitespace

□ Convert spaces to underscores

□ Scale

3. By default, the “Drop leading/trailing whitespace” box is checked. If it is not already checked, check it.

4. Click OK.

Converting Spaces to Underscores

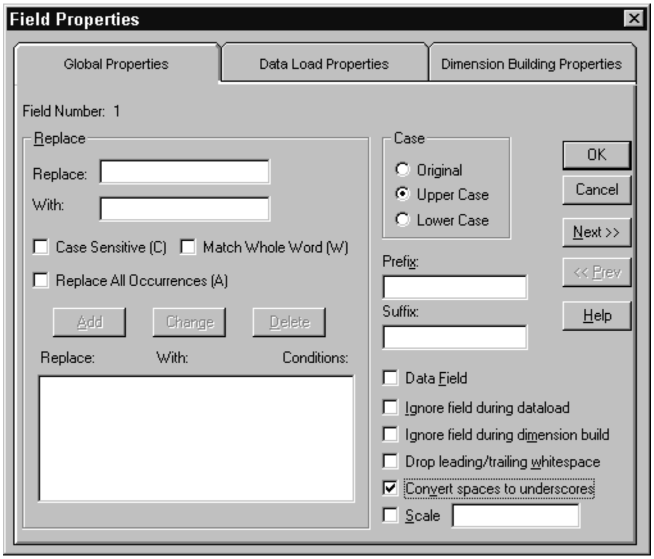
You can convert spaces in fields in the data source to underscores to make them match member names in the database.

➤ To convert spaces to underscores:

1. Select the field or fields. If you do not know how to select multiple fields, see "Selecting Multiple Fields" on page 614.

2. Choose Field > Properties or click the Field Properties button, , to open the Field Properties dialog box. Click the Global Properties tab.

Figure 345: Global Properties Page: Convert Spaces to Underscores Check Box



Field Number: 1

Case: ● Upper Case

☑ Convert spaces to underscores

3. Check Convert spaces to underscores.

4. Click OK.

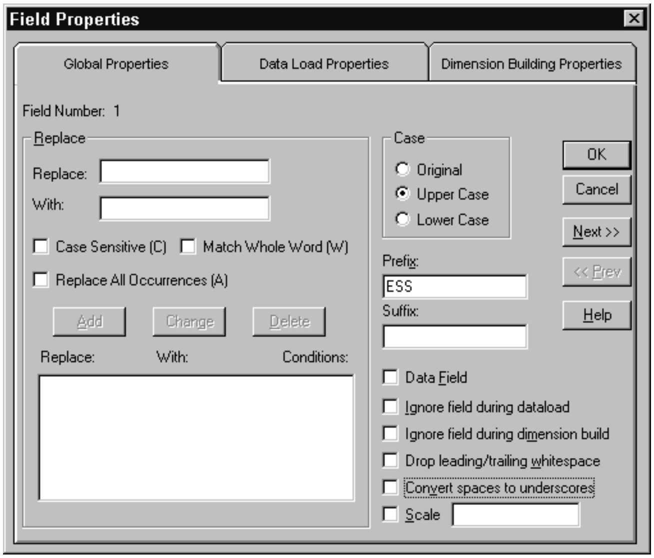
Adding Prefixes or Suffixes to Field Values

You can add prefixes or suffixes to each field value in the data source. For example, you could add ESS as the prefix to all Essbase member names during the data load.

1. Select the field or fields. If you do not know how to select multiple fields, see "Selecting Multiple Fields" on page 614.

2. Choose Field > Properties or click the Field Properties button, , to open the Field Properties dialog box. Click the Global Properties tab.

Figure 346: Global Properties Page: Prefix and Suffix Text Boxes



Field Number: 1

Case: ○ Original ● Upper Case ○ Lower Case

Prefix: ESS

Suffix:

□ Case Sensitive (C) □ Match Whole Word (W)

□ Replace All Occurrences (A)

□ Data Field

□ Ignore field during dataload

□ Ignore field during dimension build

□ Drop leading/trailing whitespace

□ Convert spaces to underscores

□ Scale

3. Enter the prefix or suffix in the Prefix or Suffix text box.

4. Click OK.

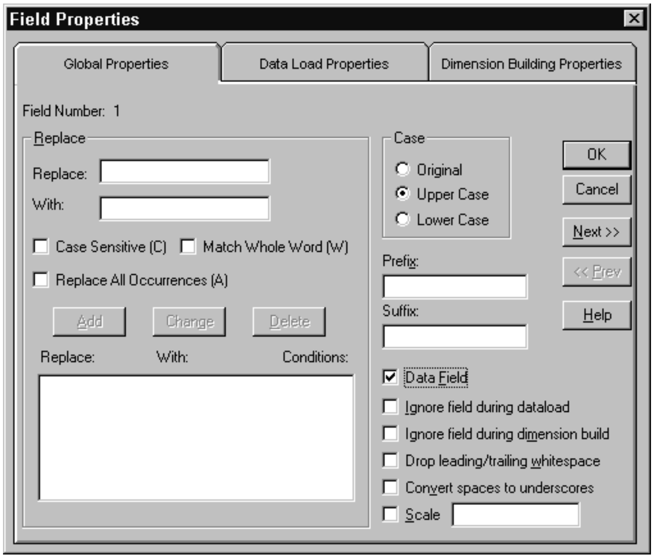
Defining a Column as a Data Field

Some data sources contain a single data column that does not map to a specific member. When this occurs, you must define the data column as a data field. You can only define one field in a record as a data field.

➤ To define a column as a data field:

1. Select the field at the top of the data column.
2. Choose Field > Properties or click the Field Properties button, , to open the Field Properties dialog box. Click the Global Properties tab.

Figure 347: Global Properties Page: Data Field Check Box



Field Properties

Global Properties

Field Number: 1

Case: ○ Original ● Upper Case ○ Lower Case

☑ Data Field

☐ Ignore field during dataload

☐ Ignore field during dimension build

☐ Drop leading/trailing whitespace

☐ Convert spaces to underscores

☐ Scale

☐ Case Sensitive (C) ☐ Match Whole Word (W)

☐ Replace All Occurrences (A)

3. Check Data Field.
4. Click OK.

Changing Data Values

By default, Essbase overwrites the existing values in the database, but the following sections describe:

- "Adding to and Subtracting from Existing Values" on page 634
- "Clearing Existing Data Values" on page 635
- "Scaling Data Values" on page 637

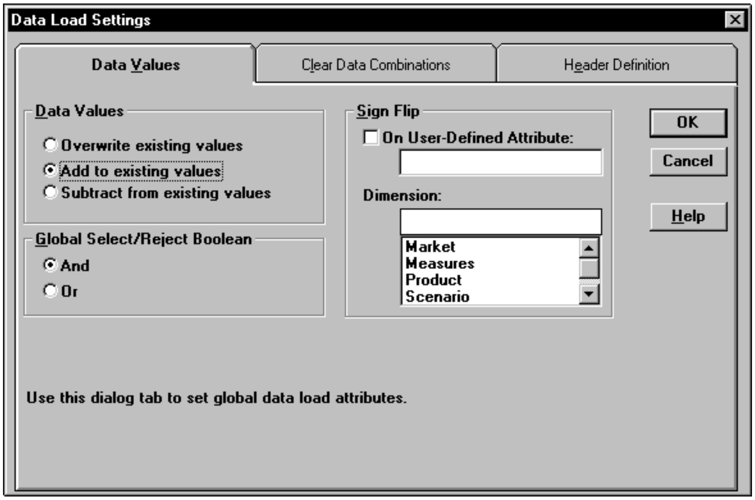
Adding to and Subtracting from Existing Values

You can add or subtract the values in incoming records to existing values in an Essbase database. For example, if you load weekly values, you can add them to create monthly values in the database.

➤ To add or subtract existing values:

1. Select the field to add to or subtract from.
2. Choose Options > Data Load Settings or click the Global Data Load Properties button, , to open the Data Load Settings dialog box.
3. Click the Data Values tab.

Figure 348: Data Values Page: Data Values Box



Data Load Settings

Data Values
○ Overwrite existing values
● Add to existing values
○ Subtract from existing values

Global Select/Reject Boolean
● And
○ Or

Sign Flip
□ On User-Defined Attribute:

Dimension:
Market
Measures
Product
Scenario

Use this dialog tab to set global data load attributes.

4. Choose “Add to existing values” to add the values or “Subtract from existing values” to subtract the values.

   CAUTION: Using this option makes it more difficult to recover if the database crashes while loading data, although Essbase lists the number of the last row committed in the application log. For more information, see “Understanding the Contents of the Application Log” on page 1212.

   To solve this problem, as a Database Transaction setting, set the Commit Row value as 0. This causes Essbase to view the entire load as a single transaction and commit the data only when the load is complete. For more information, see “Transactions” on page 1132.

5. Click OK.

Clearing Existing Data Values

You can clear existing data from the database before loading new values. By default, Essbase overwrites the existing values in the database with the new values in the data source. If you are adding and subtracting the data values, however, Essbase adds or subtracts the new values with the existing ones.

Before adding or subtracting new values, you need to make sure the existing values are correct. If you are loading the first set of values into the database, you must make sure there is no existing value.

For example, let us assume that the Sales figures for January are calculated by adding together the values for each week in January. That means:

```
January Sales = Week 1 Sales + Week 2 Sales + Week 3 Sales + Week 4 Sales
```

When you load Week 1 Sales, you must make sure that the value for January Monthly Sales is cleared in the database. If there is an existing value, Essbase performs the following calculation:

```
January Sales = Existing Value + Week 1 Sales + Week 2 Sales + Week 3 Sales
+ Week 4 Sales
```

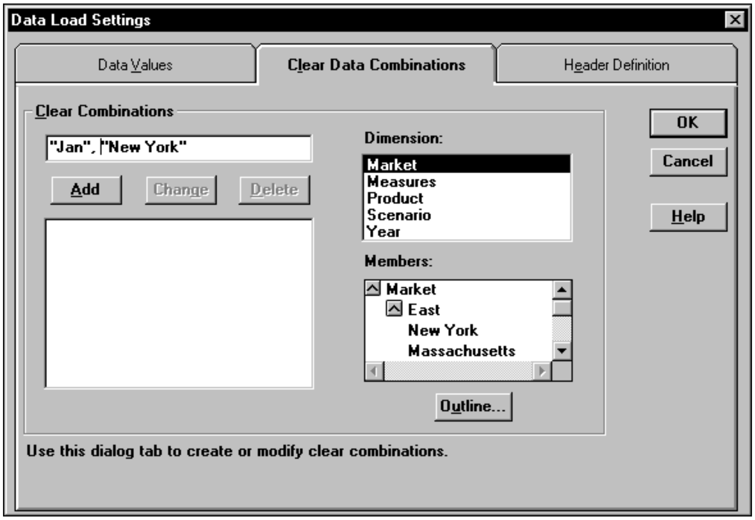
You can also clear data from fields that are not part of the data load. For example, if a data source contained data for January, February and March and you only wanted to load the March data, you could clear the January and February data.

Note: If you are using transparent partitions, you can clear the values using just the same steps as for clearing data in a local database.

➤ To clear existing values:

1. Choose Options > Data Load Settings or click the Global Data Load Properties button, , to bring up the Data Load Settings dialog box. Click the Clear Data Combinations tab.

Figure 349: Clear Data Combinations Page



Clear Combinations: "Jan", "New York"

Dimension: Market

Use this dialog tab to create or modify clear combinations.

2. Enter the member combinations to clear in the Clear Combinations text box or click the dimensions and members in their lists. If the lists are empty, click Outline to associate the rules file with an outline. See "Associating a Rules File with an Outline" on page 608 for more information on associating a rules file with an outline.

   You can enter Essbase functions in the Clear Combinations text box. For example, you could clear all descendants of Massachusetts by entering `@IDESCENDANTS(Massachusetts)`. For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

   Note: You must separate member combinations with a comma.

3. Click Add to add the member combination to the list.

   To change a member combination that is in the list, select the item in the list and click Change. It appears in the Clear Combinations text box.

   To delete member combinations from the list, select them and click Delete.

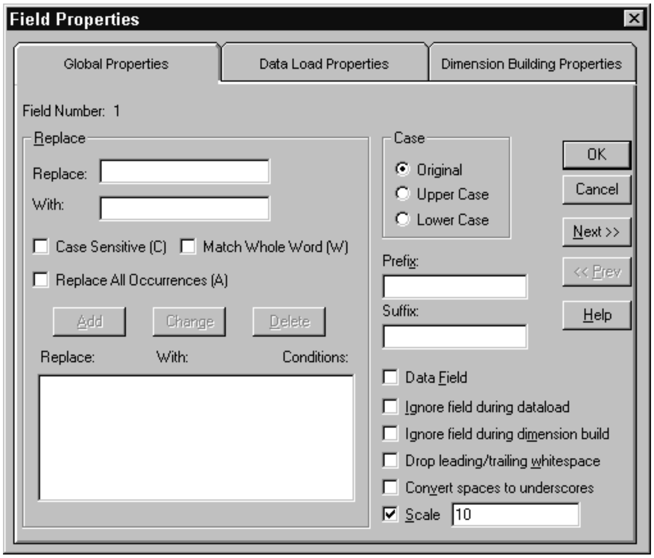
Scaling Data Values

You can scale data values if the values in the data source are not in the same scale as the values in the database. For example, the data source could track Sales in hundreds while the database tracks them in thousands. In this case, you would want to multiply the incoming values by 10.

1. Select the field to scale.

2. Choose Field > Properties or click the Field Properties button, , to open the Field Properties dialog box. Click the Global Properties tab.

Figure 350: Global Properties Page: Scale Check Box



Field Properties

Field Number: 1

Replace

Replace:

With:

□ Case Sensitive (C) □ Match Whole Word (W)

□ Replace All Occurrences (A)

Replace: With: Conditions:

Case

● Original

○ Upper Case

○ Lower Case

Prefix:

Suffix:

□ Data Field

□ Ignore field during dataload

□ Ignore field during dimension build

□ Drop leading/trailing whitespace

□ Convert spaces to underscores

☑ Scale 10

3. Check Scale. After Scale is checked, you must enter a number in the Scale text box.

4. Enter the value to multiply by in the Scale text box. For example, enter 10 to increase the value by 10 times; enter 0.1 to decrease the value by 10 times.

5. To move to the next column, click the Next button. To move to the previous column, click the Prev button.

Note: The Next and Prev buttons only work if a single field is selected.

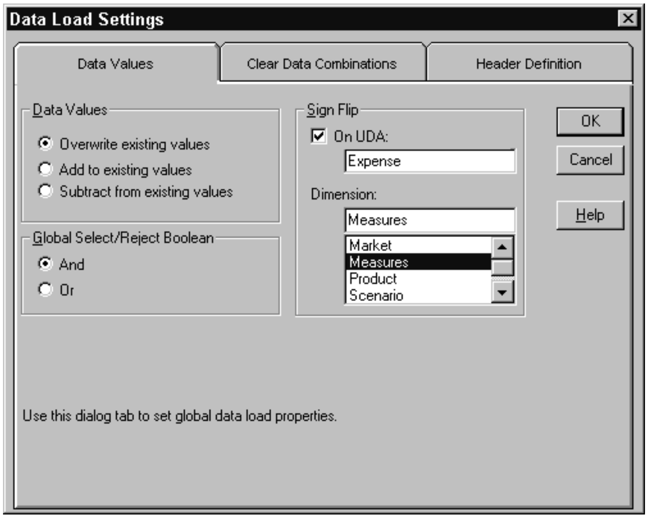
Flipping Field Signs

You can reverse or flip the value of a data field by flipping its sign. Sign flips are based on UDAs (user-defined attributes) in the outline. When loading data into the Accounts dimension, for example, you could specify that any record whose Accounts member had a UDA of Expense should change from a plus sign to a minus sign. You set UDAs in the Outline Editor. See Chapter 7, “Creating and Changing Database Outlines,” for more information on user-defined attributes.

➤ To set sign flipping:

1. Choose Options > Data Load Settings or click the Global Data Load Properties button, , to open the Data Load Settings dialog box. Click the Data Values tab.

Figure 351: Data Values Page



Data Values
- ● Overwrite existing values
- ○ Add to existing values
- ○ Subtract from existing values

Global Select/Reject Boolean
- ● And
- ○ Or

Sign Flip
- ☑ On UDA: Expense
- Dimension: Measures
- Market, Measures, Product, Scenario

Use this dialog tab to set global data load properties.

2. Check On UDA under Sign Flip.
3. Enter the UDA, for example, Expense. Use the Outline Editor to get a list of UDAs. See Chapter 7, “Creating and Changing Database Outlines,” for more information.
4. Enter the dimension or click the dimension in the Dimension list. If there are no dimensions in the list, click the Outline button, , to associate an outline with the rules file. See “Associating a Rules File with an Outline” on page 608 for more information.
5. Click OK.

Chapter

23

Performing and Debugging a Data Load

This chapter describes how to load data from one or more external data sources to your OLAP Server using the Application Manager. It shows you how to use free-form or rules file data sources to load data or build dimensions dynamically.

You can load data without updating the outline, you can update the outline without loading data, or you can do both operations simultaneously.

This chapter contains the following sections:

Tip: Use the LOADDATA or UPDATEFILE commands in ESSCMD to load data without a rules file. See the *Technical Reference* in the `docs` directory for information about these commands. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Prerequisites for Loading Data and Building Dimensions

To start loading data or building dimensions, you must have:

- An Essbase database into which to load the data or build the outline.
- A connection to your server.
- Valid data sources:
  - Microsoft Excel files with the `.XLS` extension, Version 4.0 and higher. You must load Microsoft Excel files Version 5.0 and higher as client objects or files in the file system.
  - Lotus 1-2-3 files with the `.WKS`, `.WK1`, `.WK3`, or `.WK4` extension
  - Spreadsheet audit log files
  - ASCII text files (flat files) from ASCII backups or external sources
  - Essbase export files. For information about reloading exported data, see "Reloading Exported Data" on page 1254.
  - SQL data sources
- If you are not using a rules file for loading data, a data source correctly formatted for free-form data loading. See "Rules for Free-Form Data Sources" on page 582.
- If you are using a rules file for loading data, a rules file validated for data load. For information about defining data load rules files, see Chapter 20, "Introducing Data Loading."
- If you are performing a dimension build, a rules file validated for dimension build. For more information about defining dimension build rules, see Chapter 18, "Introducing Dynamic Dimension Building" and Chapter 19, "Building Dimensions Using a Rules File."

  Note: You must use a rules file to load SQL data or to build dimensions and members dynamically.

Choosing the Data Sources Using Application Manager

You can select data sources using the Application Manager or Windows. For a list of valid data sources, see “Prerequisites for Loading Data and Building Dimensions” on page 642.

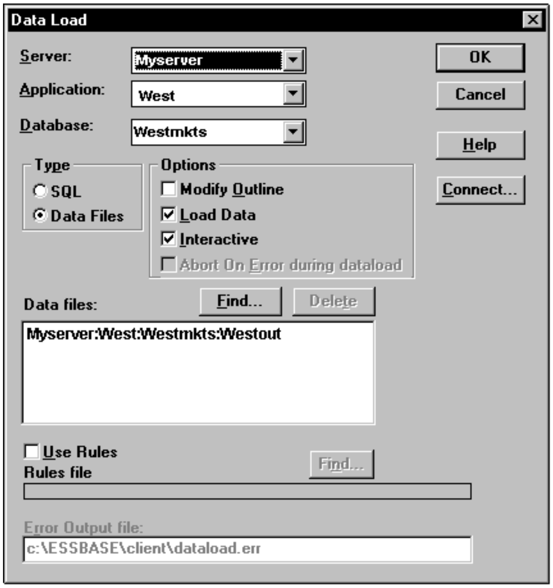
Make sure you are connected to the server before you specify the data sources.

Tip: Use the LOADDB command in ESSCMD to perform this task. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, “Automating the Production Environment” for information about ESSCMD.

➤ To select a data source using Application Manager:

1. Make sure you are connected to a server:
2. Click the Application Desktop window.
3. Select Database > Load Data to specify how to load data or build dimensions. The Data Load dialog box is displayed.

Figure 352: Data Load Dialog Box



Data Load

Server: Myserver

Application: West

Database: Westmkts

Type
○ SQL
● Data Files

Options
□ Modify Outline
☑ Load Data
☑ Interactive
□ Abort On Error during dataload

Data files:
Myserver:West:Westmkts:Westout

□ Use Rules

Rules file

Error Output file:
c:\ESSBASE\client\dataload.err

4. Click Connect to open the Hyperion Essbase System Login dialog box. Enter the correct values and click OK. Now you are ready to start:

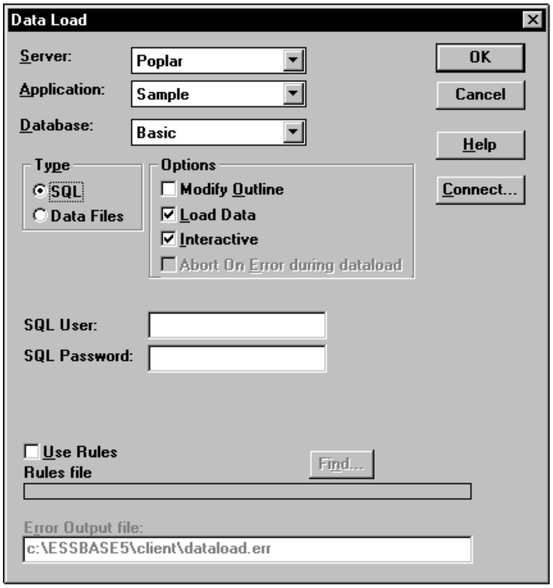
   - “Choosing SQL Data Sources” on page 644
   - “Choosing Text or Spreadsheet Files” on page 645

Choosing SQL Data Sources

➤ To load SQL data into an Essbase database:

1. Click the Application Desktop window.
2. Select the application and database to load the data into or build dimensions for.
3. Select Database > Load Data. The Data Load dialog box displays.
4. To access SQL data, you must first connect to the SQL data source. Select the SQL option. The Data Load dialog box changes to look like the one shown in Figure 353.

Figure 353: SQL Data Load Dialog Box



Data Load

Server: Poplar

Application: Sample

Database: Basic

Type

● SQL
○ Data Files

Options

☐ Modify Outline
☑ Load Data
☑ Interactive
☐ Abort On Error during dataload

SQL User:

SQL Password:

☐ Use Rules

Rules file

Error Output file: c:\ESSBASE5\client\dataload.err

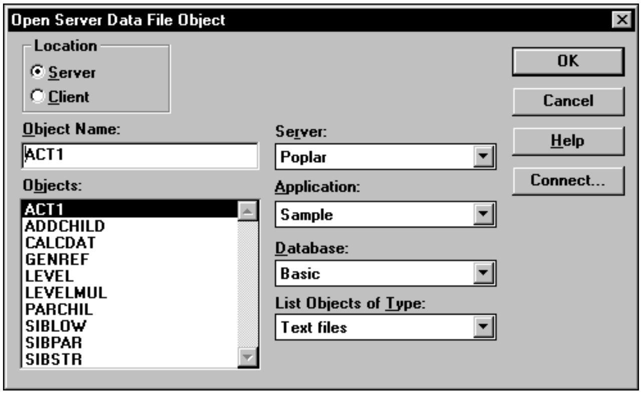
5. If your SQL data source requires you to enter your user name and password, enter them. All other connection information is specified in the rules file.
6. See "Using a Rules File with the Data Source" on page 648 to finish, because you must use a rules file to load SQL data sources.

Choosing Text or Spreadsheet Files

➤ To select text or spreadsheet files:

1. Click the Application Desktop window.
2. Select the application and database to load the data into or build dimensions for.
3. Select Database > Load Data. The Data Load dialog box displays.
4. Click the Data Files option button if it is not already selected.
5. Click Find to select a text or spreadsheet file to load. The Open Server Data File Object dialog box is displayed.

Figure 354: Open Server Data File Object Dialog Box



Open Server Data File Object

Location
● Server
○ Client

Object Name: ACT1

Objects:
ACT1
ADDCHILD
CALCDAT
GENREF
LEVEL
LEVELMUL
PARCHIL
SIBLOW
SIBPAR
SIBSTR

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Text files

6. Make sure the appropriate server, application, and database are selected from their respective lists.

7. Specify the location of the file by clicking either the Server or Client button.

   If you select Server, the data source to load must reside in the database directory under `\essbase\app\`*`application_name`*`\`*`database_name`*, where *application_name* and *database_name* represent the name of your application and database. Type the name of the data source in the Object Name text box or select it from the Objects list box. In Figure 354, for example, you could select ACT1.

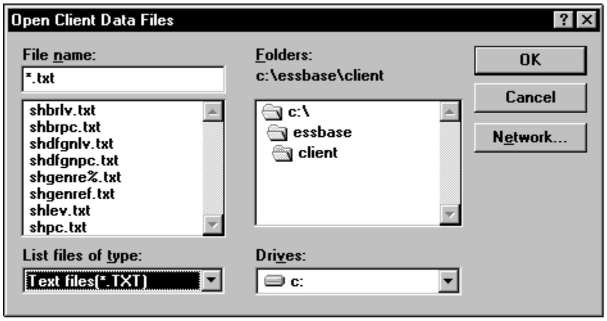
   If you select Client, the file may reside in either the application or database directory under `\essbase\client` or on the drives accessible from the client file system. Click File System to select a file from a standard Open Client Data Files dialog box. Select the file to open, for example, `asymm.xls` in the `\essbase\client\sample` directory.

   To select multiple files, hold down the Ctrl key and click the files.

Note: `essbase` is the default directory specified during installation. You may have specified a different default directory.

Load Microsoft Excel files Version 5.0 and higher as client objects or files in the file system, not as server objects.

Figure 355: Open Client Data Files Dialog Box



File name: *.txt

shbrlv.txt
shbrpc.txt
shdfgnlv.txt
shdfgnpc.txt
shgenre%.txt
shgenref.txt
shlev.txt
shpc.txt

Folders: c:\essbase\client

c:\
essbase
client

List files of type: Text files(*.TXT)

Drives: c:

8. Click OK. Return to the Data Load dialog box. Now you can to specify how to load the data or build dimensions.

Choosing the Data Sources Using Windows

You can select the data sources using Application Manager, the Windows File Manager or the Windows Explorer. For a list of valid data sources, see "Prerequisites for Loading Data and Building Dimensions" on page 642.

Make sure you are connected to the server before you select the data sources.

➤ To select a list of files:

1. Open the Windows File Manager or Explorer. Arrange your windows so that either the Application Manager or its icon is visible.
2. Locate and select the desired data sources from the Table 32 on page 647.
3. Drag the selected files from the File Manager or Explorer window to the Application Manager and release the mouse button. The Data Load dialog box displays. This is the dialog box where you specify how to load data or build dimensions.

   Note: If the data source contains blank fields for data values, replace them with `#MI` or `#MISSING`. Otherwise, the data will not load correctly. To replace a blank field with `#MI` or `#MISSING`, see "Replacing an Empty Field with Text" on page 627.

Table 32: Locating Data Sources

Task	**Action**
Select one file	Click the file name.
Select several files	Hold the Ctrl key while clicking on the files.
Select a range of files	Click the first file, then hold down the Shift key and select the last file in the range.

Specifying How to Load Data or Build Dimensions

After you select the data sources, specify how Essbase loads those data sources and whether to build dimensions dynamically using the Data Load dialog box. If you have not chosen your data sources yet, see "Choosing the Data Sources Using Application Manager" on page 643 or "Choosing the Data Sources Using Windows" on page 647.

You can set the following options:

- Modify Outline.
- Load Data

If you are loading data without a rules file, skip to "Setting a Load or Build Error Log" on page 651.

Using a Rules File with the Data Source

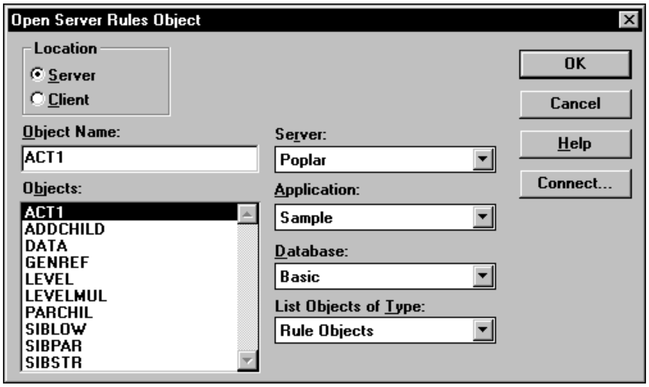
Rules files perform operations on the data as it is loaded, such as moving fields or building new dimensions.

➤ To build dimensions or load SQL data, you must use a rules file.

1. Select your data sources. If you have not chosen your data sources yet, see "Choosing the Data Sources Using Application Manager" on page 643 or "Choosing the Data Sources Using Windows" on page 647.
2. From the Data Load dialog box, select Use Rules.

3. Click the Find button to open the Open Server Rules Object dialog box.

Figure 356: Open Server Rules Object Dialog Box



Location
● Server
○ Client

Object Name: ACT1

Objects:
ACT1
ADDCHILD
DATA
GENREF
LEVEL
LEVELMUL
PARCHIL
SIBLOW
SIBPAR
SIBSTR

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Rule Objects

4. Make sure the appropriate server, application, and database are selected from the list boxes.

5. Specify the location of the file by clicking either the Server or Client button.

   If you select Server, the rules files to use must reside in the database directory under `\essbase\app\`*`application_name`*`\`*`database_name`*, where *application_name* and *database_name* represent the name of your application and database. Type the name of the rules source in the Object Name text box or select it from the Objects list box. For example, GENREF.

   If you select Client, the rules file may reside in either the application or database directory under `\essbase\client` or on the drives accessible from the client file system. Click File System to select a rules file from a standard Open Client Data Files dialog box.

   Note: The `\essbase\app` and `\essbase\client` are the default directories specified during installation. You may have set these directories differently.

6. Click OK. Return to the Data Load dialog box.

7. Decide if you want to stop the data load or dimension build if an error occurs. This occurs automatically for free-form files, but not for data sources loaded using a rules file.

   Stop if you want to learn immediately that something is wrong with the data source or rules file.

   Don’t stop if you want to load as much data as possible and then look at errors in the error log.

8. To stop the data load if an error occurs, select “Abort on error during data load.”

Building Dimensions Dynamically by Modifying the Outline

You can modify the outline using a data source and rules file. This lets you change or add new dimensions and members to the database based on data in your data source instead of by using the Outline Editor. You must use a rules file to change the outline.

CAUTION: Modifying the outline restructures your database.

To change the outline, select the following options in the Data Load dialog box:

- Modify Outline.

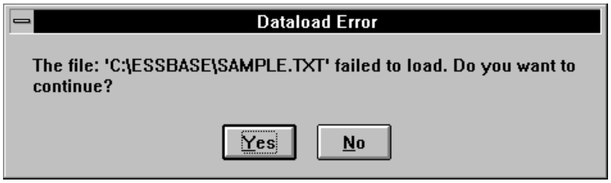
  Essbase updates the outline with any new members or dimensions found in the data source. To set up a dimension build rules file, see Chapter 18, “Introducing Dynamic Dimension Building.”

  Note: If Modify Outline is not selected, Essbase rejects any records containing new members during the data load.

- Interactive.

  Each time a data source fails, Essbase tells you which data source failed and asks you if you want to continue reading the remaining data sources.

  Figure 357: Dataload Error Dialog Box

  Dataload Error

  The file: 'C:\ESSBASE\SAMPLE.TXT' failed to load. Do you want to continue?

  To continue with the remaining data sources, click Yes. To stop, click No. Any data sources used before you stop are in the database.

  Check the error log to determine why Essbase did not load the data source or perform the dimension build operation. See "Finishing the Data Load or Dimension Build" on page 653 if you do not know how to do this.

Updating the Database Outline in Batch Mode

After you create a dimension build rules file, you may want to automate the process of updating dimensions. You can modify the outline, load data, and calculate databases using a batch job. See Chapter 45, "Automating the Production Environment," for more information.

Setting a Load or Build Error Log

You can set a file to record errors during the data load or dimension build if you are using a rules file. A data load or dimension build error file can be a valuable debugging tool if your data load or dimension build fails. By default, when performing a dimension build through Outline Editor or using one of the BUILDDIM ESSCMD commands, the error log is named `DIMBUILD.ERR`. Otherwise, the default name for the error log is `DATALOAD.ERR`. This error log contains messages from both operations when they are combined using the Application Manager Database > Load data menu command. These logs are located in the `\ESSBASE\CLIENT` directory.

CAUTION: If an error output file is not specified, Essbase does not capture data load or dimension build errors.

For more information on errors during data loading or dimension building, see “Finishing the Data Load or Dimension Build” on page 653.

Now you can start loading data or building dimensions.

Starting the Data Load or Dimension Build

After you have set the data load options in the Data Load dialog box, you can start loading the data sources or building dimensions dynamically.

- For data loads, make sure Load Data is selected.
- For dimension builds, make sure Modify Outline is selected.

Click OK.

To speed up or optimize a data load, see Chapter 50, “Optimizing Data Loads.”

Tip: You can also load data into Essbase databases using these methods:

Tool	Instructions	For More Information
MaxL	**import data** to load data; **import dimensions** to build dimensions	*Technical Reference* in the `docs` directory
ESSCMD	LOADDATA or IMPORT to load data; BUILDDIM to build dimensions	

Finishing the Data Load or Dimension Build

When the data load or dimension build finishes, Essbase displays a dialog box listing the results. Data loads and dimension builds end in one of the following:

- Complete load
- Partial load
- No load

Note: If you are loading data, the state of the load is communicated in the Data Load Completed dialog box.

If you are building dimensions, the state of the build is communicated in the Dimension Build Completed dialog box, which, except for the title, is identical to the Data Load Completed dialog box.

If you are performing a data load and dimension build simultaneously, Essbase displays both dialog boxes.

If a data load process is terminated, Essbase displays the file name as partially loaded. For more information about terminating processes, see the following information sources:

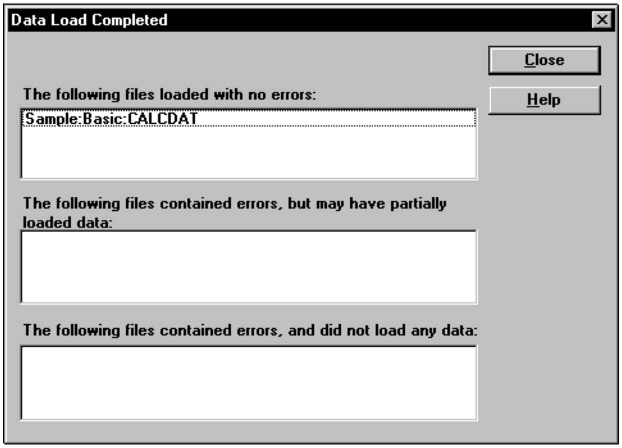
Tool	Instructions	For More Information
Administration Services	Session window	*Essbase Administration Services Online Help*
MaxL	**alter system kill request**	*Technical Reference* in the `docs` directory

Note: If you initiate a data load from a client and terminate the data load process from the server, depending on the size of the file and how much source data that Essbase has processed, it could take some time before the client responds to the termination request. Essbase reads the entire source file until all source data is read. If the process is terminated from the same machine that initiated it, the termination is immediate.

Complete Load

In a complete load, Essbase had no problems loading every specified record in each data source. When data sources load completely, Essbase lists the data sources successfully loaded in the following dialog box. In Figure 358, for example, the `Calcdat` file loaded successfully.

Figure 358: Data Load Completed Dialog Box



The following files loaded with no errors:

Sample:Basic:CALCDAT

The following files contained errors, but may have partially loaded data:

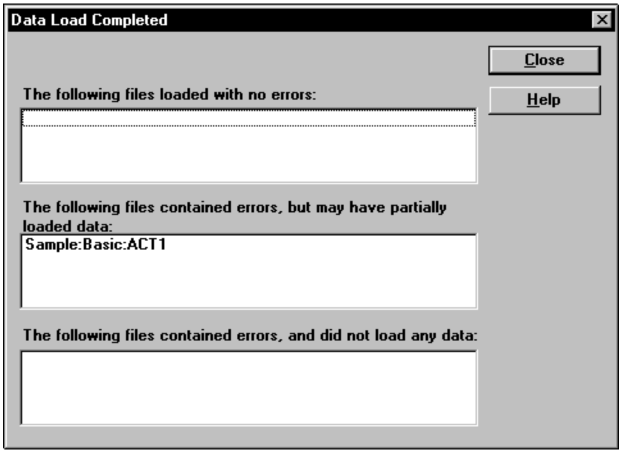
The following files contained errors, and did not load any data:

Partial Load

In a partial load, some of the data sources might have loaded and some might not have loaded. The Data Load Completed dialog box lists all files that may have partially loaded in the middle list box. Essbase lists all free-form data loads that fail here.

In Figure 359, for example, the Act1 text file did not load successfully.

Figure 359: Partial Data Load



The following files loaded with no errors:

The following files contained errors, but may have partially loaded data:

Sample:Basic:ACT1

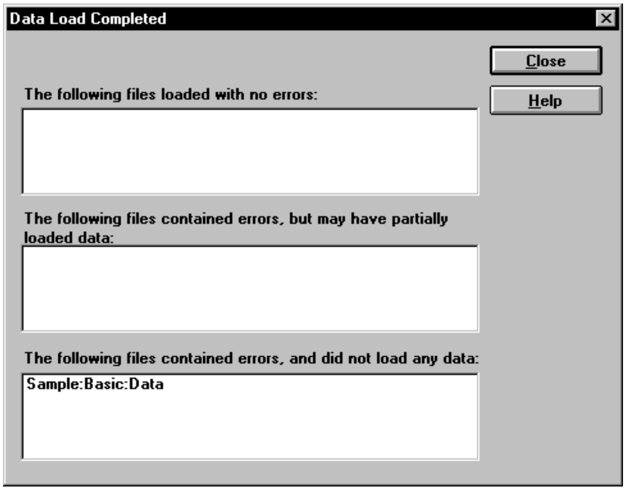
The following files contained errors, and did not load any data:

To fix the data source, see “Debugging a Data Load” on page 658.

No Load

When no data sources or records were loaded into the database, the following dialog box is displayed.

Figure 360: No Data Loaded



Data Load Completed

The following files loaded with no errors:

The following files contained errors, but may have partially loaded data:

The following files contained errors, and did not load any data:

Sample:Basic:Data

To fix the data source, see “Debugging a Data Load” on page 658.

Tips for Loading Data

This section lists tips for data loading. It describes how to load data into a parent instead of its children, how to load a subset of records in a data source, and how to load data using a spreadsheet.

Where to Load Data

If you load data into the parent member, when you calculate your database, the consolidation of the children's values can overwrite the parent's data. To prevent this from happening:

- If possible, do not load data directly into a parent.
- If you must load data into the parent member, make sure Essbase knows not to aggregate `#MISSING` values from children into their parents. Set this at the database level through Essbase Application Manager or use the SETDBSTATEITEM command in ESSCMD. You can also use the SET AGGMISSG command in calculation scripts. See the *Technical Reference* in the `docs` directory for more information.

  This only works if the children's values are empty (`#MISSING`). If the children have data values, those values will still overwrite the values of the parent. See "Aggregating #MISSING Values" on page 1417 for more information.

Loading a Range of Records

You can load a range of records from a data source. For example, you could load just the records 250 to 500 without loading the other records in the data source.

➤ To load a range of records:

1. Number the records in the data source using a text editing tool.
2. Open the data source and rules file in the Data Prep Editor.
3. Ignore the column containing the record number. See "Ignoring Fields" on page 614.
4. Define a rejection criterion to reject all records except those you want to load. For example, reject all records where the ignored column is less than 250 and greater than 500. See "Rejecting Records" on page 603.

   Note: You cannot reject more records than the error log can hold. By default, this is 1000, but you can change it by setting the DATAERRORLIMIT in the `ESSBASE.CFG` file. See the *Technical Reference* in the `docs` directory for more information.

Loading Data Using a Spreadsheet

If you load data using a spreadsheet, see the following documents:

- *Essbase Spreadsheet Add-in User's Guide* for your spreadsheet.
- If you want to use VBA functions, you must also use the Visual Basic API. See ESBIMPORT() in the *API Reference* in your `docs` directory. See the Spreadsheet Add-in Online Help for information about using VBA with the Visual Basic API.

Debugging a Data Load

If you try to load a data source into OLAP Server, but it does not load correctly, check the following:

- Are you connected to the appropriate application and database?
- Did you try to load the correct data source?

If the answer to those questions is yes, then there is probably something wrong. When you have trouble loading a data source, look at the error log generated for that data load. It lists the errors that occurred when Essbase tried to load the data source. The error log is located on the client machine in `\essbase\client\dataload.err`. For more information about the error log, see "Understanding and Using Dimension Build and Data Load Error Logs" on page 1242.

Use these sections to debug a data load:

- "Debugging Without a Data Load Build Error Log" on page 659
- "Finding Errors Written to Other Locations" on page 659
- "Resolving Problems With Data Loaded Incorrectly" on page 660
- "Verifying that the Server Is Available" on page 661
- "Verifying that the Data Source Is Available" on page 661
- "Recovering from a Server Crash" on page 662
- "Resolving Problems With Validating a Rules File" on page 662
- "Creating Rejection Criteria for End of File Markers" on page 663

When you correct the problems in the data load, you can reload the records that didn't load by reloading the error log. For more information, see "Loading Dimension Build or Data Load Error Logs" on page 1245.

Debugging Without a Data Load Build Error Log

If there is no error log, check the following:

- Did the person running the data load set one up? See "Setting a Load or Build Error Log" on page 651 for information on setting up an error log. By default, Essbase creates an error log when you load data using a rules file.
- Are you sure that the data source and server are available? See the "Verifying that the Server Is Available" on page 661 and "Verifying that the Data Source Is Available" on page 661 sections in this chapter for more information.
- Did the server crash during the data load? If so, you probably received a time-out error on the client. If the server crashed, see "Recovering from a Server Crash" on page 662.

If the error log exists but is empty, Essbase does not think that an error occurred during loading. Check the following:

- Does the rules file contain selection/rejection criteria that rejected every record in the data source? See "Selecting Records" on page 602 if you do not know how to set up selection/rejection criteria.
- Is the rules file correct? Does the rules file validate properly? See "Resolving Problems With Validating a Rules File" on page 662.

Finding Errors Written to Other Locations

When Essbase cannot load a record, it writes the record to the error log, `dataload.err,`on the client. There is a limit to the number of records that an error log can contain. The default limit is 1000 records, but you can set the limit to any value between 1 - 65000 by setting DATAERRORLIMIT in the `essbase.cfg` file. See the *Technical Reference* in the `docs` directory for more information.

When Essbase writes the maximum allowed number of records in the error log, it does not log any other errors it encounters. The data load, however, continues. Any subsequent errors are lost.

Resolving Problems With Data Loaded Incorrectly

If the data source loads correctly, but the data in the database is wrong, check the following:

- Are you sure that you loaded the correct data source? If so, check the data source again to make sure it contains the correct values.
- Are there any blank fields in the data source? You must insert `#MI` or `#MISSING` into a data field that has no value. Otherwise, the data source may not load correctly. To replace a blank field with `#MI` or `#MISSING`, see "Replacing an Empty Field with Text" on page 627.
- Is the data source formatted correctly? Are all ranges set up properly?
- Are there any implicitly shared members you were unaware of? Implicit shares happen when a parent and child share the same data value. This occurs if a parent has only one child or only one child rolls up into the parent. See the "Building Shared Members Using a Rules File" on page 517.
- Did you add incoming data to existing data instead of replacing it using a rules file? See "Changing Data Values" on page 634.
- Have you selected or rejected any records that you did not intend to select or reject using a rules file? See Chapter 21, "Setting up a Rules File to Manipulate Records."
- If the sign is reversed (for example, a minus sign instead of a plus sign), did you perform any sign flips on UDAs (user-defined attributes) in your rules file using a rules file? See "Flipping Field Signs" on page 639.
- Did you clear data combinations that you did not intend to clear using a rules file? See "Clearing Existing Data Values" on page 635.
- Did you scale the incoming values incorrectly using a rules file? See "Scaling Data Values" on page 637.
- Are all member and alias names less than 79 characters long?

Note: You can check data by exporting it, running a report on it, or by using a spreadsheet. To do exports and reports, see Chapter 35, "Developing Report Scripts" and Chapter 45, "Automating the Production Environment." To use a spreadsheet, see the *Essbase Spreadsheet Add-in User's Guide* for your particular spreadsheet.

After you fix the problem with the database or the rules file, you can load just the records that failed by loading the error log. See "Loading Dimension Build or Data Load Error Logs" on page 1245.

Verifying that the Server Is Available

Try to access the server without using Essbase to help identify if the problem is with Essbase and not with your server or network. Check the following:

- Is the server machine running? Try to connect to it without using Essbase. If you cannot, check with your system administrator.
- Is OLAP Server running? Check with your Essbase administrator.
- Can the client machine connect to the server machine? Try to connect to the server machine from the client machine without using Essbase.

Verifying that the Data Source Is Available

If Essbase cannot open the data source to load, check the following:

- Is the data source already open? This can happen if a user is editing the data source. Essbase can only load data sources that are not locked by another user or application.
- Does the data source have the correct file extension? All text files must have a file extension of `.TXT`. All rules files must have a file extension of `.RUL`.
- Is the data source name or path name correct? Check for misspellings.
- Is the data source in the specified location? Check to make sure no one has moved or deleted the data source.
- If you are using an SQL data source, is the connection information (such as the user name, password, or database name) correct?
- If you are using an SQL data source, can you connect to the SQL data source without using Essbase?

Recovering from a Server Crash

If the server crashes while you are loading data, Essbase sends you a time-out error. If you are overwriting the values in the database with the data source, reload the data sources after the server is running again.

If the Isolation Level transaction setting is Committed, you must re-start the data load from the beginning. If the Isolation Level is Uncommitted, and you are adding to or subtracting from the existing values in the database when the server crashes, do the following:

1. Determine how much data Essbase loaded before the crash. Compare the values in the data source with the values in the database. If the values you are adding to or subtracting from were not changed, restart the data load.
2. If the values you are adding to or subtracting from were changed, you must clear the values that loaded and reload the previous data sources. If, for example, you derive the monthly sales figures by adding the sales figures for each week as they are loaded, clear the sales figures in the database and re-load the sales figures for each week up to the current week.

For more information on Isolation Level settings, see "Isolation Levels" on page 1132.

Resolving Problems With Validating a Rules File

If you cannot validate your rules file, check to make sure that it is set up correctly:

- All members and dimensions are spelled correctly.
- All members are quoted, if they contain numbers or file delimiters.
- The rules file is associated with the correct outline.
- There are no extra delimiters in the data source.
- Each record contains only one member from each dimension.
- No members from the same dimension appear more than once in a record.
- Any dimensions specified in the header record are not also in the data records.
- You did not define more than one field as a data field.

Creating Rejection Criteria for End of File Markers

Some SQL data sources may have end of file markers made up of special characters that can cause a data load or dimension build to fail. To fix this problem, define a rejection criterion to reject that record.

1. Find the end of file marker in your SQL data source.
2. Determine how to search for it using the Essbase search command. This may be difficult as the end of file marker may be composed of one or more special characters. See “Ignoring Fields Based on String Matching” on page 616 for information on how to do this.
3. Define a rejection criterion that rejects the end of file marker. See “Rejecting Records” on page 603 for information on how to do this.

Part

VI

Calculating Data

This part describes how to calculate the data in OLAP Server databases, including how to create formulas, define the calculation order, calculate data values dynamically, calculate time series data, and create calculation scripts:

- Chapter 24, “Introduction to Database Calculations,” explains the basic concepts behind database calculations.
- Chapter 25, “Developing Formulas,” explains formulas and describes how to create formulas on members using the Formula Editor.
- Chapter 26, “Examples of Formulas,” contains detailed examples of formulas.
- Chapter 27, “Defining the Calculation Order,” describes how to set the calculation order of the members in a database.
- Chapter 28, “Dynamically Calculating Data Values,” describes how to set Essbase to calculate the values for dimensions and members when they are requested by users, instead of in advance.
- Chapter 29, “Calculating Time Series Data,” describes how to calculate time series data including First, Last, Average, and Period-To-Date values for both single server and partitioned applications.
- Chapter 30, “Developing Calculation Scripts,” introduces you to calculation scripts and describes how to create calculation scripts using the Calc Script Editor.
- Chapter 31, “Examples of Calculation Scripts,” contains detailed examples of calculation scripts.

Note: For optimization information, see Chapter 51, “Optimizing Calculations,” and Chapter 52, “Optimizing with Intelligent Calculation.”

Chapter

24

Introduction to Database Calculations

This chapter describes how to calculate a database. It also explains the concept of calculating a multidimensional database.

This chapter includes the following sections:

Understanding Database Calculations

A database contains two types of values. It contains the values that you enter, which are called *input data*, and the values that have been calculated from the input data.

Consider the following examples:

- You enter regional sales figures for a variety of products. You calculate the total sales for each product.
- You enter the budget and actual values for the Cost of Goods Sold for several products in several regions. You calculate the variance between budget and actual values for each product in each region.
- Your database contains regional sales figures and prices for all your products. You calculate what happens to your total profit if you increase the price of one product in one region by 5%.

Small differences in the precision of cell values may occur between calculations run on different platforms, due to operating system math library differences.

Essbase offers two ways that you can calculate a database:

- Outline calculation
- Calc script calculation

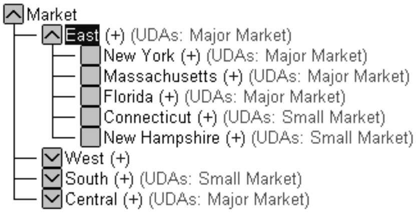
Which way you choose depends on the type of calculation that you want to do.

Outline Calculation

Outline calculation is the simplest method of calculation. Essbase bases the calculation of the database on the relationships between members in the database outline and on any formulas that have been associated with members in the outline.

For example, Figure 361 shows the relationships between the members of the Market dimension in the Sample Basic database. The values for New York, Massachusetts, Florida, Connecticut, and New Hampshire are added to calculate the value for East. The values for East, West, South, and Central are added to calculate the total value for Market.

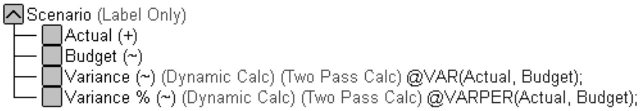
Figure 361: Relationship Between Members of the Market Dimension



```
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)
```

Figure 362 shows the Scenario dimension from the Sample Basic database. The Variance and Variance % members are calculated using the formulas attached to them.

Figure 362: Calculation of Variance and Variance %



```
Scenario (Label Only)
  Actual (+)
  Budget (~)
  Variance (~) (Dynamic Calc) (Two Pass Calc) @VAR(Actual, Budget);
  Variance % (~) (Dynamic Calc) (Two Pass Calc) @VARPER(Actual, Budget);
```

For more information on creating database outlines, see Chapter 7, “Creating and Changing Database Outlines.”

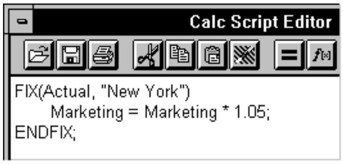
When you design an overall database calculation, it may be more efficient to calculate some member combinations when you retrieve the data, instead of pre-calculating the member combinations during the regular database calculation. You can use dynamic calculations to calculate data at retrieval time. For more information, see Chapter 28, “Dynamically Calculating Data Values.”

Calc Script Calculation

Calc script calculation is the second method of calculation. Using a calculation script, you can choose exactly how to calculate a database. For example, you can calculate part of a database or copy data values between members.

A calculator script contains a series of calculation commands, equations, and formulas. For example, the following calculator script increases the actual marketing expenses in the New York region by 5%.

Figure 363: Calc Script Editor



```
FIX(Actual, "New York")
      Marketing = Marketing * 1.05;
ENDFIX;
```

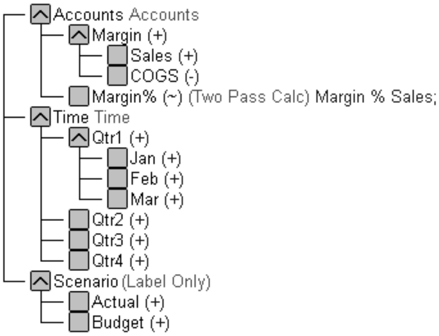
For more information on calculator scripts, see Chapter 30, “Developing Calculation Scripts.”

Calculating a Multidimensional Database

To understand the nature of multidimensional calculations, you need to know some basic multidimensional concepts.

To illustrate these concepts, consider the following, simplified database:

Figure 364: Calculating a Multidimensional Database



The database has three dimensions: Accounts, Time, and Scenario.

The Accounts dimension has four members:

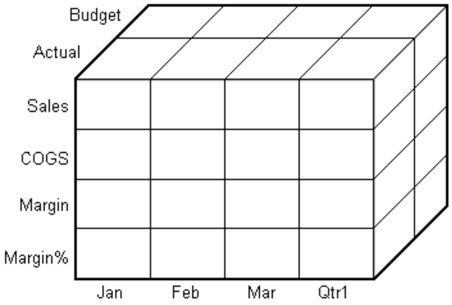
- Sales and COGS are input values.
- Margin = Sales - COGS.
- Margin% = Margin % Sales (Margin as a percentage of Sales).

The Time dimension has four quarters. The example displays only the members in Qtr1: Jan, Feb, and Mar.

The Scenario dimension has two child members: Budget for budget values and Actual for actual values.

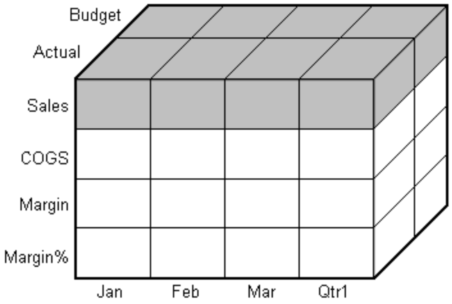
An intersection of members (one member on each dimension) represents a data value. Our example has three dimensions; therefore, the dimensions and data values in the database can be represented as a cube:

Figure 365: Three-Dimensional Database



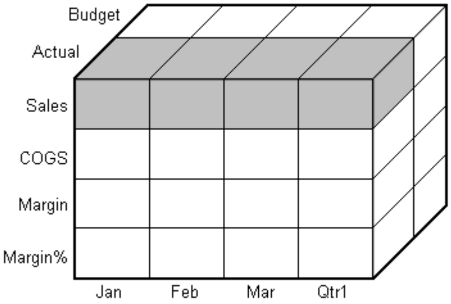
As shown in Figure 366, you can see that when you refer to Sales, you are referring to a slice of the database containing eight Sales values.

Figure 366: Sales, Actual, Budget Slice of the Database



When you refer to Actual Sales, you are referring to four Sales values:

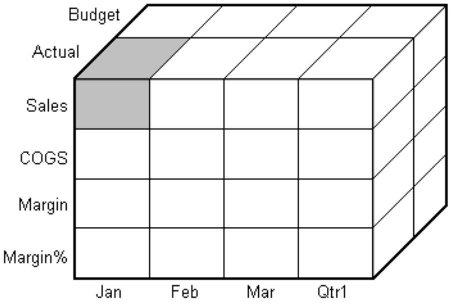
Figure 367: Actual, Sales Slice of the Database



To refer to a specific data value in a multidimensional database, you need to specify its member on each dimension. A data value is stored in a single cell in the database. In Figure 368, the cell containing the data value for Sales, Jan, Actual is shaded.

In Essbase, member combinations are denoted by a cross-dimensional operator. The symbol for the cross-dimensional operator is ->. So Sales, Jan, Actual is written Sales -> Jan -> Actual.

Figure 368: Sales, Jan, Actual Slice of the Database



When Essbase calculates the formula “Margin% = Margin % Sales,” it takes each Margin value and calculates it as a percentage of its corresponding Sales value.

Essbase cycles through the database and calculates Margin% as follows:

1. Margin -> Jan -> Actual as a percentage of Sales -> Jan -> Actual. The result is placed in Margin% -> Jan -> Actual.
2. Margin -> Feb -> Actual as a percentage of Sales -> Feb -> Actual. The result is placed in Margin% -> Feb -> Actual.
3. Margin -> Mar -> Actual as a percentage of Sales -> Mar -> Actual. The result is placed in Margin% -> Mar -> Actual.
4. Margin -> Qtr1 -> Actual as a percentage of Sales -> Qtr1 -> Actual. The result is placed in Margin% -> Qtr1 -> Actual.
5. Margin -> Jan -> Budget as a percentage of Sales -> Jan -> Budget. The result is placed in Margin% -> Jan -> Budget.
6. Essbase continues cycling through the database until it has calculated Margin% for every combination of members in the database.

For more information on database calculation order, see Chapter 27, “Defining the Calculation Order.”

Setting the Default Calculation

By default, the default calculation for a database is a CALC ALL of the database outline. CALC ALL consolidates all dimensions and members and calculates all formulas in the outline.

However, you can specify any calculator script as the default database calculation. Thus, you can assign a frequently-used script to the database rather than loading the script each time you want to perform its calculation. Also, if you want a calculator script to work with settings defined in the Calc Options group of the Database Settings dialog box, you must set the calculator script as the default calculation.

➤ To set the default calculation in Application Manager:

1. Select Database > Set Default.

   Essbase displays the Set Default Calc dialog box.
2. Select Use Calc Script Object.

3. From the list of available calculator scripts, select a calculator script.
4. Click OK.

Tip: You can set the default calculation outside Application Manager.

Tool	Instructions	For More Information
Administration Services	Use Set Default Calculation dialog box	*Essbase Administration Services Online Help*
MaxL	alter database	The *Technical Reference* in the `docs` directory
ESSCMD	SETDEFAULTCALCFILE	The *Technical Reference* in the `docs` directory See also Chapter 45, “Automating the Production Environment”

Calculating a Database

You can calculate a database using any one of these tools:

- Hyperion Enterprise Administration Services Console
- Application Manager
- Spreadsheet Add-in
- MaxL
- ESSCMD
- Administration Services

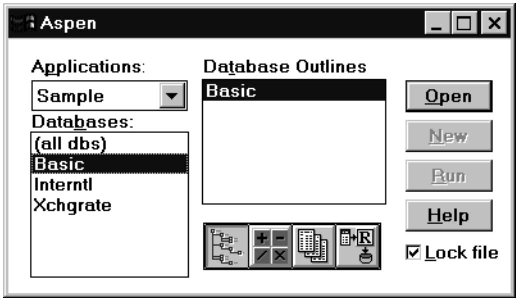
This section includes information on calculating a database using Application Manager, MaxL, and ESSCMD. For information on calculating a database from Spreadsheet Add-in, see the *Essbase Spreadsheet Add-in User’s Guide* or *Essbase Administration Services Online Help*.

Running a Calculation

With Application Manager, you can run the default outline calculation from the desktop.

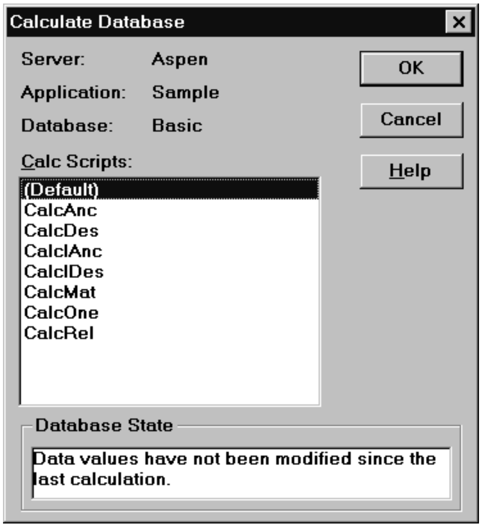
➤ To calculate a database from Application Manager:

1. In the server dialog box (in this case, Aspen), select the appropriate application and database from the Server dialog box:



2. From the Application Manager menu, select Database > Calculate. Essbase displays the Calculate Database dialog box.

Figure 369: Calculate Database Dialog Box



Server: Aspen
Application: Sample
Database: Basic

Data values have not been modified since the last calculation.

In Database State, Essbase reports the current calculation state of the database. See Table 33 on page 676 for a descriptions of each state.

3. Select “(Default)” from the list to run the default outline calculation. Or, select a calculator script to run the calculator script.

   The list includes only calculator scripts to which you have security access. For information on Essbase security privileges, see Chapter 15, “Managing Security for Users and Applications.”

4. Click OK.

   Essbase calculates the database. Essbase displays a message while calculating the database.

Tip: For lengthy calculations, use the Windows Alt + Tab key combination to switch to another Windows application. While Essbase calculates the database, other Application Manager functions are disabled.

Table 33: Calculation States and Descriptions

Calculation State	Description
Calculation in progress.	Essbase is currently calculating the database.
Data values have been modified since the last calculation.	Data values have been changed in the database since the database was last calculated. The last calculation may have been an entire calculation of the database or a calculation of a subset of the database.
Data values have not been modified since the last calculation.	No data values have been changed in the database since the database was last calculated. The last calculation may have been an entire calculation of the database or a calculation of a subset of the database.

Tip: You can perform calculations without Application Manager:

Tool	Instructions	For More Information
Administration Services	Calculate Database dialog box	*Essbase Administration Services Online Help*
MaxL	**execute calculation**	*Technical Reference* in the `docs` directory
ESSCMD	CALC, CALCDEFAULT, and CALCLINE	

Canceling a Calculation

➤ To stop a calculation before Essbase completes it, click the Cancel button in the Calculate Database dialog box.

When you cancel a calculation, Essbase does one of the following:

- Reverts all values to their previous state
- Retains any values calculated before the cancellation

How Essbase handles the cancellation depends on your Essbase Kernel Isolation Level settings. For more information on these settings, see Chapter 40, “Ensuring Data Integrity.”

Understanding Parallel vs. Serial Calculation

Essbase now supports parallel calculation in addition to the default serial calculation. This section explains the difference between parallel and serial calculation and explains how parallel calculation works in Essbase.

Serial calculation, the default, means simply that all the steps in a calculation run on a single thread. Each task is completed before the next is started.

Parallel calculation means that the Essbase calculator can analyze a calculation, and, if appropriate, assign tasks to multiple CPUs (up to 4):

For more information about parallel calculation, including how to determine whether your OLAP Server should use parallel calculation, see “Using Parallel Calculation” on page 1380.

Considering Security

In order to calculate a database, you must have calculate privileges for the database outline.

If you have calculate privileges, you can calculate any value in the database. With calculate privileges, you can calculate a value even if a security filter denies you read and update privileges. Careful consideration should be given to providing users with calculate privileges.

For more information on providing users with calculate privileges and on security filters, see Chapter 15, “Managing Security for Users and Applications.”

Chapter 25

Developing Formulas

This chapter explains how to develop and use formulas to calculate a database. It provides detailed examples of formulas, which you may want to adapt for your own use. For more examples, see Chapter 26, "Examples of Formulas."

This chapter includes the following sections:

Your use of formulas can have significant implications for calculation performance. After reading this section, use the information in Chapter 51, "Optimizing Calculations" to design and create formulas optimized for performance.

For information on using formulas with Hybrid Analysis, see "Using Formulas with Hybrid Analysis" on page 1509 in Appendix D, "Accessing Relational Data with Hybrid Analysis."

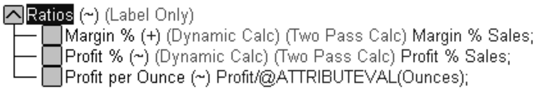
Understanding Formulas

Formulas calculate relationships between members in a database outline. You can use formulas in two ways:

- Apply them to members in the database outline. Use this method if you do not need to control database calculations carefully for accuracy or performance. This method limits formula size to less than 64 Kb if you use Application Manager. For instructions, see "Building Formulas in Formula Editor" on page 691.
- Place them in a calculation script. Use this method if you need to control database calculations carefully. For more information, see Chapter 30, "Developing Calculation Scripts."

The following figure shows the Measures dimension from the Sample Basic database. The Margin %, Profit %, and Profit per Ounce members are calculated using the formulas applied to them.

Figure 370: Calculation of Margin %, Profit %, and Profit per Ounce



```
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);
```

Essbase provides a comprehensive set of operators and functions, which you can use to construct formula calculations on a database. The rest of this section provides a description of the elements you can place in a formula, and provides basic information about formula calculation and syntax:

- "Operators" on page 681
- "Functions" on page 682
- "Dimension and Member Names" on page 684
- "Constant Values" on page 684
- "Non-Constant Values" on page 684

Operators

The following table shows the types of operators you can use in formulas:

Table 34: Descriptions of Operator Types

Operator Type	Description
Mathematical	Perform common arithmetic operations. For example, you can add, subtract, multiply, or divide values. For a complete list of the mathematical operators, see the *Technical Reference* in the `docs` directory.
Conditional	Control the flow of formula executions based on the results of conditional tests. For example, you can use an IF statement to test for a specified condition. For a list of the conditional operators, see the *Technical Reference* in the `docs` directory. For more information on writing conditional formulas, see "Specifying Conditions" on page 707.
Cross-dimensional	Point to the data values of specific member combinations. For example, point to the sales value for a specific product in a specific region. For more information, see "Using the Cross-Dimensional Operator (->)" on page 722.

See "Inserting Text and Operators in a Formula" on page 696 for information on how to add operators to formulas.

For information about using operators with #MISSING, zero, and other values, see the *Technical Reference* in the `docs` directory in the Essbase Functions section.

Functions

Functions are predefined routines that perform specialized calculations and return sets of members or data values. The following table shows the types of functions you can use in formulas:

Table 35: Descriptions of Function Types

Function Type	Description
Boolean	Provide a conditional test by returning either a TRUE (1) or FALSE (0) value. For example, you can use the @ISMBR function to determine whether the current member is one that you specify. For more information, see Chapter 26, “Examples of Formulas.”
Mathematical	Perform specialized mathematical calculations. For example, you can use the @AVG function to return the average value of a list of members. For more information, see Chapter 26, “Examples of Formulas.”
Relationship	Look up data values within a database during a calculation. For example, you can use the @ANCESTVAL function to return the ancestor values of a specified member combination. For more information, see Chapter 26, “Examples of Formulas.”
Range	Declare a range of members as an argument to another function or command. For example, you can use the @SUMRANGE function to return the sum of all members that lie within a specified range. For more information, see Chapter 26, “Examples of Formulas.”
Financial	Perform specialized financial calculations. For example, you can use the @INTEREST function to calculate simple interest or the @PTD function to calculate period-to-date values. For more information, see Chapter 26, “Examples of Formulas.”
Member Set	Generate a list of members that is based on a specified member. For example, you can use the @ICHILDREN function to return a specified member and its children. For more information, see the *Technical Reference* in the `docs` directory.

Table 35: Descriptions of Function Types (Continued)

Function Type	**Description**
Allocation	Allocate values that are input at a parent level across child members. You can allocate values within the same dimension or across multiple dimensions. For example, you can use the @ALLOCATE function to allocate sales values that are input at a parent level to the parent's children; each child's allocation is determined by its share of the previous year's sales.
Forecasting	Manipulate data for the purposes of smoothing or interpolating data, or calculating future values. For example, you can use the @TREND function to calculate future values that are based on curve-fitting to historical values.
Statistical	Calculate advanced statistics. For example, you can use the @RANK function to calculate the rank of a specified member or a specified value in a data set.
Date and Time	Use date and time characteristics in calculation formulas. For example, you can use the @TODATE function to convert date strings to numbers that can be used in calculation formulas.
Miscellaneous	This type provides two different kinds of functionality: • You can specify calculation modes that Essbase is to use to calculate a formula: cell, block, bottom-up, and top-down • You can manipulate character strings for member and dimension names; for example, to generate member names by adding a character prefix to a name or removing a suffix from a name, or by passing the name as a string.
Custom-Defined Functions	This type enables you to perform functions that you develop for your calculation operations. These custom-developed functions are written in the Java programming language and are called by the Essbase calculator framework as external functions.

For a complete list of operators, functions, and syntax, see the *Technical Reference* in the `docs` directory.

Note: Abbreviations of functions are not supported. Some commands may work in an abbreviated form, but if there is another function with a similar name, Essbase may use the wrong function. Use the complete function name to ensure correct results.

Dimension and Member Names

You can include dimension and member names in a formula, for example:

```
Scenario
100-10
Feb
```

Constant Values

You can assign a constant value to a member:

```
California = 120;
```

In this formula, California is a member in a sparse dimension and 120 is a constant value. Essbase automatically creates all possible data blocks for California and assigns the value 120 to all data cells. Many thousands of data blocks may be created. To assign constants in a sparse dimension to only those intersections that require a value, use FIX as described in "Improving Performance for Constants in a Sparse Dimension" on page 1396.

Non-Constant Values

When you assign a constant to a member in a sparse dimension, you do not need to enable Create Blocks on Equations. However, if you assign anything other than a constant to a member in a sparse dimension, and no data block exists for that member, you still need to enable Create Blocks on Equations.

For example, you need to enable Create Blocks on Equations for this formula:

```
West = California + 120;
```

➤ To enable Create Blocks on Equations, use this procedure:

1. In Application Manager, select Database > Settings.

   Essbase displays the Database Settings dialog box.

2. Check Create Blocks on Equations.
3. Click OK.

Tip: You can enable Create Blocks on Equations without Application Manager:

Tool	Instructions	For More Information
Administration Services	Database Properties window > General tab	*Essbase Administration Services Online Help*
MaxL	**alter database**	*Technical Reference* in the `docs` directory
ESSCMD	SETDBSTATE	

Understanding Formula Calculation

For formulas applied to members in a database outline, Essbase calculates formulas when you do the following:

- Run a default (CALC ALL) calculation of a database.
- Run a calculation script that calculates the member containing the formula; for example, a CALC DIM of the dimension containing the member, or the member itself. For more information, see Chapter 30, “Developing Calculation Scripts.”

For a formula in a calculation script, Essbase calculates the formula when it occurs in the calculation script.

If a formula is associated with a dynamically calculated member, Essbase calculates the formula when the user requests the data values. In a calculation script, you cannot calculate a dynamically calculated member or make a dynamically calculated member the target of a formula calculation. For more information, see Chapter 28, “Dynamically Calculating Data Values.”

Using dynamically calculated members in a formula on a database outline or in a calculation script can significantly affect calculation performance. Performance is affected because Essbase has to interrupt the regular calculation to perform the dynamic calculation.

You cannot use substitution variables in formulas that you apply to the database outline. For more information, see "Using Substitution Variables" on page 716.

Understanding Formula Syntax

When you create member formulas, make sure the formulas follow these rules:

- End each statement in the formula with a semicolon (;). For example,

  ```
  Margin % Sales;
  ```

- Enclose a member name in double quotation marks (" ") if the member name meets any of the following conditions:

  - Contains spaces; for example,

    "Opening Inventory" = "Ending Inventory" - Sales + Additions;

  - Is the same as an operator or function name. See the *Technical Reference* in the `docs` directory for a list of operators and functions.

  - Includes any non-alphanumeric character; for example, hyphens (-), asterisks (*), and slashes (/).

  - Is all numeric or starts with one or more numerals; for example, "100" or "10Prod"

  For a complete list of member names that must be enclosed in quotation marks, see "Rules for Naming Dimensions and Members" on page 179.

- End each IF statement in a formula with an ENDIF statement.

  For example, the following formula contains a simple IF... ENDIF statement. You can apply this formula to the Commission member in a database outline:

  ```
  IF(Sales < 100)
     Commission = 0;
  ENDIF;
  ```

If you are using an IF statement nested within another IF statement, end each IF with an ENDIF. For example:

```
"Opening Inventory"
(IF (@ISMBR(Budget))
   IF (@ISMBR(Jan))
   "Opening Inventory" = Jan;
   ELSE
   "Opening Inventory" = @PRIOR("Ending Inventory");
   ENDIF;
ENDIF;)
```

- You do not need to end ELSE or ELSEIF statements with ENDIFs. For example:

```
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;
```

Note: If you use ELSE IF (with a space in between) rather than ELSEIF (one word) in a formula, you must supply an ENDIF for the IF statement.

- Although ending ENDIF statements with a semicolon (;) is not required, it is good practice to follow each ENDIF statement in a formula with a semicolon.

When writing formulas, you can check the syntax using the Formula Editor syntax checker. For more information, see “Writing Formulas” on page 705.

For detailed information on syntax for Essbase functions and commands, see the *Technical Reference* in the `docs` directory.

Example: Creating a Simple Formula

This section provides a step-by-step example of creating and saving a simple formula in an outline using the Formula Editor. For an example of creating a formula in a calculation script, see Chapter 30, "Developing Calculation Scripts." For detailed information on creating formulas, and obtaining the required calculation results, consider all the information in Part VI, "Calculating Data."

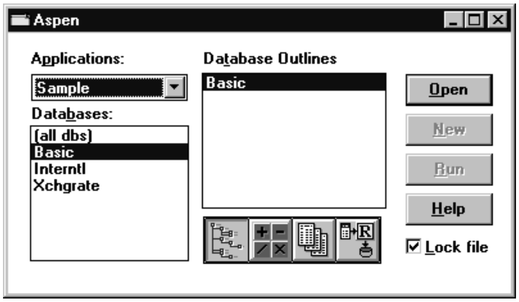
This example is based on the Sample Basic database, which is supplied with the Essbase installation. If you do not have Sample Basic installed, contact your Essbase administrator.

This example shows you how to create a formula on the Variance member of the Scenario dimension. This formula calculates the variance between Budget values and Actual values.

➤ To create the example formula:

1. Start Application Manager, and connect to the OLAP Server.
2. Select the Sample application and the Basic database, and click Open to open the Sample Basic outline.

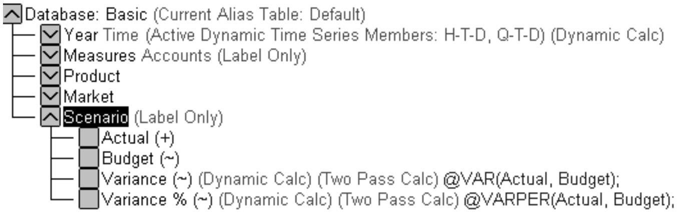
Figure 371: Application Desktop Window



If another user has Sample Basic open and locked, you can clear "Lock file" in the bottom right-hand corner of the application desktop window. However, if you clear "Lock file," you cannot save your work.

3. Double-click the Scenario dimension to display its members.

Figure 372: Scenario Dimension in Sample Basic Outline

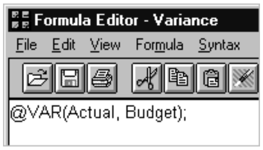


```
Database: Basic (Current Alias Table: Default)
├─ Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
├─ Measures Accounts (Label Only)
├─ Product
├─ Market
└─ Scenario (Label Only)
    ├─ Actual (+)
    ├─ Budget (~)
    ├─ Variance (~) (Dynamic Calc) (Two Pass Calc) @VAR(Actual, Budget);
    └─ Variance % (~) (Dynamic Calc) (Two Pass Calc) @VARPER(Actual, Budget);
```

4. Select the Variance member in the outline, and click the = button.

   Essbase displays the formula in Formula Editor.

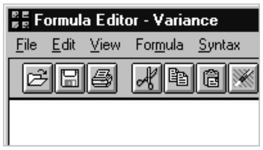
Figure 373: Formula Editor Showing Variance Formula



```
@VAR(Actual, Budget);
```

5. To re-create the formula, select the existing formula and select Edit > Delete in Formula Editor.

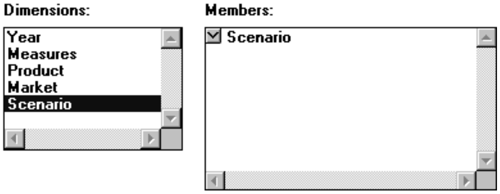
Figure 374: Formula Editor With Variance Formula Deleted



6. In the Dimensions list, select Scenario.

   Essbase displays Scenario in the Members list.

   Figure 375: Formula Editor Dimensions and Members Lists With Scenario Selected



Dimensions:

Year
Measures
Product
Market
Scenario

Members:

☑ Scenario

7. In the Members list, double-click the ☑ button next to Scenario to display the members under Scenario.

8. Select Formula > Paste Function or click the f(x) button.

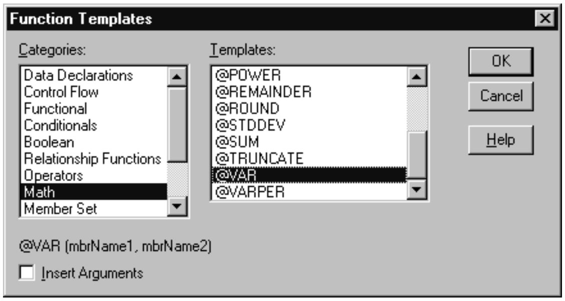
   Essbase displays the Function Templates dialog box.

9. In the Categories list, select Math.

10. In the Templates list, select @VAR.

    Essbase displays the function and the default arguments below the Categories list.

    Figure 376: Function Templates Dialog Box



Function Templates

Categories:

Data Declarations
Control Flow
Functional
Conditionals
Boolean
Relationship Functions
Operators
Math
Member Set

Templates:

@POWER
@REMAINDER
@ROUND
@STDDEV
@SUM
@TRUNCATE
@VAR
@VARPER

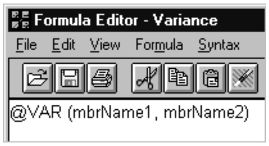
@VAR (mbrName1, mbrName2)

□ Insert Arguments

11. Check Insert Arguments to insert default, temporary arguments in Formula Editor.
12. Click OK.

    Essbase inserts @VAR `(mbrName1`, `mbrName2)` at the cursor position.

Figure 377: Formula Editor With Variance Formula Added



@VAR (mbrName1, mbrName2)

13. Click the ; button, or type a **;** (semicolon) to insert the semicolon formula end-of-line character.
14. Click the 💾 button to save the formula.
15. Close Formula Editor.
16. Click the 💾 button to save the changes to the outline.

You have recreated the formula on Variance in Sample Basic.

Building Formulas in Formula Editor

You use Formula Editor in Application Manager to create formulas. You can type the formulas directly into the formula text area, or you can use the Formula Editor user interface features to create the formula.

Formulas are ASCII text. If required, you can create a formula in the text editor of your choice and paste it into Formula Editor.

This section explains how to perform basic formula creation tasks using the Formula Editor:

- “Opening Formula Editor” on page 692
- “Displaying a Formula” on page 693
- “Adding a Formula” on page 694
- “Changing a Formula” on page 694

- “Saving a Formula” on page 695
- “Printing a Formula” on page 695
- “Deleting a Formula” on page 695
- “Inserting Text and Operators in a Formula” on page 696
- “Inserting Members in a Formula” on page 701
- “Searching for Members” on page 703

Note: You can also check syntax in some situations with the Formula Editor. For instructions, see “Checking Syntax on the Client” on page 732.

Opening Formula Editor

Open Formula Editor to create new formulas or open existing ones.

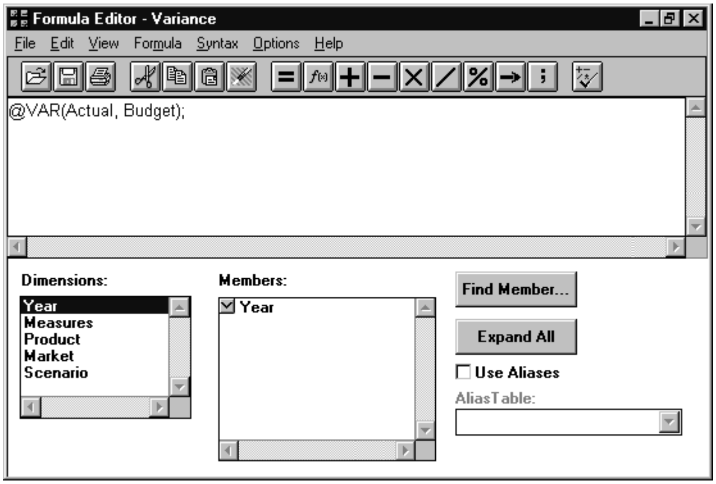
➤ To open Formula Editor from Application Manager:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database.
3. Click the Outline button to display outlines available for the application and database.
4. Double-click the outline to open it in the Outline Editor.
5. In Outline Editor, highlight the member whose formula you want to create or edit.

6. Select Edit > Formula or click the Formula Editor button, ⊟.

   Essbase opens Formula Editor for the selected member. If the member already has a formula, the formula is displayed in Formula Editor. The following figure shows Formula Editor for the Variance member in the Sample Basic database.

Figure 378: Formula Editor Window



```
@VAR(Actual, Budget);
```

Displaying a Formula

Open the database outline to display members and their associated formulas in Outline Editor. You can also highlight the member for which you want to see a formula and click the ⊟ button to open Formula Editor.

You can also use the GETMBRCALC command in ESSCMD to display member formulas. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, “Automating the Production Environment” for information about ESSCMD.

Adding a Formula

You can use Application Manager to add a formula to a database outline.

➤ To add a formula:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database.
3. Click the Outline button to display outlines available for the application and database.
4. Double-click the outline to open it in the Outline Editor.
5. Select the member for which you want to add a formula.
6. Click the = button to open Formula Editor.
7. Type or insert the formula in the Formula Editor window. See “Inserting Text and Operators in a Formula” on page 696.

Changing a Formula

To change an existing formula, open it in Formula Editor.

➤ To change a formula:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database.
3. Click the Outline button to display outlines available for the application and database.
4. Double-click the outline to open it in the Outline Editor.
5. Select the member that has the formula you want to edit.
6. Click the = button to open Formula Editor.
7. Make the required changes to the formula.

Saving a Formula

You can save formulas to the database outline.

➤ To save a formula after you have created or opened it:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database.
3. Click the Outline button to display outlines available for the application and database.
4. Double-click the outline to open it in the Outline Editor.
5. Select a member and click the Formula Editor button to open or create a formula for that member.
6. In Formula Editor, select File > Save or click the 💾 button to save the changes in Formula Editor.
7. Close Formula Editor.
8. Click the 💾 button in Outline Editor to save the changes in the database outline.

   Essbase displays the formula beside the member in the database outline.

Printing a Formula

You can print the contents of a formula from Formula Editor.

➤ To print a formula, in Formula Editor, select File > Print, or click the 🖨 button.

Deleting a Formula

You can delete a formula that has been saved to the database outline.

➤ To delete a formula:

1. In Outline Editor, select the member with the formula that you want to delete.
2. To open Formula Editor, click the = button.

   Essbase displays the formula in the Formula Editor window.

3. Select a member and click the Formula Editor button to open or create a formula for that member.
4. Select the text of the formula.
5. Select Edit > Delete to delete the text of the formula.
6. Click the 💾 button to save the changes in Formula Editor.
7. Close Formula Editor and click the 💾 button to save the changes in the database outline.

   Essbase no longer displays the formula beside the member in the database outline.

➤ To undo the last action, in Formula Editor, select Edit > Undo, or click the ↶ button.

Inserting Text and Operators in a Formula

You can type text and operators directly into the Formula Editor text area, or you can use the toolbar buttons to add the text and operators. You can also copy, cut, and search for text in Formula Editor.

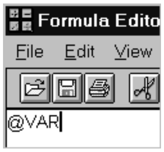
➤ To type text in Formula Editor:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database.
3. Click the Outline button to display outlines available for the application and database.
4. Double-click the outline to open it in the Outline Editor.
5. In Formula Editor, click in the formula text area below the toolbar.

6. Type the appropriate text. You are limited to less than 64 Kb.

   Text is displayed at the cursor position as you type.

   Figure 379: Adding a Formula in Formula Editor

   @VAR

➤ To insert an equal sign (=) in a formula:

1. In Formula Editor, place the cursor where you want to insert the equal sign (=).
2. Type = or click the = button.

➤ To insert a mathematical operator (+, -, X, /, %) in a formula:

1. In Formula Editor, place the cursor where you want to insert the mathematical operator.
2. Type the operator or click one of the following toolbar buttons:

   + − × / %

➤ To insert the cross-dimensional operator (->) in a formula:

1. In Formula Editor, place the cursor where you want to insert the cross-dimensional operator.
2. Type a - (hyphen) followed by a > (greater than symbol), or click the → button.

For more information on the cross-dimensional operator, see “Using the Cross-Dimensional Operator (->)” on page 722.

➤ To insert the semicolon formula end-of-line character (;) in a formula:

1. In Formula Editor, place the cursor at the end of the formula.
2. Type a ; (semicolon) or click the ; button.

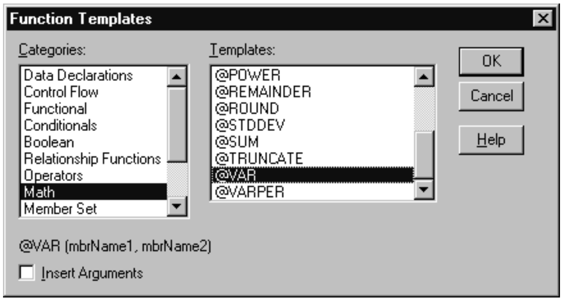
➤ To insert a function or operator in a formula:

1. In Formula Editor, place the cursor where you want to insert the function.
2. Select Formula > Paste Function, or click the f(x) button.

   Essbase displays the Function Templates dialog box.
3. In the Categories list, select a function category. For example, to insert the @VAR function, select Math.
4. In the Templates list, select the required function or operator. For example, scroll down the list and select @VAR.

   Essbase displays the function or operator and the default arguments below the Categories list.

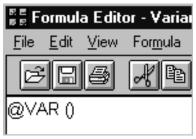
   Figure 380: Function Templates Dialog Box With Math Category Selected



5. If required, select Insert Arguments to insert default, temporary arguments in the function.
6. Click OK.

   Essbase inserts @VAR ()at the cursor position.

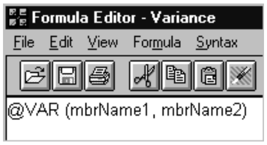
   Figure 381: Formula Editor Showing @VAR Formula

   @VAR ()

   If you checked Insert Arguments, Essbase inserts @VAR and default, temporary arguments. You can then type over the default arguments with the correct arguments.

   Figure 382: Formula Editor Showing @VAR with Arguments

   @VAR (mbrName1, mbrName2)

➤ To cut text in Formula Editor:

Select the text that you want to cut and do one of the following:

- Select Edit > Cut.
- Click the ✂ button.
- Press Ctrl + X.

➤ To copy text in Formula Editor:

Select the text that you want to copy and do one of the following:

- Select Edit > Copy.
- Click the ⎘ button.
- Press Ctrl + C.

➤ To paste text in Formula Editor:

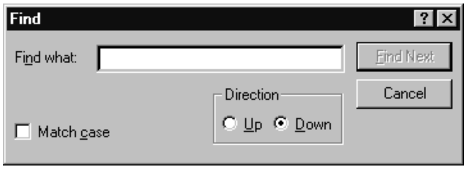
Select the text that you want to paste and do one of the following:

- Select Edit > Paste.
- Click the 📋 button.
- Press Ctrl + V.

➤ To find and replace text in Formula Editor:

1. In Formula Editor, select Edit > Find.
2. Essbase displays the Find dialog box.

   Figure 383: Formula Editor Find Dialog Box



3. In the Find what text box, type the characters that you want to search for.
4. Click the Find Next button.

➤ To do a case-sensitive search in Formula Editor:

1. In the Find dialog box, select Match case.

   For example, to search for Margin but not “margin,” type **Margin** in the Find what text box, and select Match case.

2. Click Find Next.

Inserting Members in a Formula

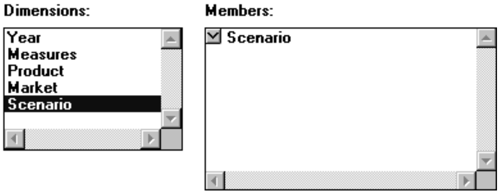
You can insert dimension and member names in Formula Editor instead of typing them.

➤ To insert a dimension name in a formula:

1. In Formula Editor, place the cursor where you want to insert the dimension name.
2. In the Dimensions list, select the dimension that you want to insert in the formula.

   The dimension name displays in the Members list. If a ☑ button displays to the left of the dimension name, then the dimension has children. The following shows the Scenario dimension in the Sample Basic database.

   Figure 384: Formula Editor Dimensions and Members Lists

3. To insert a dimension name in the formula, click the dimension name in the Members list.

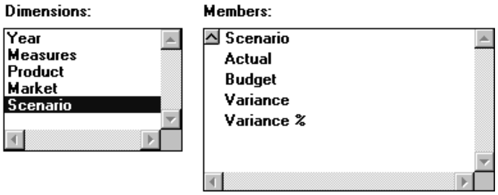
   Essbase inserts the dimension name at the cursor position. To insert a member name (a member name other than the dimension name), expand the member branch and select the member you want to insert.

➤ To expand a member branch to display a member's children:

In the Members list, double-click the ∨ button next to the member name to display the member's children.

The ∨ button changes to a ∧ button.

Figure 385: Formula Editor Dimensions and Members Lists, Expanding the Scenario Member



Dimensions:

Year
Measures
Product
Market
Scenario

Members:

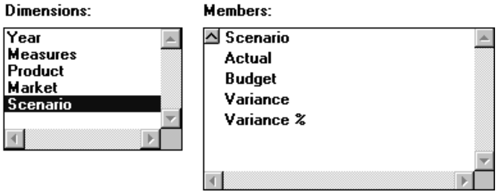
∧ Scenario
Actual
Budget
Variance
Variance %

Double-click the ∧ button to collapse the member branch.

➤ To collapse a member branch:

In the Members list, double-click the ∧ button to collapse the member branch.

Figure 386: Formula Editor Dimensions and Members Lists, Expanded Scenario Dimension



Dimensions:

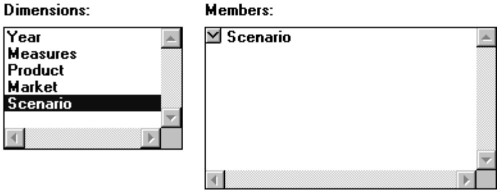
Year
Measures
Product
Market
Scenario

Members:

∧ Scenario
Actual
Budget
Variance
Variance %

The ⌃ button changes to a ⌄ button. Essbase does not display the member’s children:

Figure 387: Formula Editor Dimensions and Members Lists, Collapsing the Scenario Dimension



Searching for Members

➤ To search for a specific member in Formula Editor:

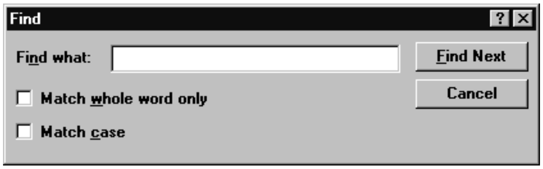
1. In the Dimensions list, select the dimension that you want to search for a member.

   For example, select the Measures dimension from the Sample Basic database.

2. Click Find Member.

   Essbase displays the Find dialog box.

   Figure 388: Formula Editor Find Dialog Box



3. In the Find what text box, enter the characters that you want to search for.

   For example, to search for the Marketing member in the Measures dimension, type **market**.

   - To search for whole words only, select Match whole word only.

     For example, to search for Margin, but not Margin %, type **margin** in the Find what text box, and select Match whole word only.

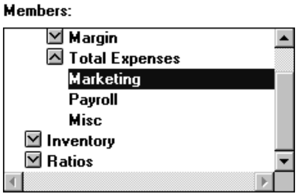
   - To search for case-sensitive characters, select Match case.

     For example to search for Margin, but not margin, type **Margin** in the Find what text box, and select Match case.

4. Click Find Next.

   Essbase finds and selects the Marketing member.

   Figure 389: Formula Editor Members List With Marketing Selected

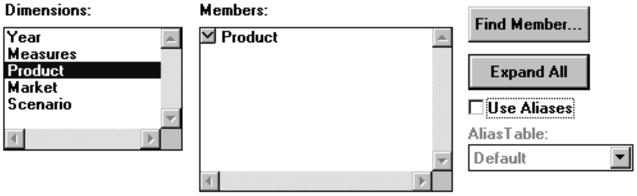


➤ To expand a dimension to display all members in Formula Editor:

1. In the Dimensions list, select the dimension for which you want to display all members.

   For example, select the Product dimension in the Sample Basic database.

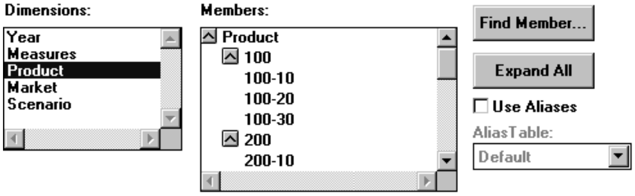
   Figure 390: Formula Editor Dimensions and Members List With Product Selected



2. Click Expand All.

   In the Members list, Essbase displays all members in the dimension.

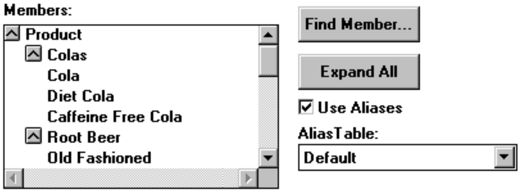
   Figure 391: Formula Editor Dimensions and Members List Showing the Children of Product



➤ To display and insert alias names in Formula Editor, check Use Aliases.

Essbase displays the alias names for the members. The following example shows the Product dimension from the Sample Basic database.

Figure 392: Formula Editor Dimensions and Members List with Alias Names



To select a different alias table, from the Alias Table list box, select a table.

When you select a member from the Members list, Essbase inserts the alias name at the cursor position. If required, Essbase automatically encloses the alias name in double quotation marks (" ").

Writing Formulas

The following sections discuss and give examples of the main types of formulas:

- “Writing Basic Equations” on page 706
- “Specifying Conditions” on page 707
- “Examples of Specifying Conditions” on page 709
- “Value-Related Formulas” on page 710

- “Member-Related Formulas” on page 717
- “Formulas with Other Function Types” on page 725

For more examples of formulas, see Chapter 26, “Examples of Formulas.”

Before writing formulas, review the guidelines in “Understanding Formula Syntax” on page 686.

Writing Basic Equations

You can apply a mathematical operation to a formula to create a basic equation. For example, you can apply the following formula to the Margin member in Sample Basic.

```
Sales - COGS;
```

In a calculation script, you define basic equations as follows:

Member = mathematical operation;

where *Member* is a member name from the database outline and *mathematical operation* is any valid mathematical operation. For example:

```
Margin = Sales - COGS;
```

Whether the example equation is in the database outline or in a calculation script, Essbase cycles through the database subtracting the values in COGS from the values in Sales and placing the results in Margin.

As another example, you can apply the following formula to a Markup member:

```
(Retail - Cost) % Retail;
```

In a calculation script, this would be:

```
Markup = (Retail - Cost) % Retail;
```

In this example, Essbase cycles through the database subtracting the values in Cost from the values in Retail, calculating the resulting values as a percentage of the values in Retail, and placing the result in Markup.

For more information on the nature of multidimensional calculations, see Chapter 24, “Introduction to Database Calculations.”

Specifying Conditions

You can define formulas that use a conditional test or a series of conditional tests to control the flow of calculation.

The IF and ENDIF commands define a *conditional block*. The formulas between the IF and the ENDIF commands are executed only if the test returns TRUE (1). You can use the ELSE and ELSEIF commands to specify alternative actions if the test returns FALSE (0). The formulas following each ELSE command are executed only if the previous test returns FALSE (0). Conditions following each ELSEIF command are tested only if the previous IF command returns FALSE (0).

For more information on the syntax of the IF and ENDIF commands, see "Understanding Formula Syntax" on page 686.

When you use a conditional formula in a calculation script, you must enclose it in parentheses and associate it with a member in the database outline, as shown in the examples in this section.

In conjunction with an IF command, you can use functions that return TRUE or FALSE (1 or 0, respectively) based on the result of a conditional test. These functions are known as *Boolean functions*.

You use Boolean functions to determine which formula to use. The decision is based on the characteristics of the current member combination. For example, you might want to restrict a certain calculation to the members in the Product dimension that contain input data. In this case, you preface the calculation with an IF test based on @ISLEV(Product,0).

If one of the function parameters is a cross-dimensional member, such as @ISMBR(Sales -> Budget), all of the parts of the cross-dimensional member must match the properties of the current cell to return a value of TRUE (1).

You can use the following Boolean functions to specify conditions.

Information You Need To Find	**Use This Function**
The current member has a specified accounts tag (for example, an Expense tag)	@ISACCTYPE
The current member is an ancestor of the specified member	@ISANCEST
The current member is an ancestor of the specified member, or the specified member itself	@ISIANCEST
The current member is a child of the specified member	@ISCHILD

Information You Need To Find	Use This Function
The current member is a child of the specified member, or the specified member itself	@ISICHILD
The current member is a descendant of the specified member	@ISDESC
The current member is a descendant of the specified member, or the specified member itself	@ISIDESC
The current member of the specified dimension is in the generation specified	@ISGEN
The current member of the specified dimension is in the level specified	@ISLEV
The current member matches any of the specified members	@ISMBR
The current member is the parent of the specified member	@ISPARENT
The current member is the parent of the specified member, or the specified member itself	@ISIPARENT
The current member (of the same dimension as the specified member) is in the same generation as the specified member	@ISSAMEGEN
The current member (of the same dimension as the specified member) is in the same level as the specified member	@ISSAMELEV
The current member is a sibling of the specified member	@ISSIBLING
The current member is a sibling of the specified member, or the specified member itself	@ISISIBLING
A specified UDA (user-defined attribute) exists for the current member of the specified dimension	@ISUDA

When you place formulas on the database outline, you can use only the IF, ELSE, ELSEIF, and ENDIF commands and Boolean functions to control the flow of the calculations. You can use additional control commands in a calculation script.

For more information on calculation scripts, see Chapter 30, “Developing Calculation Scripts.” For more information on Essbase functions and calculation commands, see the *Technical Reference* in the `docs` directory.

Examples of Specifying Conditions

You can apply the following formula to a Commission member in the database outline. In the first example, the formula calculates commission at 1% of sales if the sales are greater than 500000:

```
IF(Sales > 500000)
Commission = Sales * .01;
ENDIF;
```

If you place the formula in a calculation script, you need to associate the formula with the Commission member as follows:

```
Commission(IF(Sales > 500000)
Commission = Sales * .01;
ENDIF;)
```

Essbase cycles through the database, performing these calculations:

1. The IF statement checks to see if the value of Sales for the current member combination is greater than 500000.
2. If Sales is greater than 500000, Essbase multiplies the value in Sales by 0.01 and places the result in Commission.

In the next example, the formula tests the ancestry of the current member and then applies the appropriate Payroll calculation formula.

```
IF(@ISIDESC(East) OR @ISIDESC(West))
Payroll = Sales * .15;
ELSEIF(@ISIDESC(Central))
Payroll = Sales * .11;
ELSE
Payroll = Sales * .10;
ENDIF;
```

If you place the formula in a calculation script, you need to associate the formula with the Payroll member as follows:

```
Payroll(IF(@ISIDESC(East) OR @ISIDESC(West))
Payroll = Sales * .15;
ELSEIF(@ISIDESC(Central))
Payroll = Sales * .11;
ELSE
Payroll = Sales * .10;
ENDIF;)
```

Essbase cycles through the database, performing the following calculations:

1. The IF statement uses the @ISIDESC function to check if the current member on the Market dimension is a descendant of either East or West.
2. If the current member on the Market dimension is a descendant of East or West, Essbase multiplies the value in Sales by 0.15 and moves on to the next member combination.
3. If the current member is not a descendant of East or West, the ELSEIF statement uses the @ISIDESC function to check if the current member is a descendant of Central.
4. If the current member on the Market dimension is a descendant of Central, Essbase multiplies the value in Sales by 0.11 and moves on to the next member combination.
5. If the current member is not a descendant of East, West, or Central, Essbase multiplies the value in Sales by 0.10 and moves on to the next member combination.

For more information on the nature of multidimensional calculations, see Chapter 24, “Introduction to Database Calculations.” For more information on the @ISIDESC function, see the *Technical Reference* in the `docs` directory.

Value-Related Formulas

Use this section to find information about formulas related to values:

- “Using Interdependent Values” on page 711
- “Calculating a Variance or Percentage Variance Between Actual and Budget Values” on page 712
- “Allocating Values” on page 714
- “Using the Member Relationships to Look Up Values” on page 715
- “Using Substitution Variables” on page 716

Using Interdependent Values

Essbase optimizes calculation performance by calculating formulas for a range of members in the same dimension at the same time. However, some formulas require values from members of the same dimension, and Essbase may not yet have calculated the required values.

A good example is that of cash flow, in which the opening inventory is dependent on the ending inventory from the previous month.

In Sample Basic, the Opening Inventory and Ending Inventory values need to be calculated on a month-by-month basis.

	Jan	Feb	Mar
Opening Inventory	100	120	110
Sales	50	70	100
Addition	70	60	150
Ending Inventory	120	110	160

Assuming that the Opening Inventory value for January is loaded into the database, the required calculation is:

```
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
```

You can calculate the required results by applying interdependent, multiple equations to a single member in the database outline.

The following formula, applied to the Opening Inventory member in the database outline, calculates the correct values:

```
IF(NOT @ISMBR (Jan))
    "Opening Inventory" = @PRIOR("Ending Inventory");
ENDIF;
"Ending Inventory" = "Opening Inventory" - Sales + Additions;
```

If you place the formula in a calculation script, you need to associate the formula with the Opening Inventory member as follows:

```
"Opening Inventory" (IF(NOT @ISMBR (Jan))
"Opening Inventory" = @PRIOR("Ending Inventory");
ENDIF;
"Ending Inventory" = "Opening Inventory" - Sales + Additions;)
```

Essbase cycles through the months, performing the following calculations:

1. The IF statement and @ISMBR function check that the current member on the Year dimension is not Jan. This step is necessary because the Opening Inventory value for Jan is an input value.
2. If the current month is not Jan, the @PRIOR function obtains the value for the previous month's Ending Inventory. This value is then allocated to the current month's Opening Inventory.
3. The Ending Inventory is calculated for the current month.

Note: To calculate the correct results, it is necessary to place the above formula on a single member, Opening Inventory. If you place the formulas for Opening Inventory and Ending Inventory on their separate members, Essbase calculates Opening Inventory for all months and then Ending Inventory for all months. This means that the value of the previous month's Ending Inventory is not available when Opening Inventory is calculated.

Calculating a Variance or Percentage Variance Between Actual and Budget Values

You can use the @VAR and @VARPER functions to calculate a variance or percentage variance between budget and actual values.

You may want the variance to be positive or negative, depending on whether you are calculating variance for members on the accounts dimension that are:

- Expense items

  You want Essbase to show a positive variance if the actual values are lower than the budget values. For example, you want Essbase to show a positive variance if actual costs are lower than budgeted costs.

- Non-expense items

  You want Essbase to show a negative variance if the actual values are lower than the budget values. For example, you want Essbase to show a negative variance if actual sales are lower than budgeted sales.

By default, Essbase assumes that members are non-expense items and calculates the variance accordingly.

➤ To tell Essbase that a member is an expense item, use this procedure:

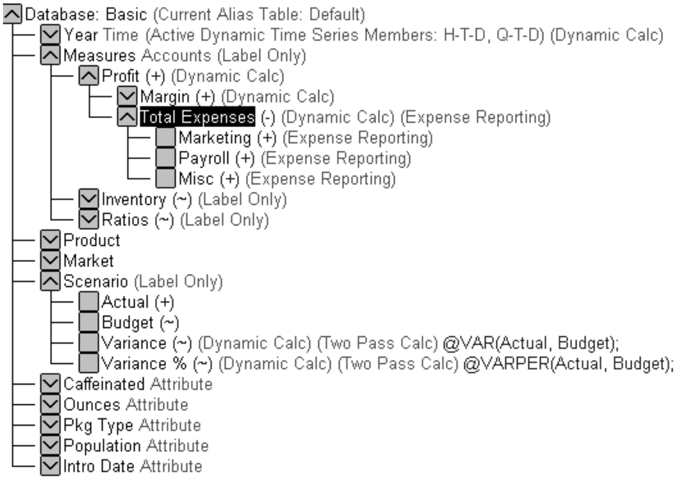
1. In Outline Editor, select the member. The member must be on the dimension tagged as accounts. See Chapter 29, “Calculating Time Series Data.”

2. Click the -$ button.

   Essbase tags the member as an expense item. When you use the @VAR or @VARPER functions, Essbase shows a positive variance if the actual values are lower than the budget values.

   For example, in Sample Basic, the children of Total Expenses are expense items. The Variance and Variance % members of the Scenario dimension calculate the variance between the Actual and Budget values.

Figure 393: Sample Basic Showing Expense Items



```
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
```

Allocating Values

You can allocate values that are input at the parent level across child members in the same dimension or in different dimensions by using the following allocation functions.

Allocated Values	Function To Use
Values from a member, cross-dimensional member, or value across a member list within the same dimension. The allocation is based on a variety of specified criteria.	@ALLOCATE
Values from a member, cross-dimensional member, or value across multiple dimensions. The allocation is based on a variety of specified criteria.	@MDALLOCATE

Note: For examples of calculation scripts using the @ALLOCATE and @MDALLOCATE functions, see "Allocating Values Within or Across Dimensions" on page 883 and the *Technical Reference* in the `docs` directory.

Forecasting Values

You can manipulate data for the purposes of smoothing data, interpolating data, or calculating future values by using the following forecasting functions.

Data Manipulation	Function To Use
To apply a moving average to a data set and replace each term in the list with a trailing average. This function modifies the data set for smoothing purposes.	@MOVAVG
To apply a moving maximum to a data set and replace each term in the list with a trailing maximum. This function modifies the data set for smoothing purposes.	@MOVMAX
To apply a moving median to a data set and replace each term in the list with a trailing median. This function modifies the data set for smoothing purposes.	@MOVMED
To apply a moving minimum to a data set and replace each term in the list with a trailing minimum. This function modifies the data set for smoothing purposes.	@MOVMIN

Data Manipulation	Function To Use
To apply a smoothing spline to a set of data points. A spline is a mathematical curve that is used to smooth or interpolate data.	@SPLINE
To calculate future values and base the calculation on curve-fitting to historical values.	@TREND

For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

Using the Member Relationships to Look Up Values

You can use the member combination that Essbase is currently calculating to look up specific values. These functions are referred to as relationship functions.

Look-up Value	Function To Use
The ancestor values of the specified member combination	@ANCESTVAL
The numeric value of the attribute from the specified numeric or date attribute dimension associated with the current member	@ATTRIBUTEVAL
The text value of the attribute from the specified text attribute dimension associated with the current member	@ATTRIBUTESVAL
The value (TRUE or FALSE) of the attribute from the specified Boolean attribute dimension associated with the current member	@ATTRIBUTEBVAL
The generation number of the current member combination for the specified dimension	@CURGEN
The level number of the current member combination for the specified dimension	@CURLEV
The generation number of the specified member	@GEN
The level number of the specified member	@LEV

Look-up Value	Function To Use
The ancestor values of the specified member combination across multiple dimensions	@MDANCESTVAL
The shared ancestor values of the specified member combination	@SANCESTVAL
The parent values of the specified member combination	@PARENTVAL
The parent values of the specified member combination across multiple dimensions	@MDPARENTVAL
The shared parent values of the specified member combination	@SPARENTVAL
A data value from another database to be used for calculation of a value from the current database	@XREF

For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

Using Substitution Variables

Substitution variables act as placeholders for information that changes regularly; for example, time period information. You can use substitution variables in formulas that you include in a calculation script. You cannot use substitution variables in formulas that you apply to the database outline.

When you run a calculation script, Essbase replaces the substitution variable with the value you have assigned to it. You can create and assign values to substitution variables using Application Manager or ESSCMD.

You can set substitution variables at the server, application, and database levels. Essbase must be able to access the substitution variable from the application and database on which you are running the calculation script.

For more information on creating and assigning values to substitution variables, see Chapter 6, “Creating Applications and Databases.”

➤ To use a substitution variable in a calculation script:

Type an ampersand (**&**) followed by the substitution variable name.

Essbase treats any text string preceded by `&` as a substitution variable.

For example, assume that the substitution variable UpToCurr is defined as Jan:Jun. You can use the following @ISMBR function as part of a conditional test in a calculation script:

```
@ISMBR(&UpToCurr)
```

Before Essbase runs the calculation script, it replaces the substitution variable, as follows:

```
@ISMBR(Jan:Jun)
```

Member-Related Formulas

This section provides information you need to create formulas that refer to members:

- “Specifying a Member List or Range” on page 718
- “Generating Member Lists” on page 719
- “Manipulating Member Names” on page 722
- “Using the Cross-Dimensional Operator (->)” on page 722

Specifying a Member List or Range

In some functions you may need to specify more than one member, or you may need to specify a range of members. For example, the @ISMBR function tests to see if a member that is currently being calculated matches any of a list or range of specified members. You can specify members using the following syntax:

Member List or Range	Syntax
A single member	The member name. For example: Mar2001
A list of members	A comma-delimited (,) list of member names. For example: Mar2001, Apr2001, May2001
A range of all members at the same level, between and including the two defining members	The two defining member names separated by a colon (:). For example: Jan2000:Dec2000
A range of all members in the same generation, between and including the two defining members	The two defining member names separated by two colons (::). For example: Q1_2000::Q4_2000
A function-generated list of members or a range of members	See “Generating Member Lists” on page 719.
A combination of ranges and list	Separate each range, list, and function with a comma (,). For example: Q1_97::Q4_98, FY99, FY2000 or @SIBLINGS(Dept01), Dept65:Dept73, Total_Dept

If you do not specify a list of members or a range of members in a function that requires either, Essbase uses the level 0 members of the dimension tagged as time. If no dimension is tagged as time, Essbase displays an error message.

Generating Member Lists

You can generate member lists that are based on a specified member by using the these member set functions.

Contents of Member List	Function
All ancestors of the specified member, including ancestors of the specified member as a shared member. This function does not include the specified member.	@ALLANCESTORS
All ancestors of the specified member, including ancestors of the specified member as a shared member. This function includes the specified member.	@IALLANCESTORS
The ancestor of the specified member at the specified generation or level.	@ANCEST
All ancestors of the specified member (optionally up to the specified generation or level) but not the specified member.	@ANCESTORS
All ancestors of the specified member (optionally up to the specified generation or level) including the specified member.	@IANCESTORS
All children of the specified member, but not including the specified member.	@CHILDREN
All children of the specified member, including the specified member.	@ICHILDREN
The current member being calculated for the specified dimension.	@CURRMBR
All descendants of the specified member (optionally up to the specified generation or level), but not the specified member nor descendants of shared members.	@DESCENDANTS
All descendants of the specified member (optionally up to the specified generation or level), including the specified member, but not descendants of shared members.	@IDESCENDANTS

Contents of Member List	Function
All descendants of the specified member (optionally up to the specified generation or level), including descendants of shared members, but not the specified member.	@RDESCENDANTS
All descendants of the specified member (optionally up to the specified generation or level), including the specified member and descendants of shared members.	@IRDESCENDANTS
All members of the specified generation in the specified dimension.	@GENMBRS
All members of the specified level in the specified dimension.	@LEVMBRS
All siblings of the specified member, but not the specified member.	@SIBLINGS
All siblings of the specified member, including the specified member.	@ISIBLINGS
All siblings that precede the specified member in the database outline, but not the specified member.	@LSIBLINGS
All siblings that follow the specified member in the database outline, but not the specified member.	@RSIBLINGS
All siblings that precede the specified member in the database outline, including the specified member.	@ILSIBLINGS
All siblings that follow the specified member in the database outline, including the specified member.	@IRSIBLINGS
Separate lists of members to be processed by functions that require multiple list arguments.	@LIST
The member with the name that is provided as a character string.	@MEMBER
A merged list of two member lists to be processed by another function.	@MERGE
A member list that crosses the specified member from one dimension with the specified member range from another dimension.	@RANGE

Contents of Member List	Function
A list of members from which some members have been removed.	@REMOVE
All members that match the specified wildcard selection.	@MATCH
The parent of the current member being calculated in the specified dimension.	@PARENT
All members of the specified generation or level that are above or below the specified member.	@RELATIVE
All members that have a common (UDA) user-defined attribute defined on OLAP Server.	@UDA
All base-dimension members that are associated with the specified attribute-dimension member.	@ATTRIBUTE
All base members that are associated with attributes that satisfy the specified conditions.	@WITHATTR

For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

Manipulating Member Names

You can work with member names as character strings by using the following functions:

Character String Manipulation	Function To Use
To create a character string that is the result of appending a member name or specified character string to another member name or character string	@CONCATENATE
To return a member name as a string	@NAME
To return a substring of characters from another character string or from a member name	@SUBSTRING

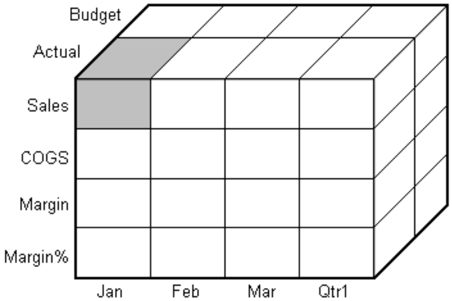
Using the Cross-Dimensional Operator (->)

The cross-dimensional operator points to data values of specific member combinations.

You create the cross-dimensional operator using a hyphen (-) and a greater than symbol (>). Do not leave spaces in between the cross-dimensional operator and the member names.

For example, in this simplified illustration, the shaded data value is Sales -> Jan -> Actual.

Figure 394: Defining a Single Data Value by Using the Cross-Dimensional Operator



The following example illustrates how to use the cross-dimensional operator. This example allocates miscellaneous expenses to each product in each market.

The value of Misc_Expenses for all products in all markets is known. The formula allocates a percentage of the total Misc_Expenses value to each Product -> Market combination. The allocation is based on the value of Sales for each product in each market.

```
Misc_Expenses = Misc_Expenses -> Market -> Product *
(Sales / ( Sales -> Market -> Product));
```

Essbase cycles through the database, performing these calculations:

1. Essbase divides the Sales value for the current member combination by the total Sales value for all markets and all products (Sales -> Market -> Product).
2. It multiplies the value calculated in step 1 by the Misc_Expenses value for all markets and all products (Misc_Expenses -> Market -> Product).
3. It allocates the result to Misc_Expenses for the current member combination.

Consider carefully how you use the cross-dimensional operator, as it can have significant performance implications. For detailed information, see Chapter 51, “Optimizing Calculations.”

In Equations in a Dense Dimension

When you use a cross-dimensional operator in an equation in a dense dimension, if the resultant values are from a dense dimension and the operand or operands are from a sparse dimension, Essbase does not automatically create the required blocks.

Consider an example from Sample Basic, in which you want to create budget sales and expense data from existing actual data. Sales and Expenses are members in the dense Measures dimension. Budget and Actual are members in the sparse Scenario dimension.

```
FIX(Budget)

      (Sales = Sales -> Actual * 1.1;

       Expenses = Expenses -> Actual * .95;)

ENDFIX
```

Note that results of the equation, Sales and Expenses, are dense dimension members, and the operand, Actual, is in a sparse dimension. The calculation script above does *not* create the required data blocks, therefore Budget data values cannot be calculated for blocks that do not already exist.

You can solve this problem using either of these techniques:

- "Using DATACOPY for Equations in a Dense Dimension" on page 724
- "Avoiding Equations in a Dense Dimension" on page 724

Using DATACOPY for Equations in a Dense Dimension

You can solve the problem in "In Equations in a Dense Dimension" on page 723 by preceding the above formulas with a DATACOPY command:

```
DATACOPY Sales -> Actual TO Sales -> Budget;
DATACOPY Expenses -> Actual TO Expenses -> Budget;
FIX(Budget)
      (Sales = Sales -> Actual * 1.1;
       Expenses = Expenses -> Actual *  .95;)
ENDFIX
```

With DATACOPY, Essbase copies the data and creates the required blocks, in this example, blocks that contain the Budget values for each corresponding Actual block that already exists.

Avoiding Equations in a Dense Dimension

You can solve the problem in "In Equations in a Dense Dimension" on page 723 and also avoid copying the data as described in "Using DATACOPY for Equations in a Dense Dimension" on page 724 by ensuring that the results members are from a sparse dimension, not from a dense dimension. In this example, Budget is the results member, and it is from a sparse dimension:

```
FIX(Sales)
      Budget = Actual * 1.1;
ENDFIX
FIX(Expenses)
      Budget = Actual *  .95;
ENDFIX
```

You can also use a member formula that contains the dense member equations:

```
FIX(Sales, Expenses)
Budget (Sales = Sales -> Actual * 1.1;
      Expenses = Expenses -> Actual * .95;)
ENDFIX
```

Formulas with Other Function Types

Use this section to find information about formulas that use other types of formulas:

- “Performing Mathematical Operations” on page 725
- “Statistical Functions” on page 727
- “Using Range Functions” on page 728
- “Calculating Financial Functions” on page 729
- “Using Date and Time Functions” on page 730
- “Using Calculation Mode Functions” on page 731
- “Using Custom-Defined Functions” on page 731

Performing Mathematical Operations

You can perform many mathematical operations in formulas by using the following mathematical functions.

Operation	**Function**
To return the absolute value of an expression	@ABS
To return the average value of the values in the specified member list	@AVG
To return the value of e (the base of natural logarithms) raised to power of the specified expression	@EXP
To return the factorial of an expression	@FACTORIAL
To return the next lowest integer value of a member or expression	@INT
To return the natural logarithm of a specified expression	@LN

Operation	Function
To return the logarithm to a specified base of a specified expression	@LOG
To return the base-10 logarithm of a specified expression	@LOG10
To return the maximum value among the expressions in the specified member list	@MAX
To return the maximum value among the expressions in the specified member list, with the ability to skip zero and `#MISSING` values	@MAXS
To return the minimum value among the expressions in the specified member list	@MIN
To return the minimum value among the expressions in the specified member list, with the ability to skip zero and `#MISSING` values	@MINS
To return the modulus produced by the division of two specified members	@MOD
To return the value of the specified member raised to the specified power	@POWER
To return the remainder value of an expression	@REMAINDER
To return the member or expression rounded to the specified number of decimal places	@ROUND
To return the summation of values of all specified members	@SUM
To return the truncated value of an expression	@TRUNCATE
To return the variance (difference) between two specified members. See "Calculating a Variance or Percentage Variance Between Actual and Budget Values" on page 712.	@VAR
To return the percentage variance (difference) between two specified members. See "Calculating a Variance or Percentage Variance Between Actual and Budget Values" on page 712.	@VARPER

For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

Statistical Functions

You can use these statistical functions to calculate advanced statistics in Essbase.

Calculated Value	Function To Use
The correlation coefficient between two parallel data sets	@CORRELATION
The number of values in the specified data set	@COUNT
The median, or middle number, in the specified data set	@MEDIAN
The mode, or the most frequently occurring value, in the specified data set	@MODE
The rank of the specified member or value in the specified data set	@RANK
The standard deviation, based upon a sample, of the specified members	@STDEV
The standard deviation, based upon the entire population, of the specified members	@STDEVP
The standard deviation, crossed with a range of members, of the specified members	@STDEVRANGE
The variance, based upon a sample, of the specified data set	@VARIANCE
The variance, based upon the entire population, of the specified data set	@VARIANCEP

For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

Using Range Functions

You can execute a function for a range of members by using these range functions.

Calculation	Function To Use
The average value of a member across a range of members	@AVGRANGE
A range of members that is based on the relative position of the member combination Essbase is currently calculating.	@CURRMBRRANGE
The maximum value of a member across a range of members	@MAXRANGE
The maximum value of a member across a range of members, with the ability to skip zero and `#MISSING` values	@MAXSRANGE
The next or *n*th member in a range of members, retaining all other members identical to the current member across multiple dimensions	@MDSHIFT
The minimum value of a member across a range of members	@MINRANGE
The minimum value of a member across a range of members, with the ability to skip zero and `#MISSING` values	@MINSRANGE
The next or *n*th member in a range of members.	@NEXT
The next or *n*th member in a range of members, with the option to skip `#MISSING,` zero, or both values.	@NEXTS
The previous or *n*th previous member in a range of members	@PRIOR
The previous or *n*th previous member in a range of members, with the option to skip `#MISSING,` zero, or both values.	@PRIORS

Calculation	Function To Use
The next or *n*th member in a range of members, retaining all other members identical to the current member and in the specified dimension	@SHIFT. In some cases, @SHIFTPLUS or @SHIFTMINUS.
The summation of values of all specified members across a range of members	@SUMRANGE

For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

Calculating Financial Functions

You can include financial calculations in formulas by using these financial functions.

Calculation	Function To Use
An accumulation of values up to the specified member	@ACCUM
The proceeds of a compound interest calculation	@COMPOUND
A series of values that represent the compound growth of the specified member across a range of members	@COMPOUNDGROWTH
Depreciation for a specific period, calculated using the declining balance method.	@DECLINE
A value discounted by the specified rate, from the first period of the range to the period in which the amount to discount is found	@DISCOUNT
A series of values that represents the linear growth of the specified value	@GROWTH
The simple interest for a specified member at a specified rate	@INTEREST
The internal rate of return on a cash flow	@IRR

Calculation	Function To Use
The Net Present Value of an investment (based on a series of payments and incomes)	@NPV
The period-to-date values of members in the dimension tagged as time	@PTD
The amount per period that an asset in the current period may be depreciated (calculated across a range of periods). The depreciation method used is straight-line depreciation.	@SLN
The amount per period that an asset in the current period may be depreciated (calculated across a range of periods). The depreciation method used is sum of the year's digits.	@SYD

For more information on Essbase functions, see the *Technical Reference* in the `docs` directory.

Using Date and Time Functions

You can use dates with other functions by using this date function.

Date Conversion	Function To Use
Convert date strings to numbers that can be used in calculation formulas	@TODATE

Using Calculation Mode Functions

You can specify which calculation mode that Essbase uses to calculate a formula by using @CALCMODE.

Specification	Function To Use
To specify that Essbase uses cell, block, bottom-up, and top-down calculation modes to calculate a formula.	@CALCMODE

Note: You can also use the configuration setting CALCMODE to set calculation modes to BLOCK or BOTTOMUP at the database, application, or server level. For details, see the *Technical Reference* in the `docs` directory, under "`essbase.cfg` Settings" for CALCMODE or "Essbase Functions" for @CALCMODE.

Using Custom-Defined Functions

Custom-defined functions are calculation functions that you create to perform calculations not otherwise supported by the Essbase calculation scripting language. You can use custom-defined functions in formulas and calculation scripts. These custom-developed functions are written in the Java programming language and registered on the server. The Essbase calculator framework calls them as external functions.

If you are connected to the server, the Custom-Defined Functions category appears in the Function Templates dialog box where you can choose function names to be inserted into the formula.

For more information about Custom-Defined Functions, see Chapter 33, "Developing Custom-Defined Calculation Functions."

Checking Syntax

Essbase includes both client-based and server-based formula syntax checking that tells you about syntax errors in formulas. For example, Essbase tells you if you have mistyped a function name. If you are connected to a server, unknown names can be validated against a list of custom-defined macro and function names. If you are not connected to a server or the application associated with the outline, Essbase may connect you to validate unknown names.

A syntax checker cannot tell you about semantic errors in a formula. Semantic errors occur when a formula does not work as you expect. To find semantic errors, run the calculation and check the results to ensure that they are as you expect.

Because server-based formula validation has access to more information about the database and outline, this form of validation can take more time to complete. For quicker syntax checking, you can use client-based formula validation to find syntax-related errors. To avoid saving outlines that contain formulas with errors, perform a server-based formula validation and correct all errors before the outline goes into production.

Checking Syntax on the Client

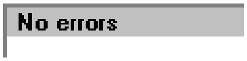
Use the client-based syntax checker to validate formulas and calculation scripts. You can use this feature whether or not you are connected to the server. The client-based syntax checker identifies each error within a formula.

➤ To use the client-based syntax checker to validate a formula in Formula Editor:

Select Syntax > Check Syntax, or click the ✓ button.

Essbase displays the syntax checker results at the bottom of the Formula Editor window. If Essbase finds no syntax errors, it displays the message shown in Figure 395.

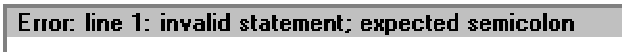
Figure 395: Formula Editor Syntax Checker, No Errors Message



No errors

If Essbase finds one or more syntax errors, it displays the number of the line that includes the error and a brief description of the error. For example, if you do not include a semicolon end-of-line character at the end of a formula, Essbase displays a message similar to the message shown in Figure 396.

Figure 396: Formula Editor Syntax Checker, Syntax Error Message



Error: line 1: invalid statement; expected semicolon

➤ To step through errors in Formula Editor:

Select Syntax > Next Error or Syntax > Previous Error.

When you reach the first or last error, Essbase displays the message shown in Figure 397.

Figure 397: Formula Editor Syntax Checker, No More Errors Message

No more errors

Essbase retains the list of error messages in Formula Editor until you check the syntax again.

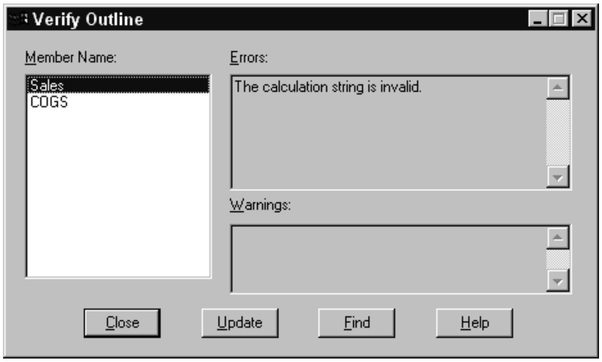
Checking Syntax on the Server

You can check the syntax on the server in two ways:

- By making sure that the toggle Options > Server Formula Validation is set ON and then saving the outline to the server. (A check mark next to the option name indicates server formula validation is set ON.)
- By selecting the Formula Editor menu option Syntax > Validate Formula on Server. If the application is not started, Essbase starts the application.

Essbase displays outline errors and warnings in the Verify Outline dialog box, similar to Figure 398.

Figure 398: Example Verify Outline Dialog Box Containing Formula Errors



Member Name:

Sales
COGS

Errors:

The calculation string is invalid.

Select a member name to see associated errors and warnings. For more information about the error or warning, if the error or warning is in the formula attached to the member, click Find to go to that member in the outline. Open Formula Editor, and use the server-based syntax checker to validate the formula.

If Essbase finds no syntax errors, it displays the No Errors message at the bottom of the Formula Editor window, as shown in Figure 399.

Figure 399: Formula Editor Syntax Checker, No Errors Message

No errors

If a formula passes validation in Formula Editor or Outline Editor, but the server detects semantic errors when the outline is saved:

- The incorrect formula is saved as part of the outline, even though it contains errors.
- The server writes a message in the application log file that indicates what the error is and displays the incorrect formula.
- The server writes an error message to the comment field of the member associated with the incorrect formula. The message indicates that the incorrect formula was not loaded. You can view this comment in Outline Editor by closing and reopening the outline.
- If you do not correct the member formula, and a calculation that includes that member is run, the formula is ignored during the calculation.

After you have corrected the formula and saved the outline, the message in the member comment is deleted. You can view the updated comment when you reopen the outline.

Estimating Disk Size for a Calculation

You can estimate the disk size that would be required for a single CALC ALL given either a full data load or a partial data load. For details, see the *Technical Reference* in the `docs` directory of your Essbase installation, in the ESSCMD section's entry for ESTIMATEFULLDBSIZE and "Estimating Calculations" on page 1379.

Working with Formulas in Partitions

An Essbase partition can span multiple servers, processors, or computers. For more information on partitioning, see Chapter 13, "Designing Partitioned Applications," and Chapter 14, "Building and Maintaining Partitions."

You can use formulas in partitioning, just as you use formulas on your local database. However, if a formula you use in one database references a value from another database, Essbase has to retrieve the data from the other database when calculating the formula. In this case, you need to ensure that the referenced values are up-to-date and to consider carefully the performance impact on the overall database calculation. For more information, see the information on writing calculation scripts for partitions in "Writing Calculation Scripts for Partitions" on page 872.

With transparent partitions, you need to consider carefully how you use formulas on the data target. For more information, see "Transparent Partitions and Member Formulas" on page 332 and "Performance Considerations for Transparent Partitions" on page 330.

Chapter 26

Examples of Formulas

This chapter provides detailed examples of formulas, which you may want to adapt for your own use. For examples of using formulas in calculation scripts, see Chapter 31, “Examples of Calculation Scripts.”

This chapter includes the following sections:

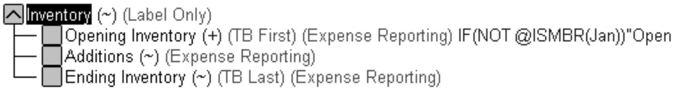
- “Calculating Period-to-Date Values” on page 737
- “Calculating Rolling Values” on page 739
- “Calculating Monthly Asset Movements” on page 740
- “Testing for #MISSING Values” on page 741
- “Calculating an Attribute Formula” on page 742

Calculating Period-to-Date Values

If the outline includes a dimension tagged as accounts, you can use the @PTD function to calculate period-to-date values. You can also use Dynamic Time Series members to calculate period-to-date values. For detailed information, see Chapter 29, “Calculating Time Series Data.”

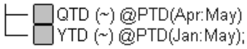
For example, the following figure shows the Inventory branch of the Measures dimension from the Sample Basic database.

Figure 400: Inventory Branch from Sample Basic Outline



```
Inventory (~) (Label Only)
├── Opening Inventory (+) (TB First) (Expense Reporting) IF(NOT @ISMBR(Jan))"Open
├── Additions (~) (Expense Reporting)
└── Ending Inventory (~) (TB Last) (Expense Reporting)
```

To calculate period-to-date values for the year and for the current quarter, add two members to the Year dimension, QTD for quarter-to-date and YTD for year-to-date:



```
QTD (~) @PTD(Apr:May)
YTD (~) @PTD(Jan:May);
```

For example, assuming that the current month is May, you would add this formula to the QTD member:

```
@PTD(Apr:May);
```

And you would add this formula on the YTD member is:

```
@PTD(Jan:May);
```

Essbase sums the values for the range of months as appropriate. However, Opening Inventory has a time balance tag, First, and Ending Inventory has a time balance tag, Last. Essbase takes these values and treats them accordingly. For more information on time balance tags and other accounts tags, see Chapter 29, "Calculating Time Series Data."

The following table provides an example of the calculation results for the members in the Inventory branch and for the Sales member:

Measures -> Time	Jan	Feb	Mar	Apr	May	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**

The values for Sales and Additions have been summed.

Opening Inventory has a First tag. For QTD, Essbase takes the first value in the current quarter, which is Apr. For YTD, Essbase takes the first value in the year, which is Jan.

Ending Inventory has a Last tag. For QTD, Essbase takes the last value in the current quarter, which is May. For YTD, Essbase takes the last value in the year, which is also May.

Calculating Rolling Values

You can use the @AVGRANGE function to calculate rolling averages and the @ACCUM function to calculate rolling year-to-date values.

For example, assume that a database contains monthly Sales data values and that the database outline includes the members AVG_Sales and YTD_Sales.

You would add this formula to the AVG_Sales member:

```
@AVGRANGE(SKIPNONE, Sales, @CURRMBRRANGE(Year, LEV, 0, , 0));
```

And you would add this formula on the YTD_Sales member:

```
@ACCUM(Sales);
```

Essbase calculates the average Sales values across the months in the dimension tagged as time. The SKIPNONE parameter means that all values are included, even #MISSING values. Essbase places the results in AVG_Sales. For more information on #MISSING values, see Chapter 51, “Optimizing Calculations.”

This table shows the results when Essbase calculates the cumulative Sales values and places the results in YTD_Sales:

Measures -> Time	Jan	Feb	Mar	Qtr1
Sales	100	200	300	**600**
AVG_Sales	100	150	200	#MISSING
YTD_Sales	100	300	600	#MISSING

The values for AVG_Sales are averages of the months-to-date. For example, AVG_Sales -> Mar is an average of Sales for Jan, Feb, and Mar.

The values for YTD_Sales are the cumulative values up to the current month. So YTD_Sales -> Feb is the sum of Sales -> Jan and Sales -> Feb.

Calculating Monthly Asset Movements

You can use the @PRIOR function to calculate values based on a previous month’s value.

For example, assume that a database contains assets data values that are stored on a month-by-month basis. You can calculate the difference between the assets values of successive months (the asset movement) by subtracting the previous month’s value from the present month’s value.

Assume these three members manage the asset values for the database:

- Assets for the monthly asset values
- Asset_MVNT for the asset movement values
- Opening_Balance for the asset value at the beginning of the year

For Jan, the Asset_MVNT value is calculated by subtracting the Opening_Balance value from the Jan value.

You would add this formula on the Asset_MVNT member:

```
IF(@ISMBR(Jan)) Asset_MVNT = Assets - Opening_Balance;
ELSE Asset_MVNT = Assets - @PRIOR(Assets);
ENDIF;
```

This table shows the results when Essbase calculates the difference between the values of assets in successive months:

Assets -> Time	**Opening_Balance**	**Jan**	**Feb**	**Mar**
Assets	1200	1400	1300	1800
Asset_MVNT		**200**	**-100**	**500**

Essbase cycles through the months, performing these calculations:

1. The IF statement and @ISMBR function check to see if the current member on the Year dimension is Jan. This check is necessary because the Asset_MVNT value for Jan cannot be calculated by subtracting the previous month's value.
2. If the current member on the Year dimension is Jan, Essbase subtracts the Opening_Balance from the Jan -> Assets value and places the result in Jan -> Asset_MVNT.
3. If the current member on the Year dimension is not Jan, the @PRIOR function obtains the value for the previous month's assets. Essbase subtracts the previous month's assets from the current month's assets. It places the result in the current month's Asset_MVNT value.

Testing for #MISSING Values

You can test for #MISSING values in a database. For more information on #MISSING values, see Chapter 51, "Optimizing Calculations."

Assume that a database outline contains a member called Commission. Commission is paid at 10% of sales when the Sales value for the current member combination is not #MISSING. When applied to a Commission member in the database outline, the following formula calculates Commission:

```
IF(Sales <> #MISSING) Commission = Sales * .9;
ELSE Commission = #MISSING;
ENDIF;
```

If you place the formula in a calculation script, you need to associate it with the commission member as follows:

```
Commission(IF(Sales <> #MISSING) Commission = Sales * .1;
ELSE Commission = #MISSING;
ENDIF;);
```

Essbase cycles through the database, performing the following calculations:

1. The IF statement checks to see if the value of the Sales member for the current member combination is not `#MISSING`.
2. If Sales is not `#MISSING`, Essbase multiplies the value in the Sales member by 0.1 and places the result in the Commission member.
3. If Sales is `#MISSING`, Essbase places `#MISSING` in the Commission member.

Calculating an Attribute Formula

You can perform specific calculations on attribute-dimension members in a database.

Note: For more information about attribute calculations, see “Calculating Attribute Data” on page 253.

For example, to calculate profitability by ounce for products sized in ounces, you can use the @ATTRIBUTEVAL function in a calculation formula. In the Sample Basic database, the Ratios branch of the Measures dimension contains a member called Profit per Ounce. The formula on this member is:

```
Profit/@ATTRIBUTEVAL(Ounces);
```

Essbase cycles through the Products dimension, performing the following calculations:

1. For each base member that is associated with a member from the Ounces attribute dimension, the @ATTRIBUTEVAL function returns the numeric attribute value (for example, 12 for the member 12 under Ounces).
2. Essbase then divides Profit by the result of @ATTRIBUTEVAL to yield Profit per Ounce.

Note: For more information on using attributes in calculation formulas, see “Using Attributes in Calculation Formulas” on page 259. For more information about the @ATTRIBUTEVAL function, see the *Technical Reference* in the `docs` directory.

Chapter

27

Defining the Calculation Order

This chapter describes the order in which Essbase calculates a database. If you use dynamic calculations, see Chapter 28, “Dynamically Calculating Data Values,” for information on the calculation order for the dynamically calculated values.

You should understand the concepts of data blocks and of sparse and dense dimensions before using this information. You should also understand the use of levels and generations. For more information, see Part II, “Designing and Creating Applications and Databases.”

This chapter includes these sections:

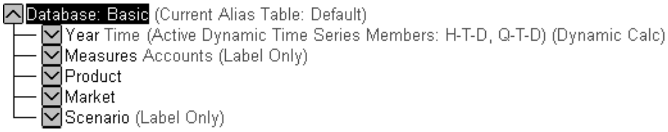
Data Storage in Data Blocks

Essbase stores data values in data blocks. Essbase creates a data block for each unique combination of sparse dimension members (providing that at least one data value exists for the combination).

Each data block contains all the dense dimension member values for its unique combination of sparse dimension members.

In the Sample Basic database, the Year, Measures, and Scenario dimensions are dense. The Product and Market dimensions are sparse.

Figure 401: Dimensions from the Sample Basic Database

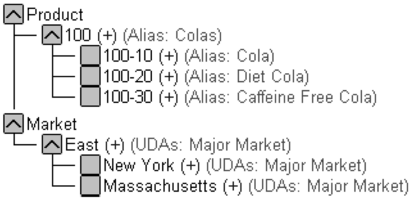


```
Database: Basic (Current Alias Table: Default)
├── Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
├── Measures Accounts (Label Only)
├── Product
├── Market
└── Scenario (Label Only)
```

Note: Sample Basic also contains five attribute dimensions. These dimensions are sparse, Dynamic Calc, meaning that attribute data is not stored in the database. For more information on attributes, see Chapter 9, "Working with Attributes."

Essbase creates a data block for each unique combination of members in the Product and Market dimensions (providing that at least one data value exists for the combination). For example, it creates one data block for the combination of 100-10, New York. This data block contains all the Year, Measures, and Scenario values for 100-10, New York.

Figure 402: Product and Market Dimensions from the Sample Basic Database



```
Product
└── 100 (+) (Alias: Colas)
    ├── 100-10 (+) (Alias: Cola)
    ├── 100-20 (+) (Alias: Diet Cola)
    └── 100-30 (+) (Alias: Caffeine Free Cola)
Market
└── East (+) (UDAs: Major Market)
    ├── New York (+) (UDAs: Major Market)
    └── Massachusetts (+) (UDAs: Major Market)
```

In Essbase, member combinations are denoted by the cross-dimensional operator. The symbol for the cross-dimensional operator is -> (a hyphen followed by a greater than symbol). So 100-10, New York is written 100-10 -> New York.

You can categorize data blocks as follows:

- Input—These blocks are created by loading data to cells in a block. Input blocks can be created for (1) sparse, level 0 member combinations or (2) sparse, upper level member combinations, when at least one of the sparse members is a parent level member. Input blocks can be level 0 or upper level blocks.
- Noninput—These blocks are created through calculations. For example, in Sample Basic, the East -> Cola block is created during a sparse calculation process (that is, the block did not exist before calculation).

- Level 0—These blocks are created for sparse member combinations when all of the sparse members are level 0 members. For example, in Sample Basic, New York -> Cola is a level 0 block because New York and Cola are level 0 members of their respective sparse dimensions. Level 0 blocks can be input or noninput blocks; for example, a level 0 noninput block is created during an allocation process, where data is loaded at a parent level and then allocated down to level 0.
- Upper level—These blocks are created for sparse member combinations when at least one of the sparse members is a parent level member. Upper level blocks can be input or noninput blocks.

For more information on levels and generations, and how Essbase stores data in data blocks, see Chapter 38, “Managing Essbase Kernel Settings.”

Member Calculation Order

Essbase calculates a database at the data block level, bringing one or more blocks into memory and calculating the required values within the block. Essbase calculates the blocks in order, according to their block numbers. The database outline tells Essbase how to order the blocks. Within each block, Essbase calculates the values in order according to the hierarchy in the database outline. Therefore, overall, Essbase calculates a database based on the database outline.

When you perform a default calculation (CALC ALL) on a database, Essbase calculates the dimensions in this order:

If both a dimension tagged as accounts and a dimension tagged as time exist, and if formulas are applied to members on the accounts dimension, Essbase calculates as follows:

1. The dimension tagged as accounts
2. The dimension tagged as time
3. Other dense dimensions (in the order they are displayed in the database outline)
4. Other sparse dimensions (in the order they are displayed in the database outline)

Otherwise, Essbase calculates in this order:

1. Dense dimensions (in the order they display in the database outline)
2. Sparse dimensions (in the order they display in the database outline)

Note: Attribute dimensions, which are not included in the database consolidation, do not affect calculation order. For more information on attribute dimensions, see Chapter 9, “Working with Attributes.”

In the Sample Basic database, the dimensions are calculated in this order: Measures, Year, Scenario, Product, and Market.

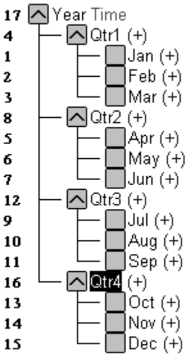
You can override the default order by using a calculation script. For more information on developing calculation scripts, see Chapter 30, “Developing Calculation Scripts.” For more information on accounts and time dimensions, see Chapter 29, “Calculating Time Series Data.”

Member Relationships

The order of calculation within each dimension depends on the relationships between members in the database outline. Within each branch of a dimension, level 0 values are calculated first followed by their level 1, parent value. Then the level 0 values of the next branch are calculated followed by their level 1, parent value. The calculation continues in this way until all levels are calculated.

Figure 403 shows the Year dimension from the Sample Basic database. The calculation order is shown on the left. This example assumes that the parent members are not tagged as Dynamic Calc. For more information on Dynamic Calc members, see Chapter 28, “Dynamically Calculating Data Values.”

Figure 403: Year Dimension from the Sample Basic Database



Jan is the first member in the first branch. Jan has no formula so it is not calculated. The same applies to Feb and Mar, the other two members in the branch.

Essbase calculates Qtr1 by consolidating Jan, Feb, and Mar. In this example, these members are added.

Essbase then calculates the Qtr2 through Qtr4 branches in the same way.

Finally, Essbase calculates the Year member by consolidating the values of Qtr1 through Qtr4. Again, in this example, these members are added.

Member Consolidation

You can choose how Essbase consolidates members by applying any calculation operator (+, -, /, *, %, ~) to the members in the database outline.

If an accounts member has a time balance tag (First, Last, or Average), Essbase consolidates it accordingly. For more information on time balance calculations, see Chapter 29, "Calculating Time Series Data."

If a parent member has a label only tag, Essbase does not calculate the parent from its children. If a member has a ~ tag, Essbase does not consolidate the member up to its parent.

Note: If you use dynamic calculations, Essbase may use a different calculation order. For information on the calculation order for dynamically calculated values, see Chapter 28, "Dynamically Calculating Data Values."

Ordering Dimensions in the Database Outline

To ensure the required calculation results, consider the calculation order of the dimensions in the database outline if you do either of these tasks:

- Use calculation operators to divide (/), multiply (*), or calculate percentages (%) for members in the database outline.
- You place formulas on members in the database outline.

You do not need to consider calculation order if you use only calculation operators to add (+) and subtract (–) members in the database outline and you do not use formulas in the outline.

Placing Formulas on Members in the Database Outline

If you place formulas on members in the database outline, consider the calculation order of the dimensions. A formula that is attached to a member on one dimension may be overwritten by a subsequent calculation on another dimension.

For example, the Sample Basic database has a Measures dimension, tagged as accounts, and a Year dimension, tagged as time. Measures is calculated first, and Year second. If you attach a formula to Margin on the Measures dimension, Essbase calculates the formula when it calculates the Measures dimension. Essbase then overwrites the formula when it aggregates the Year dimension. For detailed information, see "Cell Calculation Order" on page 755.

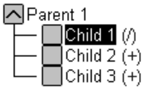
Using the Calculation Operators *, /, and %

If you use calculation operators to multiply (*), divide (/), and calculate percentages (%) for members in the database outline, consider the calculation order of the dimensions. The required calculated values may be overwritten by a subsequent calculation on another dimension.

For example, the Sample Basic database has a Measures dimension, tagged as accounts, and a Year dimension, tagged as time. Measures is calculated first, and Year second. If you multiply members on the Measures dimension, the calculated results may be overwritten when Essbase aggregates values on the Year dimension. For detailed information, see "Cell Calculation Order" on page 755.

When you use a multiplication (*), division (/), or percentage (%) operator to consolidate members, carefully order the members in the branch to achieve the required result.

Figure 404: Calculation Operators in the Database Outline



```
Parent 1
  Child 1 (/)
  Child 2 (+)
  Child 3 (+)
```

In the above example, assume that the user wants to divide the total of Child 2 and Child 3 by Child 1. However, if Child 1 is the first member, Essbase starts with Child 1, taking the value of Parent 1 (currently `#MISSING`) and dividing it by Child 1. The result is `#MISSING`. Essbase then adds Child 2 and Child 3. Obviously, this result is not the required one.

To calculate the correct result, make Child 1 the last member in the branch. For more information on #MISSING values, see Chapter 51, “Optimizing Calculations.”

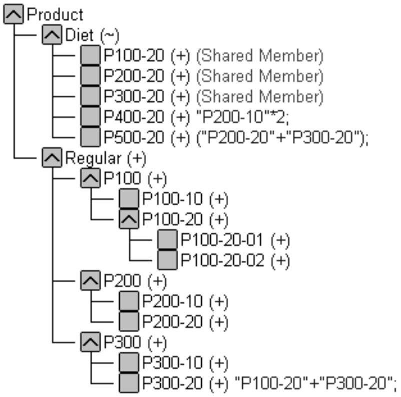
You can apply a formula to a member on the database outline to achieve the same result. However, it is far more efficient to use these calculation operators on members as in the above example.

Avoiding Forward Calculation References

To obtain the calculation results you expect, ensure that the outline does not contain forward calculation references. *Forward calculation references* occur when the value of a calculating member is dependent on a member that Essbase has not yet calculated. In these cases, Essbase may not produce the required calculation results.

For example, consider this Product dimension:

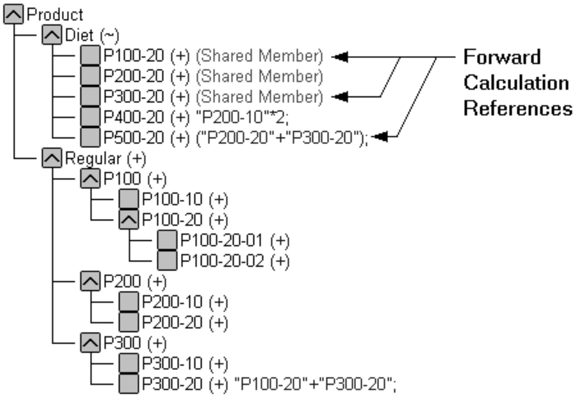
Figure 405: Example Product Dimension



```
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";
```

This Product dimension has three forward calculation references. Two shared members and one non-shared member have forward calculation references:

Figure 406: Example Product Dimension Showing Forward Calculation References

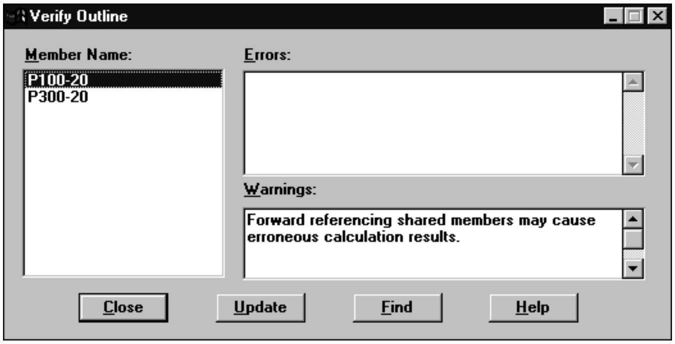


```
Product
  Diet (~)
    P100-20 (+) (Shared Member)
    P200-20 (+) (Shared Member)
    P300-20 (+) (Shared Member)
    P400-20 (+) "P200-10"*2;
    P500-20 (+) ("P200-20"+"P300-20");
  Regular (+)
    P100 (+)
      P100-10 (+)
      P100-20 (+)
        P100-20-01 (+)
        P100-20-02 (+)
    P200 (+)
      P200-10 (+)
      P200-20 (+)
    P300 (+)
      P300-10 (+)
      P300-20 (+) "P100-20"+"P300-20";
```

In the Application Manager Outline Editor, you can select Outline > Verify to identify shared members with forward calculation references. Essbase displays these members in the Verify Outline dialog box.

Note: Selecting Outline > Verify does *not* identify non-shared members that have forward calculation references.

Figure 407: Verify Outline Dialog Box Showing Forward Calculation References



Verify Outline

Member Name:

P100-20
P300-20

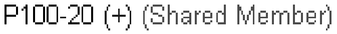
Errors:

Warnings:

Forward referencing shared members may cause erroneous calculation results.

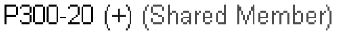
You can save and use an outline containing forward calculation references.

Consider the five members under Diet. The members P100-20, P300-20, and P500-20 have forward calculation references:



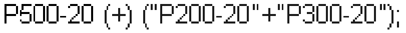
```
P100-20 (+) (Shared Member)
```

Essbase calculates the shared member P100-20 before it calculates the real member P100-20. Because the real member P100-20 has children, Essbase needs to calculate the real member by adding its children before it can accurately calculate the shared member P100-20.



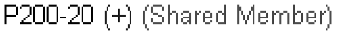
```
P300-20 (+) (Shared Member)
```

Essbase calculates the shared member P300-20 before it calculates the real member P300-20. Because the real member P300-20 has a formula, Essbase needs to calculate the real member before it can accurately calculate the shared member P300-20.



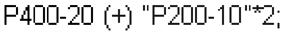
```
P500-20 (+) ("P200-20"+"P300-20");
```

The formula applied to P500-20 references members that Essbase has not yet calculated. One referenced member, P300-20, has its own formula, and Essbase needs to calculate P300-20 before it can accurately calculate P500-20. The members P200-20 and P400-20 calculate correctly, as they do not have forward calculation references:



```
P200-20 (+) (Shared Member)
```

P200-20 is *not* a forward calculation reference, even though Essbase calculates the shared member P200-20 before it calculates the real member P200-20. The real member P200-20 has no calculation dependencies (no children and no formula). Therefore Essbase does not need to calculate the real member before the shared member. Essbase simply takes the value of the real member.

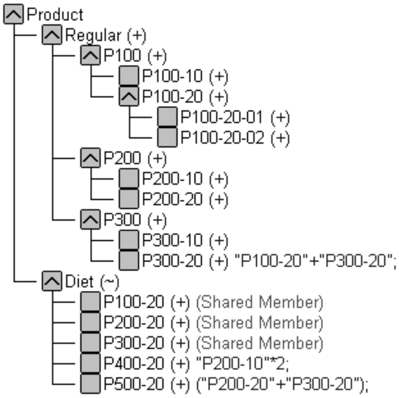


```
P400-20 (+) "P200-10"*2;
```

P400-20 is *not* a forward calculation reference, even though the formula that is applied to P400-20 references a member that Essbase has not yet calculated. The member referenced in the formula does not itself have calculation dependencies. P200-10 is the only member in the formula, and P200-10 does not itself have children or a formula. Essbase accurately calculates P400-20.

To get accurate calculation results for P100-20, P300-20, and P500-20, change the order of members in the outline. By placing the Diet shared members after the Regular members, you ensure that Essbase calculates the members in the required order.

Figure 408: Changed Product Dimension Without Forward Calculation References



Now Essbase calculates as follows:

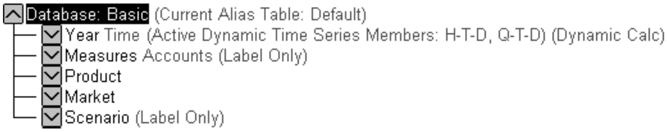
- The real member P100-20 before it calculates the shared member P100-20. So, P100-20 no longer has a forward calculation reference.
- The real member P300-20 before the shared member P300-20. So, P300-20 no longer has a forward calculation reference.
- The referenced member with a formula, P300-20, before the member P500-20. So, P500-20 no longer has a forward calculation reference.

Block Calculation Order

Essbase calculates blocks in the order in which the blocks are numbered. Essbase takes the first sparse dimension in a database outline as a starting point. It defines the sparse member combinations from this first dimension.

In the Sample Basic database, Product is the first sparse dimension in the database outline.

Figure 409: Dimensions in the Sample Basic Database

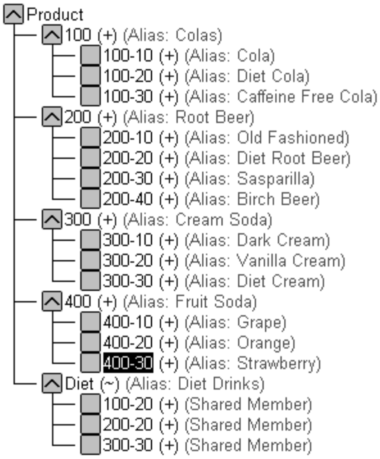


```
Database: Basic (Current Alias Table: Default)
├── Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
├── Measures Accounts (Label Only)
├── Product
├── Market
└── Scenario (Label Only)
```

Note: The attribute dimensions in the Sample Basic outline (not shown in the figure above), are not included in the database consolidation and do not affect block calculation order. For more information on attribute dimensions, see Chapter 9, "Working with Attributes."

Product has 19 members (excluding the shared members, for which Essbase does not create data blocks). Therefore, the first 19 data blocks in the database are numbered according to the calculation order of members in the Product dimension.

Figure 410: Product Dimension from the Sample Basic Database



```
Product
├── 100 (+) (Alias: Colas)
│   ├── 100-10 (+) (Alias: Cola)
│   ├── 100-20 (+) (Alias: Diet Cola)
│   └── 100-30 (+) (Alias: Caffeine Free Cola)
├── 200 (+) (Alias: Root Beer)
│   ├── 200-10 (+) (Alias: Old Fashioned)
│   ├── 200-20 (+) (Alias: Diet Root Beer)
│   ├── 200-30 (+) (Alias: Sasparilla)
│   └── 200-40 (+) (Alias: Birch Beer)
├── 300 (+) (Alias: Cream Soda)
│   ├── 300-10 (+) (Alias: Dark Cream)
│   ├── 300-20 (+) (Alias: Vanilla Cream)
│   └── 300-30 (+) (Alias: Diet Cream)
├── 400 (+) (Alias: Fruit Soda)
│   ├── 400-10 (+) (Alias: Grape)
│   ├── 400-20 (+) (Alias: Orange)
│   └── 400-30 (+) (Alias: Strawberry)
└── Diet (~) (Alias: Diet Drinks)
    ├── 100-20 (+) (Shared Member)
    ├── 200-20 (+) (Shared Member)
    └── 300-30 (+) (Shared Member)
```

The other sparse dimension is Market. The first 19 data blocks contain the first member to be calculated in the Market dimension, which is New York.

This table shows the sparse member combinations for the first 5 of these 19 data blocks.

Block #	Product Member	Market Member
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

The next member in the Market dimension is Massachusetts. Essbase creates the next 19 data blocks for sparse combinations of each Product member and Massachusetts.

This table shows the sparse member combinations for the block numbers 19 through 23.

Block #	Product Member	Market Member
19	Cola (100-10)	Massachusetts
20	Diet Cola (100-20)	Massachusetts
21	Caffeine Free Cola (100-30)	Massachusetts
22	Colas (100)	Massachusetts
23	Old Fashioned (200-10)	Massachusetts

Essbase continues until blocks have been created for all combinations of sparse dimension members for which at least one data value exists.

Essbase creates a data block only if at least one value exists for the block. For example, if no data values exist for Old Fashioned Root Beer (200-10) in Massachusetts, then Essbase does not create a data block for 200-10 -> Massachusetts. However, Essbase does reserve the appropriate block number for 200-10 -> Massachusetts in case data is loaded for that member combination in the future.

When you run a default calculation (CALC ALL) on a database, each block is processed in order, according to its block number. If you have Intelligent Calculation turned on and if the block does not need to be calculated, then Essbase skips the block and moves on to the next block. For more information, see Chapter 52, “Optimizing with Intelligent Calculation.”

Data Block Renumbering

Essbase renumbers the data blocks when you make any of these changes:

- Move a sparse dimension
- Add a sparse dimension
- Change a dense dimension to a sparse dimension
- Move any member in a sparse dimension
- Delete any member in a sparse dimension
- Add a member to a sparse dimension

Cell Calculation Order

Each data block contains all the dense dimension member values for its unique combination of sparse dimension members. Each data value is contained in a cell of the data block.

The order in which Essbase calculates the cells within each block depends on how you have configured the database. How you have configured the database defines the member calculation order of dense dimension members *within each block*. It also defines the calculation order of blocks that represent sparse dimension members.

Use these sections to understand cell calculation order in more detail:

- “Cell Calculation Order: Example 1” on page 756
- “Cell Calculation Order: Example 2” on page 757
- “Cell Calculation Order: Example 3” on page 759
- “Cell Calculation Order: Example 4” on page 761
- “Cell Calculation Order for Formulas on a Dense Dimension” on page 763

Cell Calculation Order: Example 1

Consider the simplest case in which both of these conditions are true:

- No dimensions have time or accounts tags.
- The setting for aggregating `#MISSING` values is turned on. For more information, see "Aggregating #MISSING Values" on page 1417.

Market and Year are both dense dimensions. The table shows a subset of the cells in a data block. Data values have been loaded into the input cells. Essbase calculates the shaded cells. The numbers in bold show the calculation order for these cells. The cell with multiple consolidation paths is darkly shaded.

Year -> Market	New York	Massachusetts	East
Jan	112345.00	68754.00	**3**
Feb	135788.00	75643.00	**4**
Mar	112234.00	93456.00	**5**
Qtr1	**1**	**2**	**6**

As described in "Member Calculation Order" on page 745, Essbase calculates dense dimensions in the order that they display in the database outline. Assuming that the Year dimension is displayed before the Market dimension in the database outline, the Year dimension is calculated before the Market dimension.

The cells are calculated in this order:

1. Qtr1 -> New York
2. Qtr1 -> Massachusetts
3. Jan -> East
4. Feb -> East
5. Mar -> East
6. Qtr1 -> East

Qtr1 -> East has multiple consolidation paths. It can be consolidated on Market or on Year. When consolidated on Market, it is an aggregation of Qtr1 -> New York and Qtr1 -> Massachusetts. When consolidated on Year, it is an aggregation of Jan -> East, Feb -> East, and Mar -> East.

Essbase knows that Qtr1 -> East has multiple consolidation paths. Therefore, it calculates Qtr1 -> East only once and uses the consolidation path of the dimension calculated last. In the above example, this dimension is Market.

The results are shown in this table:

Year/Market	New York	Massachusetts	East
Jan	112345.00	68754.00	181099.00
Feb	135788.00	75643.00	211431.00
Mar	112234.00	93456.00	205690.00
Qtr1	360367.00	237853.00	598220.00

Note: Qtr1 -> East has been calculated only once by aggregating the values for Qtr1.

From the calculation order, you can see that if you place a member formula on Qtr1 in the database outline, Essbase ignores it when calculating Qtr1 -> East. If you place a member formula on East in the database outline, the formula is calculated when Essbase consolidates Qtr1 -> East on the Market consolidation path. If required, you can use a calculation script to calculate the dimensions in the order you choose. For more information, see Chapter 30, “Developing Calculation Scripts.”

Cell Calculation Order: Example 2

Consider a second case in which both of these conditions are true:

- No dimensions have time or accounts tags.
- The setting for aggregating `#MISSING` values is turned off (the default). For more information, see Chapter 51, “Optimizing Calculations.”

Market and Year are both dense dimensions. The table shows a subset of the cells in a data block. Data values have been loaded into the input cells. Essbase calculates the shaded cells. The numbers in bold show the calculation order for these cells. The cell with multiple consolidation paths is darkly shaded.

Year -> Market	New York	Massachusetts	East
Jan	112345.00	68754.00	**4**
Feb	135788.00	75643.00	**5**
Mar	112234.00	93456.00	**6**
Qtr1	**1**	**2**	**3/7**

As described in "Member Calculation Order" on page 745, Essbase calculates dense dimensions in the order they are defined in the database outline. Assuming the Year dimension is positioned before the Market dimension in the database outline, the Year dimension is calculated before the Market dimension.

The cells are calculated in this order:

1. Qtr1 -> New York
2. Qtr1 -> Massachusetts
3. Qtr1 -> East
4. Jan -> East
5. Feb -> East
6. Mar -> East
7. Qtr1 -> East

In this case Qtr1 -> East is calculated on both the Year and Market consolidation paths. First, it is calculated as an aggregation of Qtr1 -> New York and Qtr1 -> Massachusetts. Second, it is calculated as an aggregation of Jan -> East, Feb -> East, and Mar -> East.

The results are identical to the previous case. However, Qtr1 -> East has been calculated twice. This fact is significant when you need to load data at parent levels. For more information, see "Cell Calculation Order: Example 3" on page 759.

Year/Market	New York	Massachusetts	East
Jan	112345.00	68754.00	**181099.00**
Feb	135788.00	75643.00	**211431.00**
Mar	112234.00	93456.00	**205690.00**
Qtr1	**360367.00**	**237853.00**	**598220.00**

From the calculation order, you can see that if you place a member formula on Qtr1 in the database outline, its result is overwritten when Essbase consolidates Qtr1 -> East on the Market consolidation path. If you place a member formula on East in the database outline, the result is retained because the Market consolidation path is calculated last.

Cell Calculation Order: Example 3

Consider the previous case in which both of these conditions are true:

- No dimensions have time or accounts tags.
- The setting for aggregating `#MISSING` values is turned off (the default). For more information, see "Aggregating #MISSING Values" on page 1417.
- Data values have been loaded at a parent levels.

Market and Year are both dense dimensions. The table shows a subset of the cells in a data block. Data values have been loaded into cells at the parent level.

Year -> Market	New York	Massachusetts	East
Jan	#MISSING	#MISSING	181099.00
Feb	#MISSING	#MISSING	211431.00
Mar	#MISSING	#MISSING	205690.00
Qtr1	#MISSING	#MISSING	

As described in “Member Calculation Order” on page 745, Essbase calculates dense dimensions in the order that they are defined in the database outline. Assuming the Year dimension is positioned before the Market dimension in the database outline, the Year dimension is calculated before the Market dimension.

The cells are calculated in the same order as in Example 2. Qtr1 -> East is calculated on both the Year and Market consolidation paths.

Because the setting for aggregating #MISSING values is turned off, Essbase does not aggregate the #MISSING values. Thus, the data that is loaded at parent levels is not overwritten by the #MISSING values below it.

However, if any of the child data values were not #MISSING, these values would be consolidated and would overwrite the parent values. For example, if Jan -> New York contained 50000.00, this value would overwrite the values that were loaded at parent levels.

Essbase first correctly calculates the Qtr1 -> East cell by aggregating Jan -> East, Feb -> East, and Mar -> East. Second, it calculates on the Market consolidation path. However, it does not aggregate the #MISSING values in Qtr1 -> New York and Qtr1 -> Massachusetts and so the value in Qtr1 -> East is not overwritten.

This table shows the results:

Year/Market	New York	Massachusetts	East
Jan	#MISSING	#MISSING	181099.00
Feb	#MISSING	#MISSING	211431.00
Mar	#MISSING	#MISSING	205690.00
Qtr1	#MISSING	#MISSING	**598220.00**

Essbase needs to calculate the Qtr1 -> East cell twice in order to ensure that a value is calculated for the cell. If Qtr1 -> East is calculated according to only the last consolidation path, the result would be #MISSING, which is not the required result.

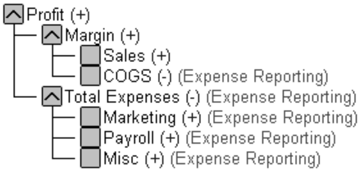
Cell Calculation Order: Example 4

Consider a case in which all of these conditions are true:

- The Year dimension is tagged as time.
- The Measures dimension is tagged as accounts.
- The setting for aggregating `#MISSING` values is turned off (the default). For more information, see "Aggregating #MISSING Values" on page 1417.

Figure 411 shows the Profit branch of the Measures dimension in the Sample Basic database. This example assumes that Total Expenses is not a Dynamic Calc member. For more information on Dynamic Calc members, see Chapter 28, "Dynamically Calculating Data Values."

Figure 411: Profit Branch of the Measures Dimension in Sample Basic Database



```
Profit (+)
    Margin (+)
        Sales (+)
        COGS (-) (Expense Reporting)
    Total Expenses (-) (Expense Reporting)
        Marketing (+) (Expense Reporting)
        Payroll (+) (Expense Reporting)
        Misc (+) (Expense Reporting)
```

This table shows a subset of the cells in a data block. Data values have been loaded into the input cells. Essbase calculates the shaded cells. The numbers in bold show the calculation order for these cells. Cells with multiple consolidation paths are darkly shaded.

The Marketing, Payroll, and Misc Expenses values have been loaded at the Qtr1, parent level.

Measures/Year	Jan	Feb	Mar	Qtr1
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
Total Expenses	**2**	**5**	**8**	**11/16**
Profit	**3**	**6**	**9**	**12/17**

As described in "Member Calculation Order" on page 745, Essbase calculates a dimension tagged as accounts first, followed by a dimension tagged as time. Therefore, in the above example, Measures is calculated before Year.

Three cells have multiple consolidation paths:

- Margin -> Qtr1
- Total Expenses -> Qtr1
- Profit -> Qtr1

Because the setting for aggregating `#MISSING` values is turned off, Essbase does not aggregate the `#MISSING` values. Thus, any data that is loaded at parent levels is not overwritten by the `#MISSING` values and Essbase calculates the three cells with multiple consolidation paths twice.

The results are shown in this table.

Measures -> Year	Jan	Feb	Mar	Qtr1
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**	**52943**

From the calculation order, you can see that if you place a member formula on, for example, Margin in the database outline, its result is overwritten by the consolidation on Qtr1.

Cell Calculation Order for Formulas on a Dense Dimension

The cell calculation order within a data block is not affected by formulas on members. When Essbase encounters a formula in a data block, it locks any other required data blocks, calculates the formula, and proceeds with the data block calculation.

When placing a formula on a dense dimension member, carefully consider the cell calculation order. As described in the examples above, the dimension calculated last overwrites previous cell calculations for cells with multiple consolidation paths. If required, you can use a calculation script to change the order in which the dimensions are calculated. For more information, see Chapter 30, “Developing Calculation Scripts.”

For more information on developing formulas, see Chapter 25, “Developing Formulas.”

Calculation Passes

Whenever possible, Essbase calculates a database in one calculation pass through the database. Thus, it reads each of the required data blocks into memory only once, performing all relevant calculations on the data block and saving it. However, in some situations, Essbase needs to perform more than one calculation pass through a database. On subsequent calculation passes, Essbase brings data blocks back into memory, performs further calculations on them, and saves them again.

When you perform a default, full calculation of a database (CALC ALL), Essbase attempts to calculate the database in one calculation pass. If you have dimensions that are tagged as accounts or time, Essbase may have to do more than one calculation pass through the database.

This table shows the number of calculation passes Essbase performs if you have dimensions that are tagged as time or accounts, and you have at least one formula on the accounts dimension.

Dimension Tagged As:		**Calculation Passes**	**During each calculation pass, Essbase calculates based on:**
Accounts	**Time**		
Dense or Sparse	None	1	All dimensions
Dense	Dense	1	All dimensions
Dense	Sparse	2	Pass 1: Accounts and time dimensions Pass 2: Other dimensions
Sparse	Sparse	2	Pass 1: Accounts and time dimensions Pass 2: Other dimensions
Sparse	Dense	2	Pass 1: Accounts dimension Pass 2: Other dimensions

If you are using formulas that are tagged as Two-Pass, Essbase may need to do an *extra* calculation pass to calculate these formulas. For more information on using Two-Pass calculations, see Chapter 51, “Optimizing Calculations.”

When you use a calculation script to calculate a database, the number of calculation passes Essbase needs to perform depends upon the calculation script. For more information, see "Calculation Passes" on page 764 and Chapter 52, "Optimizing with Intelligent Calculation." For more information on grouping formulas and calculations, see Chapter 51, "Optimizing Calculations."

When you calculate a database, Essbase automatically displays the calculation order of the dimensions for each pass through the database and tells you how many times Essbase has cycled through the database during the calculation.

Essbase displays this information in the ESSCMD window and in the application log. To display the application log, select Application > View Event Log from the Application Manager menu.

For each data block, Essbase decides whether to do a dense or a sparse calculation. The type of calculation it chooses depends on the type of values within the data block. When you run a default calculation (CALC ALL) on a database, each block is processed in order, according to its block number.

Essbase calculates the blocks using this procedure:

- If you have Intelligent Calculation turned on and if the block does not need to be calculated (if it is marked as *clean*), then Essbase skips the block and moves on to the next block. For more information, see Chapter 52, "Optimizing with Intelligent Calculation."
- If the block needs recalculating, Essbase checks to see if the block is a level 0, an input, or an upper level block. For definitions of level 0, input, and upper level blocks, see "Data Storage in Data Blocks" on page 743.
- If the block is a level 0 block or an input block, Essbase performs a dense calculation on the block. Each cell in the block is calculated. For more information, see "Cell Calculation Order" on page 755.
- If the block is an upper level block, Essbase either aggregates the values or performs a sparse calculation on the data block.

  The sparse member combination of each upper level block contains at least one parent member. Essbase aggregates or calculates the block based on the parent member's dimension. For example, if the upper level block is for Product -> Florida from the Sample Basic database, then Essbase chooses the Product dimension.

  If the sparse member combination for the block has more than one parent member, Essbase chooses the last dimension in the calculation order that includes a parent member. For example, if the block is for Product -> East and you perform a default calculation on the Sample Basic database, Essbase chooses the Market dimension, which contains East. The Market dimension is last in the default calculation order because it is placed after the Product dimension in the database outline. For more information, see "Member Calculation Order" on page 745.

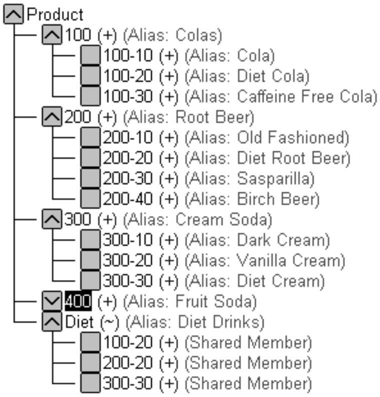
Based on the chosen sparse dimension, Essbase either aggregates the values or performs a sparse calculation on the data block:

- If a formula is applied to the data block member on the chosen sparse dimension, Essbase performs a formula calculation on the sparse dimension. Essbase evaluates each cell in the data block. The formula affects only the member on the sparse dimension, so overall calculation performance is not significantly affected.
- If the chosen sparse dimension is a default consolidation, Essbase aggregates the values, taking the values of the previously calculated child data blocks.

Calculating Shared Members

Shared members are those that share data values with other members. For example, in the Sample Basic database, Diet Cola, Diet Root Beer, and Diet Cream are consolidated under two different parents. They are consolidated under Diet. They are also consolidated under their individual product types: Colas, Root Beer, and Cream Soda.

Figure 412: Calculating Shared Members



```
Product
├─ 100 (+) (Alias: Colas)
│   ├─ 100-10 (+) (Alias: Cola)
│   ├─ 100-20 (+) (Alias: Diet Cola)
│   └─ 100-30 (+) (Alias: Caffeine Free Cola)
├─ 200 (+) (Alias: Root Beer)
│   ├─ 200-10 (+) (Alias: Old Fashioned)
│   ├─ 200-20 (+) (Alias: Diet Root Beer)
│   ├─ 200-30 (+) (Alias: Sasparilla)
│   └─ 200-40 (+) (Alias: Birch Beer)
├─ 300 (+) (Alias: Cream Soda)
│   ├─ 300-10 (+) (Alias: Dark Cream)
│   ├─ 300-20 (+) (Alias: Vanilla Cream)
│   └─ 300-30 (+) (Alias: Diet Cream)
├─ 400 (+) (Alias: Fruit Soda)
└─ Diet (~) (Alias: Diet Drinks)
    ├─ 100-20 (+) (Shared Member)
    ├─ 200-20 (+) (Shared Member)
    └─ 300-30 (+) (Shared Member)
```

The members under the Diet parent are shared members. For more information on shared members, see Chapter 7, “Creating and Changing Database Outlines.”

A calculation on a shared member is a calculation on the real member. If you use the FIX command to calculate a subset of a database and the subset includes a shared member, Essbase calculates the real member.

Chapter

28

Dynamically Calculating Data Values

This chapter explains how you calculate data values dynamically and how you benefit from doing so. Dynamically calculating some of the values in a database can significantly improve the performance of an overall database calculation.

The information in this chapter assumes that you are familiar with the concepts of member combinations, dense and sparse dimensions, and data blocks. For this information, see Part I, “Understanding Hyperion Essbase.”

This chapter includes the following sections:

Understanding Dynamic Calculations

When you design your overall database calculation, it may be more efficient to calculate some member combinations when you retrieve their data, instead of pre-calculating the member combinations during the regular database calculation.

In Essbase, you can define a member to have a *dynamic calculation*. This definition tells Essbase to calculate a data value for the member as users request it. Dynamic calculation shortens regular database calculation time, but may increase retrieval time for the dynamically calculated data values. See "Reducing the Impact on Retrieval Time" on page 784 for more information.

In Essbase you specify dynamic calculations on a per-member basis. You can define a member in your database outline as one of two types of a dynamically calculated member:

- Dynamic Calc
- Dynamic Calc And Store

Use the rest of this section to gain a basic understanding of dynamic calculation:

- "Understanding Dynamic Calc Members" on page 770
- "Understanding Dynamic Calc And Store Members" on page 771
- "Retrieving the Parent Value of Dynamically Calculated Child Values" on page 772

Understanding Dynamic Calc Members

For a member that is tagged as Dynamic Calc, Essbase does not calculate its data value during the regular database calculation; for example, during a CALC ALL. Instead, Essbase calculates the data value upon retrieval; for example, when you retrieve the data into Spreadsheet Add-in.

Specifically, Essbase calculates a data value dynamically when you request the data value in either of two ways:

- By retrieving the data value into Spreadsheet Add-in
- By running a report script that displays the data value

Essbase does not store the calculated value; it recalculates the value for any subsequent retrieval.

Understanding Dynamic Calc And Store Members

Essbase calculates the data value for a member that is tagged as Dynamic Calc And Store when you retrieve the data, in the same way as for a Dynamic Calc member. For a Dynamic Calc And Store member, however, Essbase stores the data value that is calculated dynamically. Subsequent retrievals of that data value do not require recalculation, unless Essbase detects that the value needs recalculating.

Recalculation of Data

When Essbase detects that the data value for a Dynamic Calc And Store member needs recalculating, it places an indicator on the data block that contains the value, so that it knows to recalculate the block on the next retrieval of the data value.

Essbase places the indicator on the data block containing the value and not on the data value itself. In other words, Essbase tracks Dynamic Calc And Store members at the data block level. For more information on data blocks, see Chapter 3, "Basic Architectural Elements."

If the data block needs recalculating, Essbase detects the need and places an indicator on the data block when any of the following occur:

- You do a regular (batch) calculation.
- You restructure the database.
- You use the CLEARBLOCK DYNAMIC calculation command. For more information, see the *Technical Reference* in the `docs` directory.

Essbase recalculates the indicated data blocks when you next retrieve the data value.

Effect of Updated Values on Recalculation

Essbase does *not* detect that a data block needs recalculating and does *not* place an indicator on the data block when you update the data; that is, updated blocks are recalculated only during the next batch calculation. Consider the following two scenarios:

- You do a data load.
- You do a Lock and Send from Essbase Spreadsheet Add-in.

If you load data into the children of a Dynamic Calc And Store member, and the member is a consolidation of its child members, Essbase does *not* know to recalculate the Dynamic Calc And Store member during the next retrieval. The parent member is recalculated only during the next batch calculation.

After loading data, you need to do a regular batch calculation of the database or use the CLEARBLOCK DYNAMIC calculation command to ensure that the Dynamic Calc And Store members are recalculated. For more information on CLEARBLOCK DYNAMIC, see "Clearing data and data blocks" on page 792 and the *Technical Reference* in the `docs` directory.

Retrieving the Parent Value of Dynamically Calculated Child Values

If you retrieve a parent value that is calculated from Dynamic Calc or Dynamic Calc And Store child members, Essbase must dynamically calculate the child member combinations before calculating the parent value. Essbase does not store the child values, even if they are Dynamic Calc And Store members.

For example, assume that Market is a parent member and that East and West are Dynamic Calc And Store child members that consolidate up to Market. When you retrieve a data value for Market, Essbase calculates East and West, even though you have not specifically retrieved them. However, Essbase does not store the values of East or West.

Benefitting from Dynamic Calculations

Dynamically calculating some database values can significantly improve the performance of an overall database calculation.

When you tell Essbase to calculate some data values dynamically, you achieve the following advantages:

- Reduce the regular calculation time of the database because Essbase has fewer member combinations to calculate.
- Reduce disk usage because Essbase stores fewer calculated data values. Database size and index size are reduced. For more information on database and index sizes, see Chapter 38, "Managing Essbase Kernel Settings."

- Reduce database restructure time. For example, adding or deleting a Dynamic Calc member in a dense dimension does not change the data block size, and so Essbase does not need to restructure the database for such additions and deletions. For more information, see "Restructuring a Database" on page 795.
- Reduce the time that is required to back up the database. Because database size is reduced, Essbase takes less time to do a backup.

Data values that Essbase calculates dynamically can take longer to retrieve. You can estimate the retrieval time for dynamically calculated members. See "Reducing the Impact on Retrieval Time" on page 784.

Using Dynamic Calculations

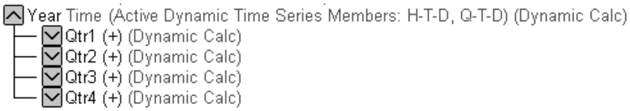
You can tag any member as Dynamic Calc or Dynamic Calc And Store, except the following:

- Level 0 members that do not have a formula
- Label-Only members
- Shared members

Which members you choose to calculate dynamically depends on your database structure and on the balance between (1) your need for reduced calculation time and disk usage and (2) your need for speedy data retrieval for users. See "Choosing Values to Calculate Dynamically" on page 774.

Outline Editor in Application Manager shows which members are Dynamic Calc and which members are Dynamic Calc And Store.

Figure 413: Sample Basic Outline Showing Dynamic Calc Members



```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
├── Qtr1 (+) (Dynamic Calc)
├── Qtr2 (+) (Dynamic Calc)
├── Qtr3 (+) (Dynamic Calc)
└── Qtr4 (+) (Dynamic Calc)
```

In Spreadsheet Add-in, users can display visual cues to distinguish dynamically calculated values. For more information, see the *Essbase Spreadsheet Add-in User's Guide*.

When developing spreadsheets that include dynamically calculated values, spreadsheet designers may want to use the spreadsheet Navigate Without Data option, so that Essbase does not dynamically calculate and store values while test spreadsheets are being built.

Choosing Values to Calculate Dynamically

Dynamically calculating some data values decreases calculation time, lowers disk usage, and reduces database restructure time, but increases retrieval time for dynamically calculated data values.

Use the guidelines described in the following sections when deciding which members to calculate dynamically:

- "Tagging Dense Dimension Members" on page 774
- "Tagging Sparse Dimension Members" on page 775
- "Tagging Two-Pass Members" on page 775
- "Dynamic Calculations and Parent-Child Relationships" on page 775
- "Calculation Scripts and Dynamic Calculations" on page 776
- "Formulas and Dynamically Calculated Members" on page 776
- "Dynamically Calculated Children of Regular Members" on page 777
- "Choosing Between Dynamic Calc and Dynamic Calc And Store" on page 778

Tagging Dense Dimension Members

Consider making the following changes to members of dense dimensions:

- Tag upper level members of dense dimensions as Dynamic Calc.
- Try tagging level 0 members of dense dimensions with simple formulas as Dynamic Calc, and assess the increase in retrieval time.
- Do *not* tag members of dense dimensions as Dynamic Calc And Store.

Simple formulas are formulas that do not require Essbase to perform an expensive calculation. Formulas containing, for example, financial functions or cross-dimensional operators (->) are complex formulas.

Tagging Sparse Dimension Members

Consider making the following changes to members of sparse dimensions:

- Tag some upper level members of sparse dimensions that have six or fewer children as Dynamic Calc or Dynamic Calc And Store.
- Tag sparse-dimension members with complex formulas as Dynamic Calc or Dynamic Calc And Store. A complex formula requires Essbase to perform an expensive calculation. For example, any formula that contains a financial function is a complex formula. For more information, see "Complex Formulas" on page 1394.
- Tag upper level members in a dimension that you frequently restructure as Dynamic Calc or Dynamic Calc And Store.
- Do *not* tag upper level, sparse-dimension members that have 20 or more descendants as Dynamic Calc or Dynamic Calc And Store.

For more information, see "Choosing Between Dynamic Calc and Dynamic Calc And Store" on page 778.

Tagging Two-Pass Members

To reduce the amount of time needed to perform batch calculations, tag two-pass members as Dynamic Calc. You can tag any Dynamic Calc or Dynamic Calc And Store member as two-pass, even if the member is not on an accounts dimension.

For more information about two-pass calculation, see "Using Two-Pass Calculation" on page 1406. For details about the interaction of members tagged as two-pass and attribute members, see Table 14 on page 9-232.

Dynamic Calculations and Parent-Child Relationships

If a parent member has a single child member and you tag the *child* as Dynamic Calc, you must tag the parent as Dynamic Calc. Similarly, if you tag the *child* as Dynamic Calc And Store, you must tag the parent as Dynamic Calc And Store. However, if a parent member has a single child member and the parent is a Dynamic Calc or Dynamic Calc And Store member, you do not have to tag the child as Dynamic Calc or Dynamic Calc And Store.

Calculation Scripts and Dynamic Calculations

When Essbase calculates, for example, a CALC ALL or CALC DIM statement in a calculation script, it bypasses the calculation of Dynamic Calc and Dynamic Calc And Store members.

Similarly, if a member set function (for example, @CHILDREN or @SIBLINGS) is used to specify the list of members to calculate, Essbase bypasses the calculation of any Dynamic Calc or Dynamic Calc And Store members in the resulting list.

If you you specify a Dynamic Calc or Dynamic Calc and Store member explicitly in a calculation script, the calculation script will fail. You cannot do a calculation script calculation of a Dynamic Calc or Dynamic Calc And Store member. If you want to use a calculation script to calculate a member explicitly, do not tag the member as Dynamic Calc. For example, the following calculation script is valid only if Qtr1 is *not* a Dynamic Calc member:

```
FIX (East, Colas)
Qtr1;
ENDFIX
```

Formulas and Dynamically Calculated Members

You *can* include a dynamically calculated member in a formula when you apply the formula to the database outline. For example, if Qtr1 is a Dynamic Calc member, you could place the following formula on Qtr1 in the database outline:

```
Qtr1 = Jan + Feb;
```

You *cannot* make a dynamically calculated member the target of a formula calculation in a calculation script; Essbase does not reserve memory space for a dynamically calculated value and, therefore, cannot assign a value to it. For example, if Qtr1 is a Dynamic Calc or Dynamic Calc And Store member, Essbase displays a syntax error if you include the following formula in a calculation script:

```
Qtr1 = Jan + Feb;
```

However, if Qtr1 is a Dynamic Calc or Dynamic Calc And Store member and Year is a regular member, you can use the following formula in a calculation script:

```
Year = Qtr1 + Qtr2;
```

This formula is valid because Essbase is not assigning a value to the dynamically calculated member.

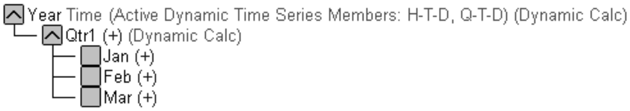
Note: When you reference a dynamically calculated member in a formula in the database outline or in a calculation script, Essbase interrupts the regular calculation to do the dynamic calculation. This interruption can significantly lower calculation performance.

Dynamically Calculated Children of Regular Members

If the calculation of a regular member depends on the calculation of Dynamic Calc or Dynamic Calc And Store child members, then Essbase has to calculate the child members during the regular database calculation in order to calculate the parent. Therefore, there is no reduction in regular calculation time. This requirement applies to members of sparse dimensions and members of dense dimensions.

For example, consider the following outline:

Figure 414: Sample Basic Outline Showing Qtr1 as a Dynamic Calc Member



```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
└─ Qtr1 (+) (Dynamic Calc)
   ├─ Jan (+)
   ├─ Feb (+)
   └─ Mar (+)
```

Jan, Feb, and Mar are regular members, Qtr1 is a Dynamic Calc member, and Year is a regular member. When Essbase calculates Year during the regular database calculation, it has to calculate Qtr1 in order to consolidate it up to Year. Essbase calculates Qtr1 even though Qtr1 is a Dynamic Calc member.

Choosing Between Dynamic Calc and Dynamic Calc And Store

In most cases you can optimize calculation performance and lower disk usage by using Dynamic Calc members instead of Dynamic Calc And Store members. However, in specific situations, using Dynamic Calc And Store members is optimal:

- "Recommendations for Sparse Dimension Members" on page 778
- "Recommendations for Members with Specific Characteristics" on page 779
- "Recommendations for Dense Dimension Members" on page 779
- "Recommendations for Data with Many Concurrent Users" on page 780

Recommendations for Sparse Dimension Members

In most cases, when you want to calculate a sparse dimension member dynamically, tag the member as Dynamic Calc instead of Dynamic Calc And Store. When Essbase calculates data values for a member combination that includes a Dynamic Calc member, Essbase calculates only the requested values of the relevant data block. These values can be a subset of the data block.

However, when Essbase calculates data values for a member combination that includes a Dynamic Calc And Store member, Essbase needs to calculate and store the whole data block, even if the requested data values are a subset of the data block. Thus, the calculation takes longer and the initial retrieval time is greater.

Essbase stores only the data blocks that contain the requested data values. If Essbase needs to calculate any intermediate data blocks in order to calculate the requested data blocks, it does not store the intermediate blocks.

Calculating the intermediate data blocks can significantly increase the initial retrieval time. For example, in the Sample Basic database, Market and Product are the sparse dimensions. Assume that Market and the children of Market are Dynamic Calc And Store members. When a user retrieves the data value for the member combination Market -> Cola -> Jan -> Actual -> Sales, Essbase calculates and stores the Market -> Cola data block. In order to calculate and store Market -> Cola, Essbase calculates the intermediate data blocks: East -> Cola, West -> Cola, South -> Cola, and Central -> Cola. Essbase does not store these intermediate data blocks.

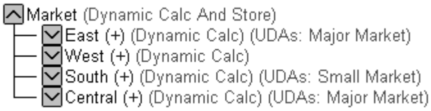
Recommendations for Members with Specific Characteristics

Using Dynamic Calc And Store may slow initial retrieval; however, subsequent retrievals are faster than for Dynamic Calc members. Use Dynamic Calc And Store instead of Dynamic Calc for the following members:

- An upper-level sparse dimension member with children on a remote database. Essbase needs to retrieve the value from the remote database, which increases retrieval time. For more information on partitions, see "Dynamically Calculating Data in Partitions" on page 797.
- A sparse dimension member with a complex formula. A complex formula requires Essbase to perform an expensive calculation. For example, any formula that contains a financial function or a cross-dimensional member is a complex formula.
- If users frequently retrieve an upper level member of a sparse dimension, speedy retrieval time is very important.

For example, in the Sample Basic database, if most users retrieve data at the Market level, you probably want to tag Market as Dynamic Calc And Store and its children as Dynamic Calc.

Figure 415: Sample Basic Outline, Market is Dynamic Calc And Store Member



Market (Dynamic Calc And Store)
East (+) (Dynamic Calc) (UDAs: Major Market)
West (+) (Dynamic Calc)
South (+) (Dynamic Calc) (UDAs: Small Market)
Central (+) (Dynamic Calc) (UDAs: Major Market)

Recommendations for Dense Dimension Members

Use Dynamic Calc members for dense dimension members. Defining members as Dynamic Calc And Store on a dense dimension provides only a small decrease in retrieval time and in regular calculation time. In addition, database size (disk usage) does not decrease significantly because Essbase reserves space for the member's data values in the data block.

Recommendations for Data with Many Concurrent Users

Use Dynamic Calc members for data with concurrent users. If many users are concurrently retrieving Essbase data, the initial retrieval time for Dynamic Calc And Store members can be significantly higher than for Dynamic Calc members.

Dynamic Calc And Store member retrieval time increases as the number of concurrent user retrievals increases. However, Dynamic Calc member retrieval time does not increase as concurrent user retrievals increase.

If many users are concurrently accessing data, you may see significantly lower retrieval times if you use Dynamic Calc members instead of Dynamic Calc And Store members.

Dynamic Calculation Changes to Calculation Order

Using dynamically calculated data values changes the order in which Essbase calculates the values and can have implications for the way you administer a database:

- "Calculation Order for Dynamic Calculations" on page 780
- "Calculation Order for Dynamically Calculating Two-Pass Members" on page 781
- "Calculation Order for Asymmetric Data" on page 782

Calculation Order for Dynamic Calculations

When Essbase dynamically calculates data values, it calculates the data in an order that is different from the regular database calculation order. For detailed information on calculation order, see Chapter 27, "Defining the Calculation Order."

During regular calculations, Essbase calculates the database in this order:

1. Dimension tagged as accounts
2. Dimension tagged as time
3. Other dense dimensions (in the order they appear in the database outline)

4. Other sparse dimensions (in the order they appear in the database outline)
5. Two-pass calculations

For dynamically calculated values, on retrieval, Essbase calculates the values by calculating the database in the following order:

1. Sparse dimensions:
   a. If the dimension tagged as time is sparse, and the database outline uses time series data, Essbase bases the sparse calculation on the time dimension.
   b. Otherwise, Essbase uses the dimension that it uses for a regular calculation. For more information, see Chapter 27, “Defining the Calculation Order.”
2. Dense dimensions:
   a. Dimension tagged as accounts, if dense
   b. Dimension tagged as time, if dense
   c. Time series calculations
   d. Remaining dense dimensions
   e. Two-pass calculations

Calculation Order for Dynamically Calculating Two-Pass Members

Consider the following information to ensure that Essbase produces the required calculation result when it dynamically calculates data values for members that are tagged as two-pass. For more information on two-pass calculations, see “Using Two-Pass Calculation” on page 1406.

If more than one Dynamic Calc or Dynamic Calc And Store dense dimension member is tagged as two-pass, Essbase performs the regular dynamic calculation and then calculates the two-pass members in the following order:

1. Two-pass members in the accounts dimension, if any exist.
2. Two-pass members in the time dimension, if any exist.
3. Two-pass members in the remaining dense dimensions in the order in which the dimensions appear in the outline.

For example, in the Sample Basic database, assume the following:

- Margin% in the dense Measures dimension (the dimension tagged as accounts) is tagged as both Dynamic Calc and two-pass.
- Variance in the dense Scenario dimension is tagged as both Dynamic Calc and two-pass.

Essbase calculates the accounts dimension member first. So, Essbase calculates Margin% (from the Measures dimension) and then calculates Variance (from the Scenario dimension).

If Scenario is a sparse dimension, Essbase calculates Variance first, following the regular calculation order for dynamic calculations. See "Calculation Order for Dynamic Calculations" on page 780. Essbase then calculates Margin%.

This calculation order does not produce the required result because Essbase needs to calculate Margin % -> Variance using the formula on Margin %, and not the formula on Variance. You can avoid this problem by making Scenario a dense dimension. This problem does not occur if the Measures dimension (the accounts dimension) is sparse, because Essbase then still calculates Margin% first.

Calculation Order for Asymmetric Data

Because the calculation order used for dynamic calculations differs from the calculation order used for regular database calculations, in some database outlines you may get different calculation results if you tag certain members as Dynamic Calc or Dynamic Calc And Store. These differences happen when Essbase dynamically calculates asymmetric data.

Symmetric data calculations produce the same results no matter which dimension is calculated. Asymmetric data calculations calculate differently along different dimensions.

Table 36: Example of a Symmetric Calculation

Time -> Accounts	Jan	Feb	Mar	Qtr1
Sales	100	200	300	600
COGS	50	100	150	300
Profit **(Sales – COGS)**	50	100	150	300

The calculation for Qtr1 -> Profit produces the same result whether you calculate along the dimension tagged as time or the dimension tagged as accounts. Calculating along the time dimension, add the values for Jan, Feb, and Mar:

50+100+150=300

Calculating along the accounts dimension, subtract Qtr1 -> COGS from Qtr1 -> Sales:

600–300=300.

Table 37: Example of an Asymmetric Calculation

Market -> Accounts	New York	Florida	Connecticut	East
UnitsSold	10	20	20	50
Price	5	5	5	15
Sales (Price * UnitsSold)	50	100	100	250

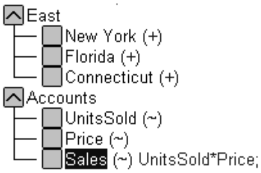
The calculation for East -> Sales produces the correct result when you calculate along the Market dimension, but produces an incorrect result when you calculate along the accounts dimension. Calculating along the Market dimension, adding the values for New York, Florida, and Connecticut produces the correct results:

50+100+100=250

Calculating along the accounts dimension, multiplying the value East -> Price by the value East -> UnitsSold produces incorrect results:

15 * 50=750

In this outline, East is a sparse dimension, and Accounts is a dense dimension:



```
East
    New York (+)
    Florida (+)
    Connecticut (+)
Accounts
    UnitsSold (~)
    Price (~)
    Sales (~) UnitsSold*Price;
```

If East and Sales are tagged as Dynamic Calc, then Essbase calculates a different result than it does if East and Sales are not tagged as Dynamic Calc.

If East and Sales are not Dynamic Calc members, Essbase produces the correct result by calculating as follows:

1. The dense Accounts dimension, calculating the values for UnitsSold, Price, and Sales for New York, Florida, and Connecticut.
2. The sparse East dimension, by aggregating the calculated values for UnitsSold, Price, and Sales for New York, Florida, and Connecticut to obtain the Sales values for East.

If East and Sales are Dynamic Calc members, Essbase produces an incorrect result by calculating as follows:

1. The sparse East dimension, by aggregating the values for UnitsSold, Price, and Sales for New York, Florida, and Connecticut to obtain the values for East.
2. The values for East -> Sales, by taking the aggregated values in the East data blocks and performing a formula calculation with these values to obtain the value for Sales.

To avoid this problem and ensure that you obtain the required results, do not tag the Sales member as Dynamic Calc or Dynamic Calc And Store.

Reducing the Impact on Retrieval Time

The increase in retrieval time when you dynamically calculate a member of a dense dimension is not significant unless the member contains a complex formula.

Note: The increase in retrieval time may be significant when you tag members of sparse dimensions as Dynamic Calc or Dynamic Calc And Store.

The following sections discuss ways you can analyze and manage the effect of Dynamic Calc members on a database:

- "Displaying a Retrieval Factor" on page 785
- "Displaying a Summary of Dynamically Calculated Members" on page 786
- "Increasing the Retrieval Buffer Size" on page 787
- "Using Dynamic Calculator Caches" on page 788

Note: For a list of functions that have the most significant effect on query retrieval, see "Choosing Between Member Set Functions and Performance" on page 1417 for details.

Displaying a Retrieval Factor

To help you estimate any increase in retrieval time, Essbase calculates a retrieval factor for a database outline when you save the outline. Essbase calculates this retrieval factor based on the dynamically calculated data block that is the most expensive for Essbase to calculate. The retrieval factor takes into account only aggregations. It does not consider the retrieval impact of formulas.

The retrieval factor is the number of data blocks that Essbase must retrieve from disk or from the database in order to calculate the most expensive block. If the database has Dynamic Calc or Dynamic Calc And Store members in dense dimensions only (no Dynamic Calc or Dynamic Calc And Store members in sparse dimensions), the retrieval factor is 1.

An outline with a high retrieval factor (for example, greater than 2000) can cause long delays when users retrieve data. However, the actual impact on retrieval time also depends on how many dynamically calculated data values a user retrieves. The retrieval factor is only an indicator. In some applications, using Dynamic Calc members may reduce retrieval time because the database size and index size are reduced.

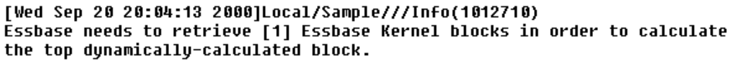
Essbase displays the retrieval factor value in the application log.

➤ To view an estimated retrieval factor in Application Manager:

1. In the application desktop window, select an application and database.
2. Select Application > View Event Log.
3. In the View Log File dialog box, select Display All or Date.
4. Click OK.

   Essbase displays the Log Viewer window for the database you selected.

A message similar to this sample indicates a retrieval factor:



```
[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.
```

This message tells you that Essbase needs to retrieve one block in order to calculate the most expensive dynamically calculated data block.

Tip: You can also use the View Log dialog box in Administration Services to view logs. For more information, see *Essbase Administration Services Online Help*.

Displaying a Summary of Dynamically Calculated Members

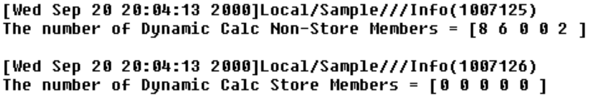
When you add Dynamic Calc or Dynamic Calc And Store members to a database outline and save the outline, Essbase provides a summary of how many members are tagged as Dynamic Calc and Dynamic Calc And Store. Essbase displays the summary in the application log.

➤ To view a summary of dynamically calculated members in Application Manager:

1. In the application desktop window, select an application and database.
2. Select Application > View Event Log.
3. In the View Log File dialog box, select Display All or Date.
4. Click OK.

   Essbase displays the Log Viewer window for the database you selected.

A message similar to this sample is displayed:



```
[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 ]
```

This message tells you that there are eight Dynamic Calc members in the first dimension of the database outline, six in the second dimension, and two in the fifth dimension. Dynamic Time Series members are included in this count. This example does not include Dynamic Calc And Store members.

Essbase includes Dynamic Time Series members in these numbers. For more information, see Chapter 29, “Calculating Time Series Data.”

Increasing the Retrieval Buffer Size

When you retrieve data into Essbase Spreadsheet Add-in or use Report Writer to retrieve data, Essbase uses the retrieval buffer to optimize the retrieval. Essbase processes the data in sections. Increasing the retrieval buffer size can significantly reduce retrieval time because Essbase can process larger sections of data at one time.

By default, the retrieval buffer size is 10 KB. However, you can speed up retrieval time if you set the retrieval buffer size greater than 10 KB.

➤ To set the retrieval buffer size in Application Manager:

1. In the application desktop window, select an application and database.
2. Select Database > Settings.

   Essbase displays the Database Settings dialog box.
3. Type the required size in the Retrieval Buffer Size text box.
4. Click OK.

Tip: You can set the size of the retrieval buffer outside the Essbase Application Manager.

Tool	Instructions	For More Information
Administration Services	Click the General tab on the Database Properties window	*Essbase Application Manager Online Help*
MaxL	**alter database**	*Technical Reference* in the `docs` directory
ESSCMD	SETDBSTATEITEM	

For more information on sizing the retrieval buffer, see “Setting the Retrieval Buffer Size” on page 1446.

Using Dynamic Calculator Caches

By default, when Essbase calculates a Dynamic Calc member in a dense dimension (for example, for a query), it writes all blocks needed for the calculation into an area in memory called the dynamic calculator cache. When Essbase writes these blocks into the dynamic calculator cache, it expands them to include all Dynamic Calc members in the dense dimensions.

Using the Essbase dynamic calculator cache enables centralized control of memory usage for dynamic calculations. Managing data blocks in the dynamic calculator cache also reduces the overall memory space requirement and can improve performance by reducing the number of calls to the operating system to do memory allocations.

Note: The dynamic calculator cache uses a different approach to optimizing calculation performance than the calculator cache. For information about the calculator cache, see "Managing Caches to Improve Performance" on page 1403.

Sizing Dynamic Calculator Caches

Essbase uses a separate dynamic calculator cache for each open database. The single DYNCALCCACHEMAXSIZE setting in the configuration file, `essbase.cfg,` specifies the maximum size for each dynamic calculator cache on the server. By default, the maximum size is 20MB. Essbase allocates area in a dynamic calculator cache for data blocks, as needed, until it has allocated the maximum memory area specified by this setting.

Five additional configuration file settings are relevant to dynamic calculator caches. The optimum settings for the dynamic calculator cache depend on the amount of memory on the server computers, the configuration of the databases, and the nature of user queries. Table 38 describes each setting and includes recommendations on how to determine values for your system. To fit your situation, you may need to test and adjust the settings.

Table 38: Settings for Dynamic Calculator Caches in `essbase.cfg`

DYNCALCCACHEMAXSIZE	
Description	This setting specifies the maximum size Essbase can allocate to each dynamic calculator cache on the server.
Recommended Setting	Recommended Setting value = C * S * U C is the value of the appropriate CALCLOCKBLOCK setting in the `essbase.cfg` file. (The SET LOCKBLOCK command specifies which CALCLOCKBLOCK setting to use.) S is the size of the largest expanded block across all databases on the computer. To calculate the expanded block size, multiply the number of members (including Dynamic Calc members) in each dense dimension together, then multiply the result by the size of each member cell, 8 bytes. For example, for the dense dimensions in Sample Basic, 12 (Year) * 8 (Measures) * 3 (Scenario) * 8 bytes = 2304 bytes. U is the maximum number of expected concurrent users on the database that has the largest number of concurrent users. Assigning the value 0 (zero) to DYNCALCACHEMAXSIZE tells Essbase not to use dynamic calculator caches. The default size is 20 megabytes (20,971,520 bytes).
DYNCALCCACHEONLY	
Description	For situations when the dynamic calculator cache has no more room, this setting specifies whether or not Essbase may perform Dynamic Calc calculations in memory outside the dynamic calculator cache.
Recommended Setting	Setting value = FALSE (The default value). Only set this value to TRUE for one or more of the following circumstances: • The operating system is not properly reclaiming memory outside the dynamic calculator cache. • A severe memory shortage exists. • Tighter control is required over memory usage for dynamic calculations.

Table 38: Settings for Dynamic Calculator Caches in `essbase.cfg` *(Continued)*

DYNCALCCACHEWAITFORBLK	
Description	If Essbase uses up the maximum area for a dynamic calculator cache, this setting tells Essbase whether to wait until space becomes available in the cache or to immediately write and calculate the blocks in memory outside the dynamic calculator cache. If DYNCALCCACHEONLY is set to TRUE, instead of using memory outside the dynamic calculator cache, Essbase generates an error message. If the dynamic calculator cache is too small, it is possible for more than one thread to be in queue, each thread waiting to calculate its data blocks.
Recommended Setting	Setting value = FALSE (The default value). If you cannot add physical memory to the server computer, before setting to TRUE, iteratively increase the value of DYNCALCCACHEMAXSIZE and test until you determine that you cannot spare more memory for the dynamic calculator cache.
DYNCALCCACHEBLKTIMEOUT	
Description	If Essbase is to wait for available space in the dynamic calculator cache, this setting defines how long it will wait.
Recommended Setting	Setting value = WT / B WT is the maximum tolerable wait time for a query; for example, 5 seconds. B is the total number of logical blocks accessed in the largest query. To determine the value of B, check the messages in the application log for the largest number of Dyn.Calc.Cache “Big Block Allocs” for a query, as shown in Figure 416.
DYNCALCCACHEBLKRELEASE	
Description	Essbase uses this setting when it has waited the specified time and space is still not available in the dynamic calculator cache. A TRUE setting tells Essbase to use space in the dynamic calculator cache that is made available by swapping out blocks and temporarily compressing them in a dynamic calculator cache compressed-block buffer. A FALSE setting tells Essbase to write and calculate the blocks immediately outside the dynamic calculator cache. If DYNCALCCACHEONLY is set to TRUE, instead of using memory outside the dynamic calculator cache, Essbase generates an error message.

Table 38: Settings for Dynamic Calculator Caches in `essbase.cfg` *(Continued)*

Recommended Setting	Setting value = FALSE (The default value). Set to TRUE only if you are experiencing severe memory shortage problems.
DYNCALCCACHECOMPRBLKBUFSIZE	
Description	This setting is the size of the dynamic calculator cache compressed-block buffer to be used if Essbase has waited the specified wait time and the DYNCALCCACHEBLKRELEASE setting is TRUE.
Recommended Setting	Setting value = (C * S) / 2 C is the value of the current CALCLOCKBLOCK setting in the `essbase.cfg` file. The SET LOCKBLOCK command specifies which CALCLOCKBLOCK configuration setting is current. S is the size of the largest expanded block across all databases on the computer. Calculate S as described for the DYNCALCCACHEMAXSIZE setting.

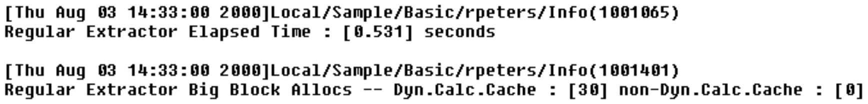
Note: For the new values to take effect, after changing any parameter in the `essbase.cfg` file, you must close and restart Essbase.

For more information about dynamic calculator cache settings, see the *Technical Reference* in the `docs` directory.

Reviewing Dynamic Calculator Cache Usage

Essbase writes two messages to the application log for each data retrieval. As shown in the example in Figure 416, the first message describes the total amount of time required for the retrieval.

Figure 416: Application Log Example of Memory Usage for Data Blocks Containing Dynamic Calc Members



```
[Thu Aug 03 14:33:00 2000]Local/Sample/Basic/rpeters/Info(1001065)
Regular Extractor Elapsed Time : [0.531] seconds

[Thu Aug 03 14:33:00 2000]Local/Sample/Basic/rpeters/Info(1001401)
Regular Extractor Big Block Allocs -- Dyn.Calc.Cache : [30] non-Dyn.Calc.Cache : [0]
```

If a dynamic calculator cache is used, a second message displays the number of blocks calculated within the data calculator cache (Dyn.Calc.Cache: (*n*)) and the number of blocks calculated in memory outside dynamic calculator cache (non-Dyn.Calc.Cache: [*n*]).

To determine if the dynamic calculator cache is being used effectively, review both of these messages and consider what your settings are in the `essbase.cfg` file. For example, if the message indicates that blocks were calculated outside as well as in a dynamic calculator cache, you may need to increase the DYNCALCCACHEMAXSIZE setting. If the specified maximum size is all that you can afford for all dynamic calculator caches on the server and if using regular memory to complete dynamically calculated retrievals results in unacceptable delays (for example, because of swapping or paging activity), set DYNCALCCACHEWAITFORBLK to TRUE.

You can use the GETPERFSTATS command in ESSCMD to view a summary of dynamic calculator cache activity. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, “Automating the Production Environment” for information about ESSCMD

Using Dynamic Calculations with Standard Procedures

Using dynamic calculations with standard Essbase procedures affects these processes:

- Clearing data and data blocks

  You can use the CLEARBLOCK DYNAMIC command to remove data blocks for Dynamic Calc And Store member combinations.

  You can use the CLEARDATA command to mark Dynamic Calc And Store data blocks, so that Essbase knows to recalculate the blocks. The CLEARDATA command has no effect on data values for Dynamic Calc members.

- Copying data

  You cannot copy data to a dynamically calculated data value. You cannot specify a Dynamic Calc or Dynamic Calc And Store member as the target for the DATACOPY calculation command.

- Converting currencies

  You cannot specify a Dynamic Calc or Dynamic Calc And Store member as the target for the CCONV command.

- Loading data

  When you load data, Essbase does not load data into member combinations that contain a Dynamic Calc or Dynamic Calc And Store member. Essbase skips these members during data load. Essbase does not display an error message.

  To place data into Dynamic Calc and Dynamic Calc And Store members, after loading data, you need to ensure that Essbase recalculates Dynamic Calc And Store members. To do this, see "Effect of Updated Values on Recalculation" on page 771.

- Exporting data

  Essbase does not calculate dynamically calculated values before exporting data. Essbase does not export values for Dynamic Calc members. Essbase exports values for Dynamic Calc And Store members only if a calculated value exists in the database from a previous user retrieval of the data.

- Reporting data

  Essbase cannot use the SPARSE data extraction method for dynamically calculated members. The SPARSE data extraction method optimizes performance when a high proportion of the reported data rows are `#MISSING`. For more information, see the <SPARSE command in the *Technical Reference* in the `docs` directory.

- Including dynamic members in calculation scripts

  When calculating a database, Essbase skips the calculation of any Dynamic Calc or Dynamic Calc And Store members. Essbase displays an error message if you attempt to do a member calculation of a Dynamic Calc or Dynamic Calc And Store member in a calculation script. For more information, see "Calculation Scripts and Dynamic Calculations" on page 776.

Creating Dynamic Calc and Dynamic Calc And Store Members

In Application Manager Outline Editor, you can tag members as Dynamic Calc or as Dynamic Calc And Store. When you build a dimension, you can use properties to indicate Dynamic Calc and Dynamic Calc And Store members.

➤ To tag a member as Dynamic Calc using Application Manager:

1. In Outline Editor, open the database.
2. Select the member that you want to tag as Dynamic Calc.
3. Click the button.

   Alternatively, in the Outline Editor menu, select Edit > Properties. Then, in the Data Storage group of the Member Specification dialog box, select Dynamic Calc.

   Essbase labels the member as Dynamic Calc.

➤ To tag a member as Dynamic Calc And Store using Application Manager:

1. In Outline Editor, open the database.
2. Select the member that you want to tag as Dynamic Calc And Store.
3. Click the button.

   Alternatively, in the Outline Editor menu, select Edit > Properties. Then, in the Data Storage group of the Member Specification dialog box, select Dynamic Calc And Store.

   Essbase labels the member as Dynamic Calc And Store.

➤ To remove a Dynamic Calc or Dynamic Calc And Store tag using Application Manager:

1. In Outline Editor, open the database.
2. Select the member from which you want to remove the Dynamic Calc or Dynamic Calc And Store tag.
3. Click the button.

   Alternatively, in the Outline Editor menu, select Edit > Properties. Then, in the Data Storage group of the Member Specification dialog box, select Store Data.

Building Dimensions with Dynamic Calc Members

You can build dimensions with Dynamic Calc and Dynamic Calc And Store members. In the dimension build data file, use the property X for Dynamic Calc and the property V for Dynamic Calc And Store. For more information, see "Setting Member Properties" on page 557

Restructuring a Database

When you add a Dynamic Calc member to a dense dimension, Essbase does not reserve space in the data block for the member's values. Therefore, Essbase does not need to restructure the database. However, when you add a Dynamic Calc And Store member to a dense dimension, Essbase does reserve space in the relevant data blocks for the member's values and therefore needs to restructure the database.

When you add a Dynamic Calc or a Dynamic Calc And Store member to a sparse dimension, Essbase updates the index, but does not change the relevant data blocks. For more information on the database index, see Chapter 38, "Managing Essbase Kernel Settings."

Essbase can save changes to your database outline significantly faster if it does not have to restructure the database.

In the following cases, Essbase does not restructure the database. Essbase only has to save the database outline, which is very fast.

Essbase does *not* restructure the database or change the index when you do any of the following:

- Add, delete, or move a dense dimension Dynamic Calc member. (But Essbase *does* restructure the database if the member is Dynamic Calc And Store.)
- Change a dense dimension Dynamic Calc And Store member to a regular member
- Change a sparse dimension Dynamic Calc or Dynamic Calc And Store member to a regular member
- Rename any Dynamic Calc or Dynamic Calc And Store member

In the following cases, Essbase does not restructure the database, but does have to restructure the database index. Restructuring the index is significantly faster than restructuring the database.

Essbase restructures only the database index when you do either of the following:

- Add, delete, or move sparse dimension Dynamic Calc or Dynamic Calc And Store members
- Change a regular dense dimension member to a Dynamic Calc And Store member

However, Essbase does restructure your database when you do any of the following:

- Add, delete, or move a dense dimension Dynamic Calc And Store member. (But Essbase does *not* restructure the database if the member is Dynamic Calc.)
- Change a dense dimension Dynamic Calc And Store member to a Dynamic Calc member
- Change a dense dimension Dynamic Calc member to a Dynamic Calc And Store member
- Change a dense dimension regular member to a Dynamic Calc member
- Change a dense dimension Dynamic Calc member to a regular member
- Change a sparse dimension regular member to a Dynamic Calc or Dynamic Calc And Store member

For detailed information on the types of database restructuring, see "Understanding Database Restructuring" on page 1343.

Dynamically Calculating Data in Partitions

You can define Dynamic Calc and Dynamic Calc And Store members in transparent, replicated, or linked regions of your partitions. For more information on partitions, see Chapter 13, "Designing Partitioned Applications."

For example, you might want to tag an upper level, sparse dimension member with children that are on a remote database (transparent database partition) as Dynamic Calc And Store. Because Essbase needs to retrieve the child values from the other database, retrieval time is increased. You could use Dynamic Calc instead of Dynamic Calc And Store; however, the impact on subsequent retrieval time might be too great.

For example, assume that your local database is the Corporate database, which has transparent partitions to the regional data for East, West, South, and Central. You could tag the parent member Market as Dynamic Calc And Store.

In a transparent partition, the definition on the remote database takes precedence over any definition on the local database. For example, if a member is tagged as Dynamic Calc in the local database but not in the remote database, Essbase retrieves the value from the remote database and does not do the local calculation.

If you are using a replicated partition, then you might want to use Dynamic Calc members instead of Dynamic Calc And Store members. When calculating replicated data, Essbase does not retrieve the child blocks from the remote database, and therefore the impact on retrieval time is not great.

Note: When Essbase replicates data, it checks the time stamp on each source data block and each corresponding target data block. If the source data block is more recent, Essbase replicates the data in the data block. However, for dynamically calculated data, data blocks and time stamps do not exist. Therefore Essbase always replicates dynamically calculated data.

Chapter 29

Calculating Time Series Data

This chapter explains how to calculate time series data. For example, you can do inventory tracking by calculating the first and last values for a specific time period. You can also calculate period-to-date values.

This chapter includes the following sections:

Calculating First, Last, or Average Values

Using time balance and variance reporting tags on the dimension tagged as accounts, you can tell Essbase how to perform time balance calculations on accounts data.

Essbase usually calculates a parent in the dimension tagged as time by consolidating or calculating the formulas on the parent's children. However, you can use accounts tags, such as time balance and variance reporting tags, to consolidate a different kind of value. For example, if you tag a parent member in the accounts dimension with a time balance property of First, Essbase calculates the member by consolidating the value of the member's first child. For example, in the Sample Basic database, the Opening Inventory member in the Measures dimension (the accounts dimension) has a time balance property of First. This member represents the inventory at the beginning of the time period. If the time period is Qtr1, Opening Inventory represents the inventory available at the beginning of Jan (the first member in the Qtr1 branch).

To use accounts tags, you must have a dimension tagged as accounts and a dimension tagged as time. You use the First, Last, and Average tags (time balance properties) and the Expense tag (variance reporting property) only on members of a dimension tagged as accounts. The dimensions you tag as time and accounts can be either dense or sparse dimensions.

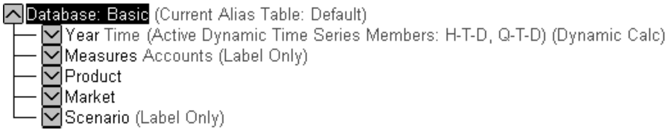
Note: If you are using Intelligent Calculation, changing accounts tags in the database outline does not cause Essbase to restructure the database. You may have to tell Essbase explicitly to recalculate the required data values. See Chapter 52, “Optimizing with Intelligent Calculation.”

Specifying Accounts and Time Dimensions

When you tag a dimension as accounts, Essbase knows that the dimension contains members with accounts tags. When you tag a dimension as time, Essbase knows that this dimension is the one on which to base the time periods for the accounts tags.

In the Sample Basic database, the Measures dimension is tagged as accounts, and the Year dimension is tagged as time.

Figure 417: Sample Basic Outline Showing Accounts and Time Tags



```
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)
```

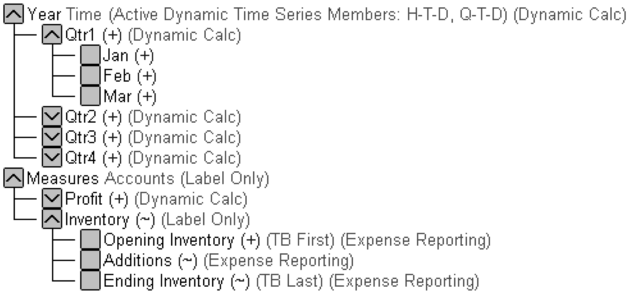
For information on tagging accounts and time dimensions, see Chapter 8, “Setting Dimension and Member Properties.”

Reporting the Last Value for Each Time Period

For an accounts dimension member, you can tell Essbase to move the last value for each time period up to the next level. To report the last value for each time period, set the member's time balance property as Last. (In the database outline, the TB Last tag is displayed.)

For example, in the Sample Basic database, the accounts member Ending Inventory consolidates the value for the last month in each quarter and uses that value for that month's parent. For example, the value for Qtr1 is the same as the value for Mar.

Figure 418: Sample Basic Outline Showing Last Tag



```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
    Qtr1 (+) (Dynamic Calc)
        Jan (+)
        Feb (+)
        Mar (+)
    Qtr2 (+) (Dynamic Calc)
    Qtr3 (+) (Dynamic Calc)
    Qtr4 (+) (Dynamic Calc)
Measures Accounts (Label Only)
    Profit (+) (Dynamic Calc)
    Inventory (~) (Label Only)
        Opening Inventory (+) (TB First) (Expense Reporting)
        Additions (~) (Expense Reporting)
        Ending Inventory (~) (TB Last) (Expense Reporting)
```

For information on tagging an accounts member as Last, see Chapter 8, "Setting Dimension and Member Properties."

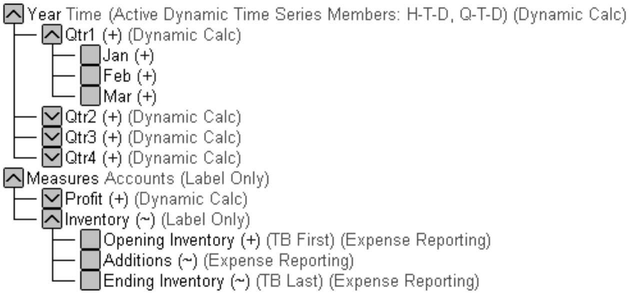
By default, Essbase does not skip `#MISSING` or zero (0) values when calculating a parent value. You can choose to skip these values. See "Skipping #MISSING and Zero Values" on page 803.

Reporting the First Value for Each Time Period

For an accounts dimension member, you can tell Essbase to move the first value for each time period up to the next level. To report the first value for each time period, set the member's time balance property as First. (The tag displays as TB First in the database outline.)

For example, in the Sample Basic database, the accounts member Opening Inventory consolidates the value of the first month in each quarter and uses that value for that month's parent. For example, the value for Qtr1 is the same as the value for Jan.

Figure 419: Sample Basic Outline Showing First Tag



```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)
    Qtr1 (+) (Dynamic Calc)
        Jan (+)
        Feb (+)
        Mar (+)
    Qtr2 (+) (Dynamic Calc)
    Qtr3 (+) (Dynamic Calc)
    Qtr4 (+) (Dynamic Calc)
Measures Accounts (Label Only)
    Profit (+) (Dynamic Calc)
    Inventory (~) (Label Only)
        Opening Inventory (+) (TB First) (Expense Reporting)
        Additions (~) (Expense Reporting)
        Ending Inventory (~) (TB Last) (Expense Reporting)
```

For information on tagging an accounts member as First, see Chapter 8, "Setting Dimension and Member Properties."

By default, Essbase does not skip `#MISSING` or zero (0) values when calculating a parent value. You can choose to skip these values. See "Skipping #MISSING and Zero Values" on page 803.

Reporting the Average Value for Each Time Period

For an accounts dimension member, you can tell Essbase to average values across time periods and consolidate the average up to the next level. For example, you can tell Essbase to average the values for Jan, Feb, and Mar and then use that value for the Qtr1 value. To report the average value for each time period, set the member's time balance property as Average.

For information on tagging an accounts member as Average, see Chapter 8, "Setting Dimension and Member Properties."

By default, Essbase does not skip #MISSING or zero (0) values when it calculates a parent value. Thus, when it calculates the average, Essbase aggregates the child values and divides by the number of children, regardless of whether the children have #MISSING or zero values. You can tell Essbase to skip #MISSING and zero values. See "Skipping #MISSING and Zero Values" on page 803.

Skipping #MISSING and Zero Values

You can tell Essbase how to treat #MISSING and zero (0) values when doing time balance calculations. A #MISSING value is a marker in Essbase that indicates that the data in this location does not exist, does not contain any meaningful value, or was never entered.

By default, Essbase does not skip #MISSING or 0 (zero) values when calculating a parent value.

You can override this default using the following tags:

To Perform This Task	Use this Tag
Skip #MISSING values when calculating a parent value	**Skip Missing** In Outline Editor, select the parent member from the accounts dimension and click the #MI button.
Skip 0 values when calculating a parent value	**Skip Zeros** In Outline Editor, select the parent member from the accounts dimension and click the 0 button.
Skip #MISSING values and 0 values when calculating a parent value	**Skip Missing and Skip Zeros** In Outline Editor, select the parent member from the accounts dimension and click the #MI, 0, X button.

For example, if you tag an accounts dimension member as Last and Skip Missing, then Essbase consolidates the last non-missing child to the parent. Consider the following example:

Accounts -> Time	Jan	Feb	Mar	Qtr1
Accounts Member (**Last, Skip Missing**)	60	70	#MI	70

Tagging an account as Average and Skip Missing may produce different results from tagging that account as Average and Skip None. A calculation performed with Average and Skip None will produce correct results because no data is skipped. But because grandparents with children are consolidated by summing the averages, results of a calculation on an account with Average and Skip Missing will be incorrect unless you use Dynamic Calc or Two Pass tags.

Considering the Effects of First, Last, and Average Tags

The following table shows how Essbase consolidates the time dimension based on the time balance (TB) First, Last, and Average tags on accounts dimension members.

Accounts -> Time	Jan	Feb	Mar	Qtr1	
Accounts Member1	11	12	13	36	Value of Jan+Feb+Mar
Accounts Member2 (**TB First**)	20	25	21	20	Value of Jan
Accounts Member3 (**TB Last**)	25	21	30	30	Value of Mar
Accounts Member4 (**TB Average**)	20	30	28	26	Average of Jan, Feb, Mar

Placing Formulas on Time and Accounts Dimensions

If you place a member formula on a time or accounts dimension, it may be overwritten by a time balance calculation.

Consider the following example from Sample Basic, in which Opening Inventory is tagged as First:

Measures -> Year	**Jan**	**Feb**	**Mar**	**Qtr1**
Opening Inventory: **First**	30000	28000	27000	30000

Because Opening Inventory is tagged as First, Essbase calculates Opening Inventory for Qtr1 by taking the Opening Inventory for Jan value. Any member formula that is placed on Qtr1 in the database outline is overwritten by this time balance calculation.

Calculating Period-to-Date Values

You can calculate period-to-date values for your data. For example, you can calculate the sales values for the current quarter up to the current month. If the current month is May, using a standard calendar quarter, the quarter total is the total of the values for April and May.

In Essbase, you can calculate period-to-date values in two ways:

- During a batch calculation, using the @PTD function
- Dynamically, when a user requests the values, using Dynamic Time Series members

This section explains how to use Dynamic Time Series members to dynamically calculate period-to-date values. Using Dynamic Time Series members is the most efficient method in almost all cases. For an example using the @PTD function to calculate period-to-date values, see Chapter 26, “Examples of Formulas.”

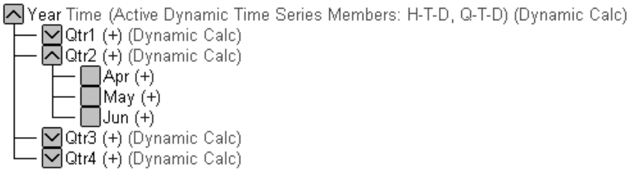
Using Dynamic Time Series Members

In order to calculate period-to-date values dynamically, you need to use a Dynamic Time Series member for a period on the dimension tagged as time. See "Specifying Accounts and Time Dimensions" on page 800.

You do not create the Dynamic Time Series member directly in the database outline. Instead, you enable a predefined Dynamic Time Series member and associate it with an appropriate generation number. This procedure creates a Dynamic Time Series member for you.

For example, if you want to calculate quarter-to-date values, you enable the Q-T-D member and associate it with the generation to which you want to apply the Dynamic Time Series member. In Sample Basic, this is generation number 2, which contains the Qtr1, Qtr2, Qtr3, and Qtr4 members. Essbase creates a Dynamic Time Series member called Q-T-D and associates it with generation 2. The Q-T-D member calculates monthly values up to the current month in the quarter. For more information, see "Enabling Dynamic Time Series Members" on page 808.

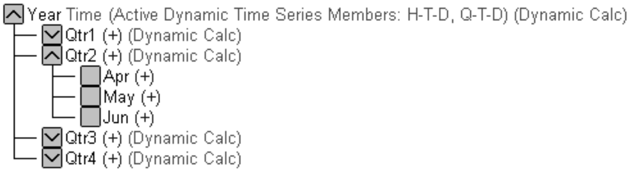
Figure 420: Sample Basic Outline Showing Time Dimension



```
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)
```

Dynamic Time Series members are not displayed as members in the database outline. Instead, Essbase lists the currently active Dynamic Time Series members in a comment on the time dimension. In the following outline, H-T-D (history-to-date) and Q-T-D (quarter-to-date) are active. H-T-D is associated with generation 1; Q-T-D is associated with generation 2.

Figure 421: Sample Basic Outline Showing Dynamic Time Series



```
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 provides eight predefined Dynamic Time Series members:

H-T-D	History-to-date
Y-T-D	Year-to-date
S-T-D	Season-to-date
P-T-D	Period-to-date
Q-T-D	Quarter-to-date
M-T-D	Month-to-date
W-T-D	Week-to-date
D-T-D	Day-to-date

These eight members provide up to eight levels of period-to-date reporting. How many members you use and which members you use depends on your data and your database outline.

For example, if your database contains hourly, daily, weekly, monthly, quarterly, and yearly data, you might want to report day-to date (D-T-D), week-to-date (W-T-D), month-to-date (M-T-D), quarter-to-date (Q-T-D), and year-to-date (Y-T-D) information.

If your database contains monthly data for the past 5 years, you might want to report year-to-date (Y-T-D) and history-to-date (H-T-D) information, up to a specific year.

If your database tracks data for seasonal time periods, you might want to report period-to-date (P-T-D) or season-to-date (S-T-D) information.

You can associate a Dynamic Time Series member with any generation in the time dimension except the highest generation number, irrespective of the data. For example, if you choose, you can use the P-T-D member to report quarter-to-date information. You cannot associate Dynamic Time Series members with level 0 members of the time dimension.

Enabling Dynamic Time Series Members

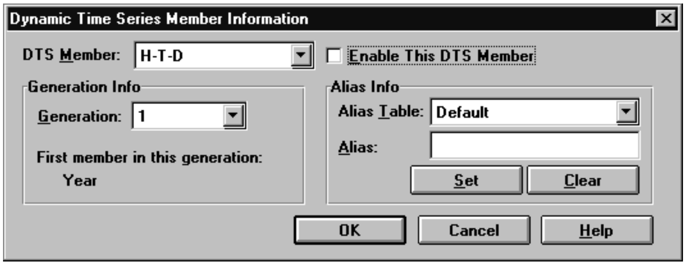
To use Dynamic Time Series members, you need to enable them. If required, you can specify aliases for Dynamic Time Series members. See "Specifying Alias Names for Dynamic Time Series Members" on page 811.

➤ To enable Dynamic Time Series members using Application Manager:

1. In Outline Editor, open the database.
2. Select Outline > Dynamic Time Series.

   Essbase displays the Dynamic Time Series Member Information dialog box.

   Figure 422: Dynamic Time Series Member Information Dialog Box for Sample Basic



Dynamic Time Series Member Information

DTS Member: H-T-D

□ Enable This DTS Member

Generation: 1

First member in this generation: Year

Alias Table: Default

Alias:

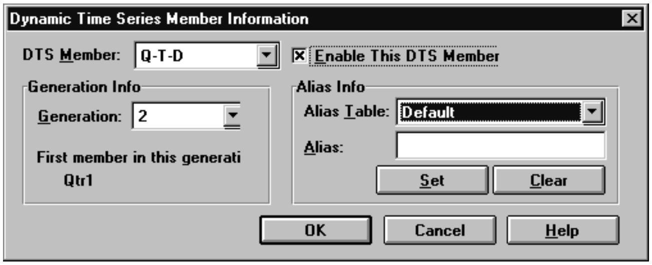
3. In the DTS Member list, select the required Dynamic Time Series member. For example, select Q-T-D (quarter-to-date).
4. In the Generation list, select the appropriate generation number. For example, in the Sample Basic database, select generation 2 to associate the Q-T-D member with that generation.

   Note: How many generations the Generation list displays depends on how many generations are in the time dimension. You cannot associate Dynamic Time Series members with the highest generation (level 0 members).

5. Check Enable This DTS Member.

   In Sample Basic, the DTS member may already be enabled by default.

   Figure 423: Dynamic Time Series Member Information Dialog Box Showing Q-T-D on Sample Basic



Dynamic Time Series Member Information

DTS Member: Q-T-D

☒ Enable This DTS Member

Generation Info

Generation: 2

First member in this generati
Qtr1

Alias Info

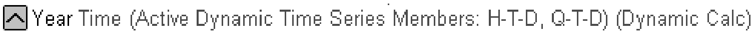
Alias Table: Default

Alias:

6. If desired, you can create an alias name for the DTS member. See “Specifying Alias Names for Dynamic Time Series Members” on page 811.
7. To enable the Dynamic Time Series member, click OK.

   In the database outline, Essbase adds a comment to the dimension tagged as time. Figure 424 shows the Sample Basic database with H-T-D and Q-T-D defined.

   Figure 424: Sample Basic Outline Showing Dynamic Time Series Members



Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D) (Dynamic Calc)

Disabling Dynamic Time Series Members

To disable a Dynamic Time Series member, you tell Essbase not to use the predefined member.

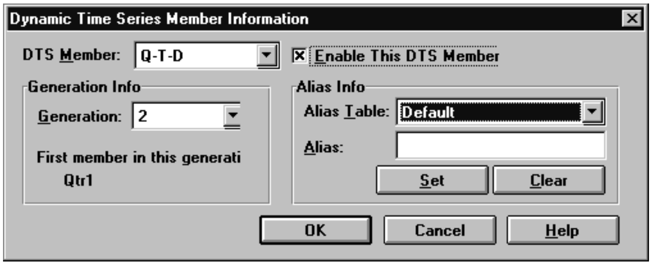
➤ To disable a Dynamic Time Series member using Application Manager:

1. In Outline Editor, open the database.
2. Select Outline > Dynamic Time Series.

   Essbase displays the Dynamic Time Series Member Information dialog box.

3. In the DTS Member list, select the Dynamic Time Series member you want to disable.

Figure 425: Dynamic Time Series Member Information Dialog Box Showing Enabled Q-T-D



Dynamic Time Series Member Information

DTS Member: Q-T-D

☒ Enable This DTS Member

Generation Info

Generation: 2

First member in this generati

Qtr1

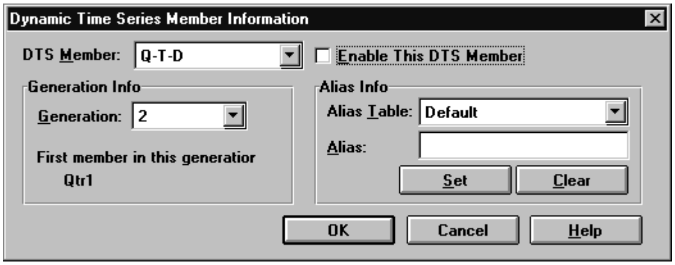
Alias Info

Alias Table: Default

Alias:

4. Clear Enable This DTS Member.

Figure 426: Dynamic Time Series Member Information Dialog Box Showing Q-T-D Disabled



Dynamic Time Series Member Information

DTS Member: Q-T-D

☐ Enable This DTS Member

Generation Info

Generation: 2

First member in this generatior

Qtr1

Alias Info

Alias Table: Default

Alias:

5. Click OK.

Specifying Alias Names for Dynamic Time Series Members

You can specify alias names for predefined Dynamic Time Series members. You can then use the alias names to retrieve the Dynamic Time Series members in Spreadsheet Add-in or in a report.

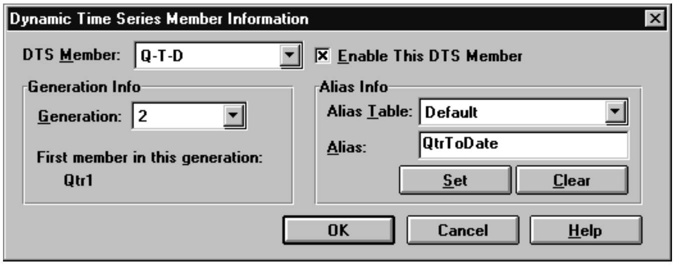
You can create up to eight alias names for each Dynamic Time Series member. Essbase saves each alias name in the Dynamic Time Series alias table that you specify.

➤ To specify an alias name for a Dynamic Time Series member:

1. In the Dynamic Time Series Member Information dialog box, enable the DTS member. See "Enabling Dynamic Time Series Members" on page 808.
2. In the Alias Table list, select the alias table in which you want to save the alias name.
3. In the Alias text box, type the alias name for the DTS member.

   For example, type **`QtrToDate`**.

   Figure 427: Dynamic Time Series Member Information Dialog Box Showing Q-T-D Alias

   DTS Member: Q-T-D

   ☒ Enable This DTS Member

   Generation: 2

   First member in this generation: Qtr1

   Alias Table: Default

   Alias: QtrToDate

4. Click Set.
5. To save the alias name to the chosen alias table, click OK.

For more information on specifying and displaying alias names, see Chapter 7, "Creating and Changing Database Outlines."

Applying Predefined Generation Names to Dynamic Time Series Members

When you enable a Dynamic Time Series member and associate it with a generation number, Essbase creates a predefined generation name for that generation number. These generation names display in the Generation and Level Names dialog box. To open this dialog box, select Outline > Gen/Level Names in Application Manager Outline Editor. For more information on creating generation names, see Chapter 7, “Creating and Changing Database Outlines.”

This table shows the Dynamic Time Series members and their corresponding generation names:

Member	**Generation Name**	**Member**	**Generation**
H-T-D	History	Q-T-D	Quarter
Y-T-D	Year	M-T-D	Month
S-T-D	Season	W-T-D	Week
P-T-D	Period	D-T-D	Day

These member and generation names are reserved for use by Essbase. If you use one of these generation names to create a generation name on the time dimension, Essbase automatically creates and enables the corresponding Dynamic Time Series member for you.

For example, in Sample Basic, you can create a generation name called Quarter for generation number 2. Quarter contains quarterly data in the members Qtr1, Qtr2, and so on. When you create the generation name Quarter, Essbase creates and enables a Dynamic Time Series member called Q-T-D.

Retrieving Period-to-Date Values

When you retrieve a Dynamic Time Series member, you need to tell Essbase the time period up to which you want to calculate the period-to-date value. This time period is known as the *latest time period* and must be a level 0 member on the time dimension.

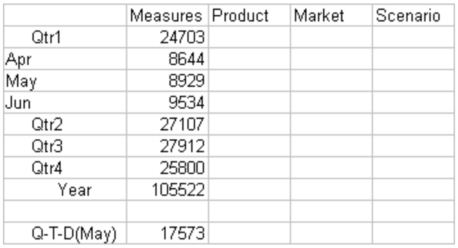
➤ To specify the latest time period:

- For a specific member, in Spreadsheet Add-in, specify the latest period member name. Place that name after the Dynamic Time Series member or alias name. For example, Q-T-D(May) returns the quarter-to-date value by adding values for April and May.
- For a retrieval, do one of the following:
  - Use the <LATEST command in Report Writer. For more information, see Part VII, “Retrieving Data.”
  - Specify the Latest Time Series option in the Spreadsheet Add-in Essbase Options dialog box. For more information, see the *Essbase Spreadsheet Add-in User’s Guide.*

The member-specific setting—for example, Q-T-D(May)—takes precedence over the <LATEST or Latest Time Series option setting.

The following example shows Sample Basic data. Q-T-D(May) displays the period-to-date value for May that is obtained by adding the values for Apr and May (8644 + 8929 = 17573):

Figure 428: Spreadsheet Add-in Showing Period-To-Date Value for May



	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			

Using Dynamic Time Series Members in Partitions

If Dynamic Time Series members are part of the shared area between databases, you need to define the Dynamic Time Series members in both databases, just as you would for regular members. For example, if your partition definition includes Qtr1, Qtr2, Qtr3, Qtr4, and the Dynamic Time Series member Q-T-D, then you need to define the Q-T-D member in both the source database and the target database.

If a Dynamic Time Series member is *not* part of the shared area between databases, Essbase gets the data for that member from the source database. You do not need to define the Dynamic Time Series member in both databases. However, this configuration is generally less efficient than including the Dynamic Time Series member in the partition definition.

For more information on partitioning, see Chapter 14, “Building and Maintaining Partitions.”

Chapter 30

Developing Calculation Scripts

This chapter explains how to develop calculation scripts and how to use them to control how Essbase calculates a database. It provides some examples of calculation scripts, which you may want to adapt for your own use. For more examples, see Chapter 31, “Examples of Calculation Scripts.”

This chapter includes the following sections:

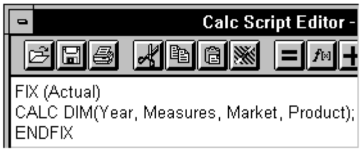
For information on developing formulas, see Chapter 25, “Developing Formulas.”

Using a Calculation Script

A calculation script contains a series of calculation commands, equations, and formulas. You use a calculation script to define calculations other than the calculations that are defined by the database outline.

For example, the following calculation script calculates the Actual values in the Sample Basic database.

Figure 429: Calculation Script Editor



```
FIX (Actual)
CALC DIM(Year, Measures, Market, Product);
ENDFIX
```

You can use a calculation script to specify exactly how you want Essbase to calculate a database. For example, you can calculate part of a database or copy data values between members. You can design and run custom database calculations quickly by separating calculation logic from the database outline.

For most database calculations, a default calculation provides the required results. However, in certain cases, you may need to write a calculation script to control how Essbase calculates a database.

For example, you need to write a calculation script if you want to do any of the following:

- Use the FIX command to calculate a subset of a database. For more information, see "Calculating a Subset of a Database" on page 869 and the *Technical Reference* in the `docs` directory.
- Change the calculation order of the dense and sparse dimensions in a database.
- Perform a complex calculation in a specific order or perform a calculation that requires multiple iterations through the data. For example, some two-pass calculations require a calculation script. For more information, see Chapter 51, "Optimizing Calculations."
- Perform any two-pass calculation on a dimension without an accounts tag. For more information, see Chapter 51, "Optimizing Calculations."
- Perform a currency conversion. For more information, see Chapter 12, "Designing and Building Currency Conversion Applications."

- Calculate member formulas that differ from formulas in the database outline. Formulas in a calculation script override formulas in the database outline.
- Use an API interface to create a custom calculation dynamically.
- Use logic in a calculation; for example, if you want to use the IF … ELSE … ENDIF or the LOOP … ENDLOOP commands. For more information, see "Controlling the Flow of Calculations" on page 825 and the *Technical Reference* in the `docs` directory.
- Clear or copy data from specific members. For more information, see "Controlling the Flow of Calculations" on page 825, "Copying Data" on page 868, and the *Technical Reference* in the `docs` directory.
- Define temporary variables for use in a database calculation. For more information, see "Declaring Data Variables" on page 825 and the *Technical Reference* in the `docs` directory.
- Force a recalculation of data blocks after you have changed a formula or an accounts property on the database outline. For more information, see Chapter 52, "Optimizing with Intelligent Calculation."
- Control how Essbase uses the Intelligent Calculation feature when calculating a database. For more information, see Chapter 52, "Optimizing with Intelligent Calculation."

Creating a Calculation Script

This section provides a step-by-step example of creating and saving a calculation script.

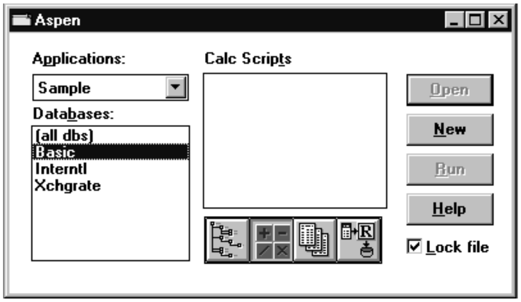
For detailed information on creating formulas and obtaining the required calculation results, consider all the information in Part VI, "Calculating Data."

This example is based on the Sample Basic database, which is supplied with the Essbase server installation. This example increases all budget values by 5%. The example assumes that you have Essbase Spreadsheet Add-in installed on your computer.

➤ To create the example calculation script using Application Manager:

1. Start Application Manager and connect to the OLAP Server.
2. Select the **Sample** application and the **Basic** database, and click the **Calc Scripts** button, .

   Figure 430: Application Desktop Window

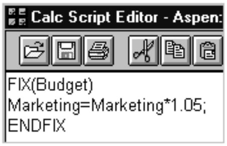
   If you do not have Sample Basic installed, contact the Essbase administrator.

   If another user has Sample Basic open and locked, you can clear “Lock file” in the bottom-right corner of the application desktop window. However, if you clear “Lock file”, you cannot save your work.

3. Click **New** to open Calc Script Editor.
4. Type the following calculation script. This script increases the expense values of Budget -> Marketing by 5%.

```
FIX(Budget)
Marketing = Marketing * 1.05;
ENDFIX
```

   Figure 431: Simple Calculation Script

```
FIX(Budget)
Marketing=Marketing*1.05;
ENDFIX
```

   For more information on the FIX command, see “Using the FIX Command” on page 870 and the *Technical Reference* in the `docs` directory.

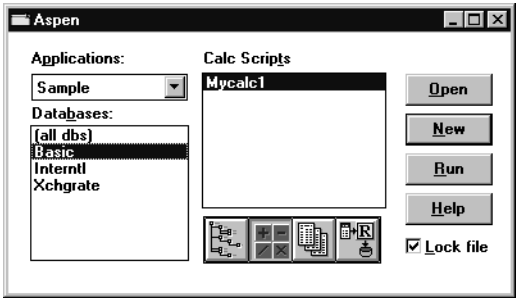
5. Click the **Check Syntax** button, , to verify that the syntax of the formula you entered is correct.

   The message “No errors” should be displayed at the bottom of Calc Script Editor.

6. Click the **Save** button, , to save the calculation script.
7. Type **Mycalc1** for the calculation script object name, and save the calculation script on the server (the default).
8. Close the **Calc Script Editor** window.

In Figure 432 Mycalc1 is displayed in the list of calculation scripts for Sample Basic.

Figure 432: Application Desktop Window Showing a Calculation Script



Aspen

Applications: Sample

Databases: (all dbs), Basic, Interntl, Xchgrate

Calc Scripts: Mycalc1

☑ Lock file

Calculating Sample Basic Data

You are now ready to calculate the Sample Basic database, increasing the Budget values by 5%. However, first take a look at the Sample Basic Budget values before running the calculation.

➤ To review the Budget values of Sample Basic:

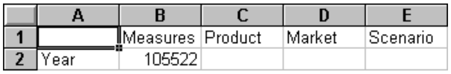
1. Open Essbase Spreadsheet Add-in, and select **Essbase** > **Connect** to connect to Sample Basic.

   This example assumes that you have not changed the default Essbase Spreadsheet Add-in Retrieval Options. For more information, see the *Essbase Spreadsheet Add-in User’s Guide*.

2. Select **Essbase** > **Retrieve**.

   Essbase displays the data value from the top level of each dimension.

   Figure 433: Essbase Spreadsheet Add-in Showing Initial Data



	A	B	C	D	E
1		Measures	Product	Market	Scenario
2	Year	105522			

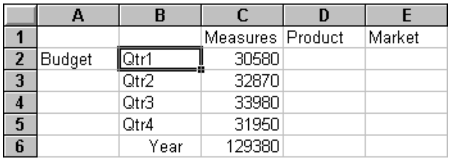
3. Double-click **Scenario** to display its members.
4. Select **Budget** and select **Essbase** > **Keep Only**.

   Essbase displays the Budget values.

5. Double-click **Year**.

   Essbase displays the Budget values for each quarter in the year.

   Figure 434: Essbase Spreadsheet Add-in Showing Retrieved Budget Data for Each Quarter

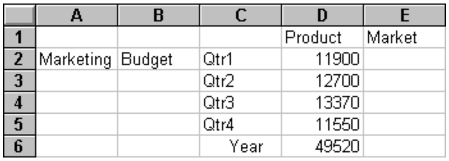


	A	B	C	D	E
1			Measures	Product	Market
2	Budget	Qtr1	30580		
3		Qtr2	32870		
4		Qtr3	33980		
5		Qtr4	31950		
6		Year	129380		

6. Double-click **Measures**, then **Profit**, and then **Total Expenses** to display the Marketing member.
7. Select **Marketing** and select **Essbase** > **Keep Only**.

   Essbase displays the Budget -> Marketing values. These are the values that will increase by 5%.

   Figure 435: Essbase Spreadsheet Add-in Showing Retrieved Data for Budget -> Marketing



	A	B	C	D	E
1				Product	Market
2	Marketing	Budget	Qtr1	11900	
3			Qtr2	12700	
4			Qtr3	13370	
5			Qtr4	11550	
6			Year	49520	

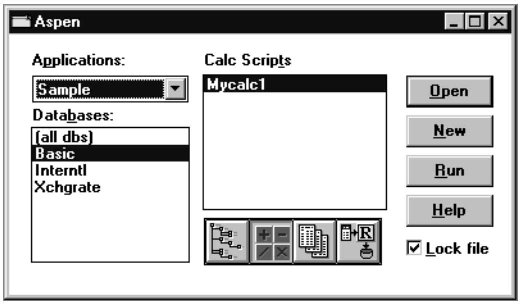
Running a Calculation Script

Now you are ready to run the Mycalc1 calculation script, which increases the Budget -> Marketing values by 5%.

➤ To run the Mycalc1 calculation script:

1. Minimize but do not close the **Essbase Spreadsheet Add-in** window.
2. In Application Manager, connect to OLAP Server, if you are not already connected.
3. Select the **Sample** application and the **Basic** database, and click the **Calc Scripts** button, .

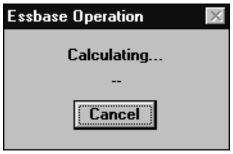
Figure 436: Application Desktop Window Showing calculation Script



4. Select **Mycalc1** and click **Run**.
5. When Essbase prompts you to select a database, ensure that **Sample Basic** is selected in the **Select Database** dialog box. Click **OK**.

   Essbase calculates the database.

Figure 437: Calculating Message Box



Calculating...

--

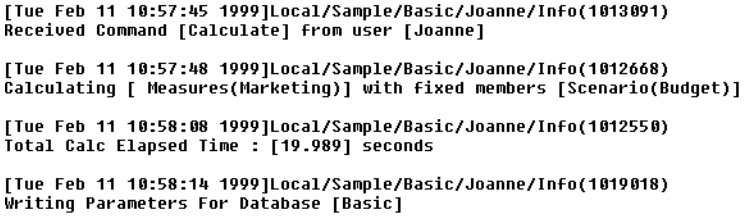
Checking a Calculation

After Essbase finishes a calculation, you can check the dimensions calculated and the calculation time in the application log.

➤ To review the application log:

1. In Application Manager, select the **Application** > **View Event Log** menu command.
2. In the **View Log File** dialog box, select **Date** to view the entries for the current date.
3. When Essbase displays the application log, scroll to the end of the file to see the entries for a calculation. The entries will be similar to the ones shown in Figure 438.

Figure 438: Application Log Showing Calculation Messages



```
[Tue Feb 11 10:57:45 1999]Local/Sample/Basic/Joanne/Info(1013091)
Received Command [Calculate] from user [Joanne]

[Tue Feb 11 10:57:48 1999]Local/Sample/Basic/Joanne/Info(1012668)
Calculating [ Measures(Marketing)] with fixed members [Scenario(Budget)]

[Tue Feb 11 10:58:08 1999]Local/Sample/Basic/Joanne/Info(1012550)
Total Calc Elapsed Time : [19.989] seconds

[Tue Feb 11 10:58:14 1999]Local/Sample/Basic/Joanne/Info(1019018)
Writing Parameters For Database [Basic]
```

From the entries, you can see that Essbase calculated data values for the Marketing member on the Measures dimension, fixing on the Budget values. Essbase calculated the database in 19.989 seconds.

Note: Check the “fixed members [ ]” part of the message to ensure that all members you fixed on were actually included in the calculation.

These entries are an example of the default level of messages that Essbase provides. If required, you can display more detailed calculation messages in the application log by using the SET MSG command. For more information, see the *Technical Reference* in the `docs` directory.

4. Close the application log viewer window.

Tip: You can also review the application log using the Log Analyzer window in Administration Services. For more information, see *Essbase Administration Services Online Help*.

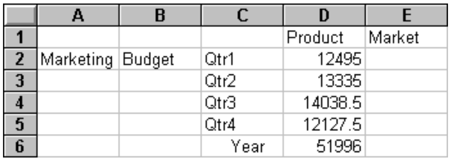
➤ To view newly calculated data:

1. Maximize the **Essbase Spreadsheet Add-in** window.

   The pre-calculation data values should still be displayed. If the pre-calculation data values are not displayed, repeat the steps in “Calculating Sample Basic Data” on page 819.

2. Select **Essbase** > **Retrieve** to display the new data values as shown in Figure 439.

Figure 439: Essbase Spreadsheet Add-in Showing Retrieved New Data



	A	B	C	D	E
1				Product	Market
2	Marketing	Budget	Qtr1	12495	
3			Qtr2	13335	
4			Qtr3	14038.5	
5			Qtr4	12127.5	
6			Year	51996	

As you can see, the data values have increased by 5%. The calculation is successful. If required, you can reload the default data into Sample Basic. See Part V, “Building Dimensions and Loading Data.”

Building a Calculation Script in the Calc Script Editor

You use either Calc Script Editor in Application Manager or Calculation Script Editor in Administration Services to build a calculation script. You can type the calculation script directly into the text area of Calc Script Editor, or you can use the user interface features of Calc Script Editor to build the calculation script.

Calculation scripts are ASCII text. If desired, you can create a calculation script in the text editor of your choice and paste it into Calc Script Editor.

Essbase provides a flexible set of commands that you can use to control how a database is calculated. You can construct calculation scripts from commands and formulas. Several types of commands, such as these, are discussed in the following sections:

- Computation
- Control flow

- Data declaration
- Global settings

Implementing Outline Calculations

You can use the following calculation commands to perform a database calculation that is based on the structure and formulas in the database outline.

Note: For a complete list of calculation commands and syntax, see the *Technical Reference* in the `docs` directory.

Calculation	Command
The entire database, based on the outline	CALC ALL
A specified dimension or dimensions	CALC DIM
All members tagged as two-pass on the dimension tagged as accounts	CALC TWOPASS
The formula applied to a member in the database outline, where *membername* is the name of the member to which the formula is applied	*membername*
All members tagged as Average on the dimension tagged as accounts (see Chapter 29, “Calculating Time Series Data”)	CALC AVERAGE
All members tagged as First on the dimension tagged as accounts (see Chapter 29, “Calculating Time Series Data”)	CALC FIRST
All members tagged as Last on the dimension tagged as accounts (see Chapter 29, “Calculating Time Series Data”)	CALC LAST
Currency conversions (see Chapter 12, “Designing and Building Currency Conversion Applications”)	CCONV

Controlling the Flow of Calculations

You can use the following commands to manipulate the flow of calculations. For detailed information on these commands, see the *Technical Reference* in the `docs` directory.

Calculation	Commands
Calculate a subset of a database	FIX … ENDFIX
Specify the number of times that commands are iterated	LOOP … ENDLOOP

You can also use the IF and ENDIF commands to specify conditional calculations. See "Controlling the Flow of Calculations" on page 825.

Note: You cannot branch from one calculation script to another calculation script.

Declaring Data Variables

You can use the following commands to declare temporary variables and, if required, to set their initial values. *Temporary variables* store the results of intermediate calculations.

You can also use substitution variables in a calculation script. See "Using Substitution Variables" on page 866.

Calculation	Command
Declare one-dimensional array variables	ARRAY
Declare a temporary variable that contains a single value	VAR

For detailed information on the these commands, see the *Technical Reference* in the `docs` directory.

Values stored in temporary variables exist only while the calculation script is running. You cannot report on the values of temporary variables.

Variable and array names are character strings that contain any of the following characters:

- Alphabetic letters: a through z
- Numerals: 0 through 9
- Special characters: $ (dollar sign), # (pound sign), and _ (underscore)

Typically, arrays are used to store variables as part of a member formula. The size of the array variable is determined by the number of members in the corresponding dimension. For example, if the Scenario dimension has four members, the following command creates an array called `Discount` with four entries. You can use more than one array at a time.

```
ARRAY Discount[Scenario];
```

Specifying Global Settings for a Database Calculation

You can use the following commands to define calculation behavior.

Note: For a complete list of commands, see the *Technical Reference* in the `docs` directory.

Calculation	Command
To specify how Essbase treats `#MISSING` values during a calculation	SET AGGMISSG
To adjust the default calculator cache size	SET CACHE
To optimize the calculation of large, flat database outlines (see "Performance for Database Outlines with Two or More Flat Dimensions" on page 51-1373)	SET CALCHASHTBL
To enable parallel calculation (see "Using Parallel Calculation" on page 51-1380)	SET CALCPARALLEL
To increase the number of dimensions used to identify tasks for parallel calculation (see "Using Parallel Calculation" on page 51-1380)	SET CALCTASKDIMS

Calculation	Command
To optimize the calculation of sparse dimension formulas in large database outlines (see Chapter 51, “Optimizing Calculations”)	SET FRMLBOTTOMUP

To display messages to trace a calculation.	SET MSG
	SET NOTICE

To turn on and turn off Intelligent Calculation (see Chapter 52, “Optimizing with Intelligent Calculation”)	SET UPDATECALC
To control how Essbase marks data blocks for the purpose of Intelligent Calculation (see Chapter 52, “Optimizing with Intelligent Calculation”)	SET CLEARUPDATESTATUS
To specify the maximum number of blocks that Essbase can lock concurrently when calculating a sparse member formula	SET LOCKBLOCK
For currency conversions, to restrict aggregations to parents that have the same defined currency (see Chapter 12, “Designing and Building Currency Conversion Applications”)	SET UPTOLOCAL

SET commands in a calculation script are procedural. A SET command in a calculation script stays in effect until the next occurrence of the same SET command.

For example, consider the following calculation script:

```
SET MSG DETAIL;
CALC DIM(Year);

SET MSG SUMMARY;
CALC DIM(Measures);
```

Essbase displays messages at the detail level when calculating the Year dimension. However, when calculating the Measures dimension, Essbase displays messages at the summary level.

Now, consider this calculation script:

```
SET AGGMISSG ON;
Qtr1;

SET AGGMISSG OFF;
East;
```

Essbase calculates member combinations for Qtr1 with SET AGGMISSG (aggregate missing values) turned on. Essbase then does a second calculation pass through the database and calculates member combinations for East with SET AGGMISSG turned off. For more information on the setting for aggregating missing values, see the SET AGGMISSG command in the *Technical Reference* in the `docs` directory. For more information on calculation passes, see Chapter 51, “Optimizing Calculations.”

Adding Comments

You can include comments to annotate calculation scripts. Essbase ignores these comments when it runs the calculation script.

To include a comment, start the comment with /* and end the comment with */. For example, consider the following comment:

```
/*   This is a calculation script comment
     that spans two lines.*/
```

Composing Calculation Script Syntax

When you create a calculation script, you need to apply the following rules:

- End each formula or calculation script command with a semicolon (;), as shown in these examples:

  Example 1

  ```
  CALC DIM(Product, Measures);
  ```

  Example 2

  ```
  DATACOPY Plan TO Revised_Plan;
  ```

  Example 3

  ```
  "Market Share" = Sales % Sales -> Market;
  ```

  Example 4

```
IF
  (Sales <> #MISSING) Commission = Sales * .9;
ELSE
  Commission = #MISSING;
ENDIF;
```

You do not need to end the following commands with semicolons: IF, ELSE, ELSEIF, FIX, ENDFIX, LOOP, and ENDLOOP.

- Enclose the member name in double quotation marks (" ") if the member name meets any of the following conditions:
  - Contains spaces; for example,

    ```
    "Opening Inventory" = "Ending Inventory" - Sales +
    Additions;
    ```

  - Is the same as an operator or function name.
  - Includes any non-alphanumeric character; for example, hyphens (-), asterisks (*), and slashes (/).
  - Is all numeric or starts with a numeral; for example, “100” or “10Prod”.
  - Begins with an ampersand (&). The leading ampersand (&) is reserved for substitution variables. If a member name begins with &, enclose it in quotation marks. Do not enclose substitution variables in quotation marks in a calculation script.
  - Contains a dot (.); for example, 1999.Jan or .100.

  For a complete list of member names that must be enclosed in quotation marks, see “Rules for Naming Dimensions and Members” on page 179.

- If you are using an IF statement or an interdependent formula, enclose the formula in parentheses and associate it with the specified member. For example, the following formula is associated with the Commission member in the database outline:

  ```
  Commission
  (IF(Sales < 100)
     Commission = 0;
  ENDIF;)
  ```

- End each IF statement in a formula with an ENDIF statement. For example, the following formula contains a simple IF...ENDIF statement:

  ```
  Profit
  (IF (Sales > 100)
      Commission = Sales * .1;
  ENDIF;)
  ```

- If you are using an IF statement that is nested within another IF statement, end each IF with an ENDIF. For example, consider the following example:

  ```
  "Opening Inventory"
  (IF (@ISMBR(Budget))
      IF (@ISMBR(Jan))
      "Opening Inventory" = Jan;
      ELSE
      "Opening Inventory" = @PRIOR("Ending Inventory");
      ENDIF;
  ENDIF;)
  ```

- You do not need to end ELSE or ELSEIF statements with ENDIF statements. For example, consider the following:

  ```
  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;)
  ```

  Note: If you use ELSE IF (with a space in between) rather than ELSEIF (one word) in a formula, you must supply an ENDIF for the IF statement.

- Although ending ENDIF statements with a semicolon (;) is not required, it is good practice to follow each ENDIF statement in a formula with a semicolon.

- End each FIX statement with an ENDFIX statement. For example:

  ```
  FIX(Budget,@DESCENDANTS(East))
  CALC DIM(Year, Measures, Product);
  ENDFIX
  ```

  The FIX and ENDFIX statements do not need to be followed by a semicolon (;).

When you write a calculation script, you can use the Calc Script Editor syntax checker to check the syntax. For more information, see “Checking Syntax” on page 863.

Note: For detailed information on calculation script syntax, see the *Technical Reference* in the `docs` directory.

Opening Calc Script Editor

Use Calc Script Editor in Application Manager or Calc Script Editor in Administration Services to create a new calculation script or open an existing calculation script.

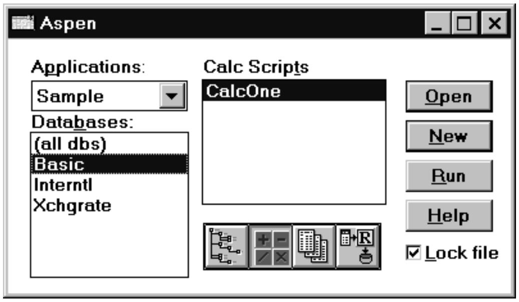
For information on opening an existing calculation script, see “Changing a Calculation Script” on page 833.

➤ To open Calc Script Editor using Application Manager:

1. Open Application Manager and connect to the OLAP Server.
2. In the application desktop server window, select the desired application and database.
3. Click the **Calculation Scripts** button, .

   Essbase displays a list of all the calculation scripts associated with the application and database that you selected.

Figure 440: Application Desktop Window



Applications: Sample

Databases: Basic

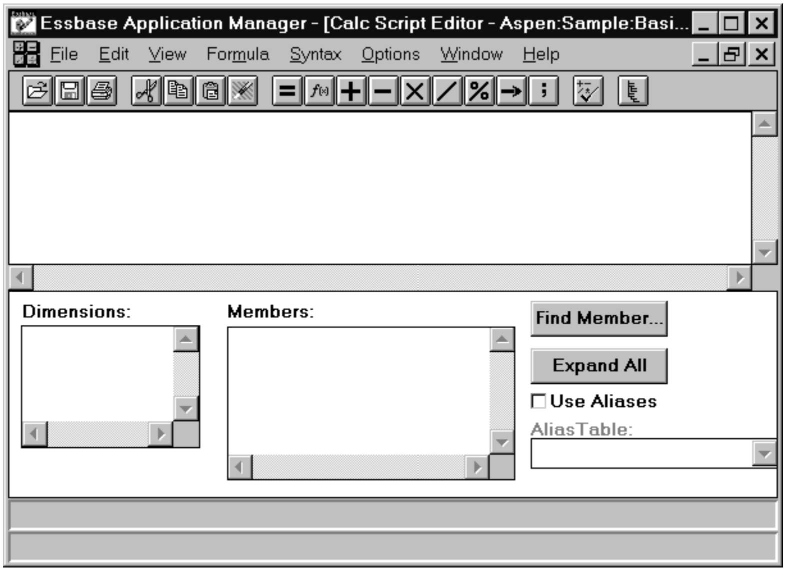
Calc Scripts: CalcOne

4. To create a new calculation script, click **New**.

   To open an existing calculation script, select it in the **Calc Scripts** list and click **Open**.

   Essbase opens Calc Script Editor, shown in Figure 441.

   Figure 441: Calc Script Editor



Adding a Calculation Script

You can use Application Manager to add a new calculation script.

➤ To add a calculation script:

1. Open Application Manager and connect to the OLAP Server.
2. In the application desktop server window, select the application and database with which you want to associate the new calculation script.
3. Click the **Calc Scripts** button, ⊞, and then click **New**.

   Alternatively, from the Application Manager menu, select **File** > **New** > **Calc Scrip**t.

Essbase opens Calc Script Editor. You can now build a calculation script. Essbase prompts you to name a calculation script when you save it. See “Saving a Calculation Script” on page 837.

Changing a Calculation Script

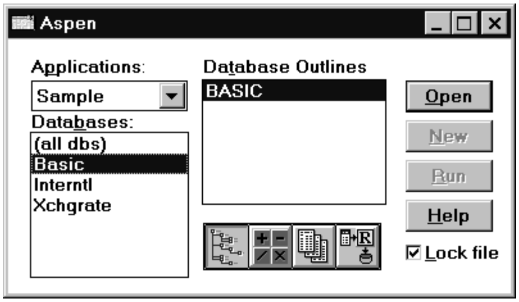
To change a calculation script, open it in Calc Script Editor. How you do that depends on where the calculation script is stored.

➤ To open a calculation script that is on the current server:

1. Open Application Manager and connect to the OLAP Server.
2. In the application desktop server window, select the application and database that contains the calculation script.

   The following example shows the application desktop server window for an Essbase server called **Aspen**. The **Sample Basic** database is selected as shown in Figure 442.

   Figure 442: Application Desktop Server Window



Applications: Sample

Databases: Basic

Database Outlines: BASIC

☑ Lock file

3. Click the **Calc Scripts** button, +−∕×.

   Essbase displays the calculation script files (`.CSC` files) stored in the `\`*`ARBORPATH`*`\`app`\`*`appname`*`\`*`dbname`* directory on the server computer, where *`ARBORPATH`* is the directory in which you installed Essbase and *`appname`* and *`dbname`* are the current application and database.

   For example, assuming that the Essbase install directory is `c:\essbase`, if you select Sample Basic, Essbase displays the `.CSC` files in the `c:\essbase\app\sample\basic` directory.

4. In the **Calc Scripts** list, select the calculation script you want to modify.
5. Click **Open**.

   Essbase opens Calc Script Editor.

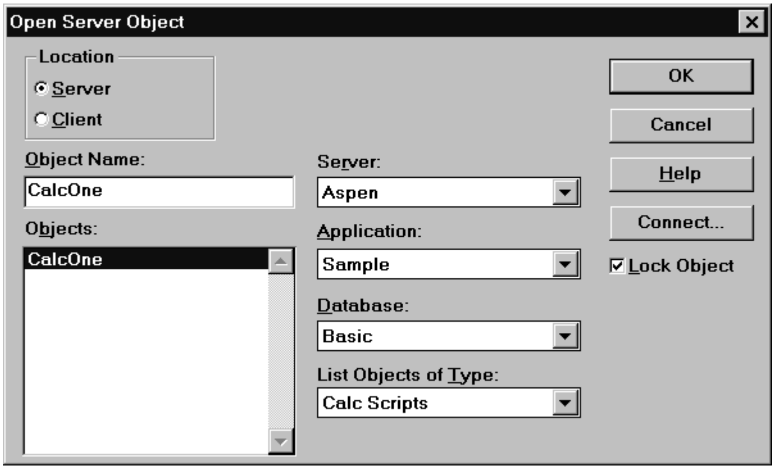
You can now edit the calculation script.

➤ To open a calculation script that is on a different server:

1. Open Application Manager and connect to the OLAP Server.
2. Ensure that the focus is on the application desktop server window.
3. From the Application Manager menu, select **File** > **Open**.

   Essbase displays the **Open Server Object** dialog box as shown in Figure 443.

   Figure 443: Open Server Object Dialog Box



Open Server Object

Location: ● Server ○ Client

Object Name: CalcOne

Objects: CalcOne

Server: Aspen

Application: Sample

Database: Basic

List Objects of Type: Calc Scripts

☑ Lock Object

4. Click **Connect** to connect to the other server, and click **OK**.
5. In the **Open Server Object** dialog box, select the application and database that contain the calculation script.
6. In the **Objects** list, select the required calculation script, and click **OK**.

   Essbase displays the calculation script in Calc Script Editor.

➤ To open a calculation script that is on a client computer:

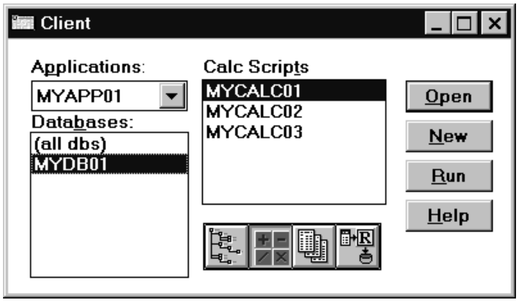
1. Open Application Manager and connect to the OLAP Server.
2. In the application desktop client window, select the application and database that contains the calculation script.
3. Click the **Calc Scripts** button, .

   Essbase displays the calculation script files (`.CSC` files) stored in the `\`*`ARBORPATH`*\CLIENT`\`*`appname`*`\`*`dbname`* directory on your client computer, where *ARBORPATH* is the directory in which you installed Essbase, and *appname* and *dbname* are the current application and database on your client computer.

   For example, assuming that the Essbase install directory is `C:\ESSBASE`, if you have an application called MYAPP01 and a database called MYDB01 on your client computer, Essbase displays the `.CSC` files in the `C:\ESSBASE\CLIENT\MYAPP01\MYDB01` directory on your client computer.

   In the example shown in Figure 444, there are three calculation scripts already created for the MYDB01 database.

   Figure 444: Application Desktop Client Window



Applications: MYAPP01

Databases: (all dbs), MYDB01

Calc Scripts: MYCALC01, MYCALC02, MYCALC03

4. In the **Calc Scripts** list, select the required calculation script.
5. Click **Open**.

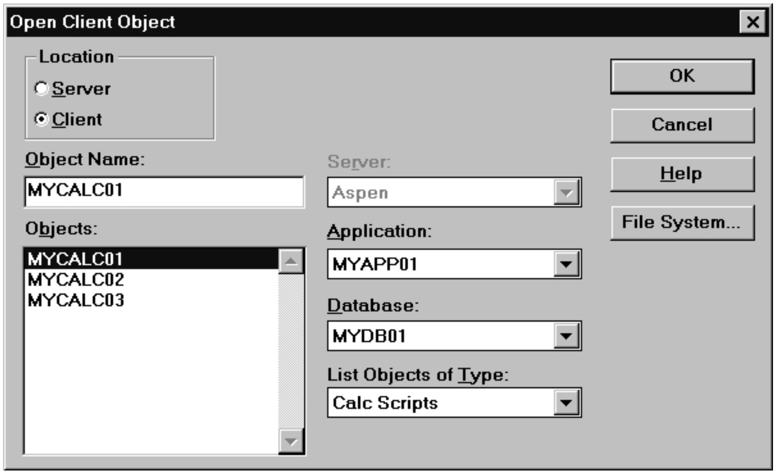
   Essbase opens **Calc Script Editor**. You can now edit the calculation script.

➤ To open a calculation script that is on a client computer but is not saved as a Essbase object:

1. From the Application Manager menu, select **File** > **Open**.

   Essbase displays the **Open Client Object** dialog box as shown in Figure 445.

   Figure 445: Open Client Object Dialog Box



Location
○ Server
● Client

Object Name: MYCALC01

Objects:
MYCALC01
MYCALC02
MYCALC03

Server: Aspen

Application: MYAPP01

Database: MYDB01

List Objects of Type: Calc Scripts

2. Click **File System** to connect to the other server.

   Essbase displays the **Open Client File** dialog box.

3. Select the file that contains the required calculation script, and click **OK**.

Essbase displays the calculation script in Calc Script Editor.

Saving a Calculation Script

You can save a calculation script as either of the following:

- An object on the Essbase server or on a client computer. You can associate the calculation script object with either of the following:
  - An application and all the databases within the application, which means that you can run the calculation script on any of the databases in the application
  - A database, which means that you can run the calculation script on the database
- A file on your client computer

If you want other users to have access to the calculation script, you need to save it on the Essbase server. If you save a calculation script on your client computer, other users do not have access to the calculation script. While you are developing a calculation script, you may want to save it on your client computer. Then move the completed script to the Essbase server.

When you save a calculation script from Calc Script Editor, by default Essbase associates it with the current application and database.

Calc scripts created using Application Manager are given a `.CSC` extension by default. If you run a calculation script from Application Manager or from Essbase Spreadsheet Add-in, it must have a `.CSC` extension. However, a calculation script is an ASCII file, and you can use MaxL or ESSCMD to run any ASCII file as a calculation script.

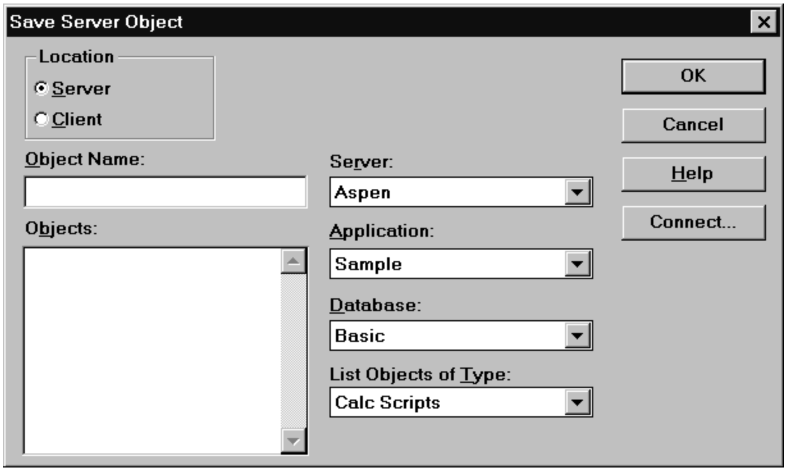
A calculation script can also be a string defined in memory. You can access this string via the API on the Essbase client or Essbase server. Thus, from dialog boxes, you can dynamically create a calculation script that is based on user selections.

➤ To save a calculation script as an object on the Essbase OLAP Server:

1. Open Application Manager and connect to the OLAP Server.
2. Select an application and database in the application desktop window.
3. Click the **Calc Script Editor** button and select an existing calculation script, or create a new calculation script, and display it in the Calc Script Editor.
4. In the **Calc Script Editor**, click the **Save** button, 💾.

   Essbase displays the **Save Server Object** dialog box as shown in Figure 446.

   Figure 446: Save Server Object Dialog Box

   Location: ● Server ○ Client

   Object Name:

   Server: Aspen

   Application: Sample

   Database: Basic

   List Objects of Type: Calc Scripts

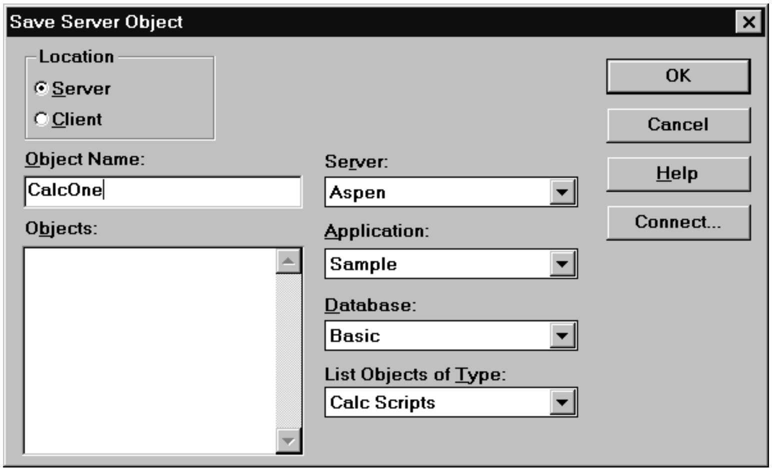
5. Associate the calculation script with an application or database.

   To associate the calculation script with an application and all the databases within the application: First, in the **Application** list, select the required application. Then, in the **Database** list, select **(all dbs)** to associate the calculation with all databases.

   To associate the calculation script with a database: First, in the **Application** list, select the application containing the database. Then, in the **Database** list, select the required database.

6. In the **Object Name** text box, type the name that you want to give the calculation script; for example, CalcOne as shown in Figure 447. You can type up to 8 alphanumeric characters.

Figure 447: Save Server Object Dialog Box



Save Server Object

Location

● Server

○ Client

Object Name: CalcOne

Server: Aspen

Application: Sample

Database: Basic

List Objects of Type: Calc Scripts

7. Click **OK**.

   Essbase saves the calculation script as a calculation script object on the Essbase server.

Calculation script objects associated with an application are saved in the `\`*`arborpath`*`\`app`\`*`appname`* directory on the Essbase server computer. Calculation script objects associated with a database are saved in the `\`*`arborpath`*`\`app`\`*`appname`*`\`*`dbname`* directory on the Essbase server computer. *ARBORPATH* is the Essbase install directory, and *appname* and *dbname* are the application and database with which you have associated the calculation script.

For example, consider the following:

- If you associate a calculation script called CalcTwo with the Sample Basic database and the Essbase install directory is `C:\ESSBASE`, Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\APP\SAMPLE\BASIC`.
- If you associate a calculation script called CalcTwo with the Sample application and the Essbase install directory is `C:\ESSBASE`, Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\APP\SAMPLE`.

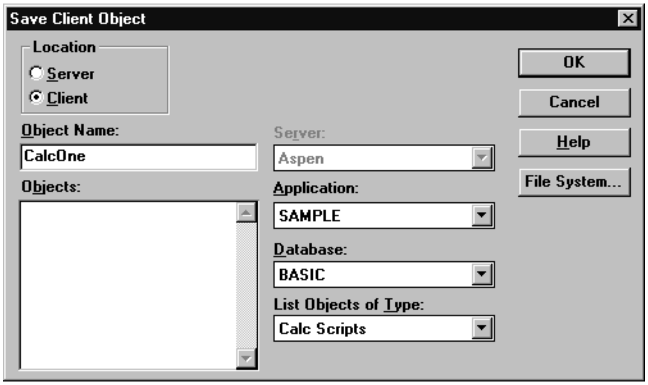
➤ To save a calculation script as an object on your client computer:

1. Open Application Manager and connect to the OLAP Server.
2. Select an application and database in the application desktop window.
3. Click the **Calc Script Editor** button and select an existing calculation script, or create a new calculation script, and display it in the **Calc Script Editor**.
4. In the **Calc Script Editor**, click the **Save** button, 💾.

   If the calculation script is new, Essbase displays either the Save Server Object dialog box or the Save Client Object dialog box, depending on whether you opened Calc Script Editor from the application desktop server window or the client window.

   If Essbase displays the **Save Server Object** dialog box, under **Location**, select **Client**. The **Save Client Object** dialog box shown in Figure 448 replaces the Save Server Object dialog box.

   Figure 448: Save Client Object Dialog Box



Save Client Object

Location
○ Server
● Client

Object Name: CalcOne

Objects:

Server: Aspen

Application: SAMPLE

Database: BASIC

List Objects of Type: Calc Scripts

5. Associate the calculation script with an application or database.

   To associate the calculation script with an application on your client computer and all the databases within the application: First, in the **Application** list, select the required application. Then, in the **Database** list, select **(all dbs)** to associate the calculation script with all databases.

   To associate the calculation script with a database: First, in the **Application** list, select the application containing the database. Then, in the **Database** list, select the required database.

6. In the **Object Name** text box, type the name that you want to give the calculation script. You can type up to 8 alphanumeric characters.

7. Click **OK**.

   Essbase saves the calculation script as a calculation script object on your client computer.

Calculation script objects associated with an application are saved in the `\arborpath\client\appname` directory on the Essbase client computer. Calculation script objects associated with a database are saved in the `\arborpath\client\appname\dbname` directory on the Essbase client computer. *ARBORPATH* is the Essbase install directory, and *appname* and *dbname* are the application and database with which you associate the calculation script.

For example, consider the following factors:

- If you associate a calculation script called CalcTwo with the MYDB01 database in the MYAPP01 application and the Essbase install directory is `C:\ESSBASE`, Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\CLIENT\MYAPP01\MYDB01`.
- If you associate a calculation script called CalcTwo with the MYAPP01 application and the Essbase install directory is `C:\ESSBASE`, Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\CLIENT\MYAPP01`.

➤ To save a calculation script in the file system of a client computer:

1. Open Application Manager and connect to the OLAP Server.
2. Select an application and database in the application desktop window.
3. Click the **Calc Script Editor** button and select an existing calculation script, or create a new calculation script, and display it in the **Calc Script Editor**.
4. In **Calc Script Editor**, click the **Save** button, 💾.

   If the calculation script is new, Essbase displays either the Save Server Object dialog box or the Save Client Object dialog box, depending on whether you opened Calc Script Editor from the application desktop server window or the client window.

   If Essbase displays the **Save Server Object** dialog box, under **Location**, select **Client**. The **Save Client Object** dialog box shown in Figure 449 replaces the Save Server Object dialog box.

5. In the **Save Client Object** dialog box, click **File System**.

   Essbase displays the **Save Client File** dialog box.

6. Enter the required directory and file name, and click **OK**.

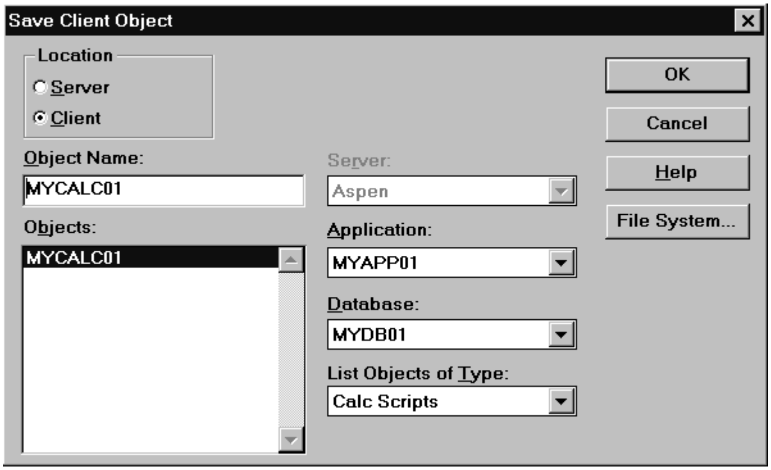
   Essbase saves the calculation script in the directory you specified.

➤ To copy a calculation script from a client computer to the OLAP Server:

1. Open Application Manager and connect to the OLAP Server.
2. Select an application and database in the application desktop window.
3. Click the **Calc Script Editor** button and select an existing calculation script, or create a new calculation script, and display it in the **Calc Script Editor**.
4. From the Application Manager menu, select **File** > **Save As**.

   Essbase displays the **Save Client Object** dialog box as shown in Figure 449.

   Figure 449: Save Client Object Dialog Box



Location: ○ Server ● Client

Object Name: MYCALC01

Objects: MYCALC01

Server: Aspen

Application: MYAPP01

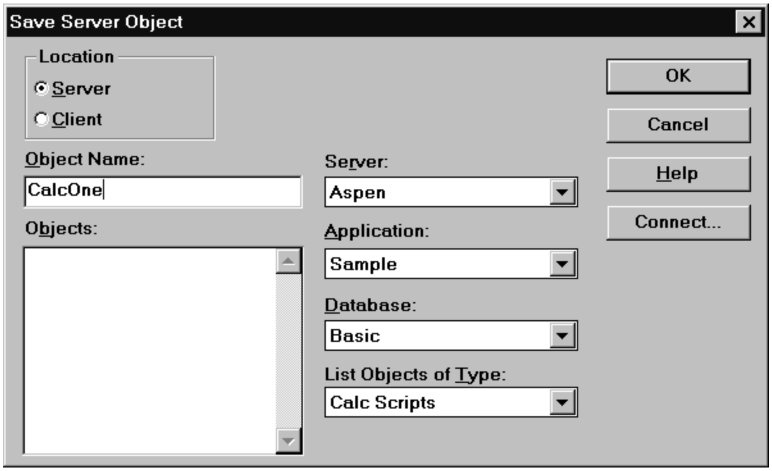
Database: MYDB01

List Objects of Type: Calc Scripts

5. Under **Location**, select **Server**.

The **Save Server Object** dialog box shown in Figure 450 replaces the Save Client Object dialog box.

Figure 450: Save Server Object Dialog Box



Save Server Object

Location: ● Server ○ Client

Object Name: CalcOne

Server: Aspen

Application: Sample

Database: Basic

List Objects of Type: Calc Scripts

6. Select the application or database with which you want to associate the calculation script. For more information, see “Saving a Calculation Script” on page 837.

7. In the **Object Name** text box, type the name that you want to give the calculation script. You can type up to 8 alphanumeric characters.

8. Click **OK**.

   Essbase saves the calculation script on the Essbase server. For more information, see“Saving a Calculation Script” on page 837.

Running a Calculation Script

You can run a calculation script from any of the following:

- Application Manager
- Administration Services (See *Essbase Application Manager Online Help*)
- Essbase Spreadsheet Add-in (See the *Essbase Spreadsheet Add-in User’s Guide*.)

- MaxL (using **execute calculation**)
- ESSCMD

If you run a calculation script from the Application Manager, you can run it on your Essbase client computer or on the OLAP Server.

When you run a calculation script from Application Manager or from Essbase Spreadsheet Add-in, you can view the calculation messages in the application log. When you use MaxL or ESSCMD to run a calculation script, Essbase displays the messages in the ESSCMD window. To display the application log, select Application > View Event Log from the Application Manager menu.

Essbase displays both of the following:

- The calculation order of the dimensions for each pass through the database
- The total calculation time

You can use these messages to tune a database during calculation. To display more detailed information, you can use the SET MSG SUMMARY, SET MSG DETAIL, and SET NOTICE commands in a calculation script. For more information, see "Specifying Global Settings for a Database Calculation" on page 826.

You can run a calculation script from the Application Manager desktop or from the Application Manager menu.

Note: Before you can run a calculation script in Application Manager, you must save it as a calculation script object on the Essbase server or on your client computer. See "Saving a Calculation Script" on page 837. To run a calculation script saved as an object on your client computer, you must run the calculation script from the desktop.

➤ To run a calculation script using Application Manager:

1. If the calculation script is saved on the Essbase server, open Application Manager and select the application desktop server window.

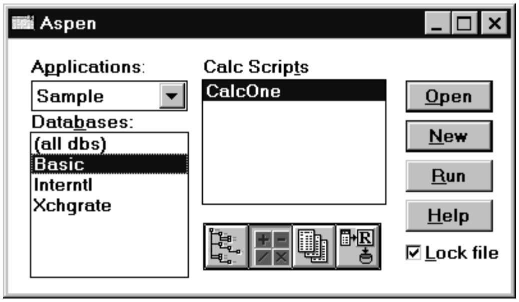
   If the calculation script is saved on your client computer, open the application desktop client window.

2. In the application desktop server window or the client window, select the application and database that contains the calculation script you want to run.

3. Click the **Calc Scripts** button, , to display the calculation scripts associated with the application and database that you selected.

   Figure 451 shows the application desktop server window for a server called Aspen.

   Figure 451: Application Desktop Server Window



Aspen

Applications: Sample

Databases: (all dbs), Basic, Interntl, Xchgrate

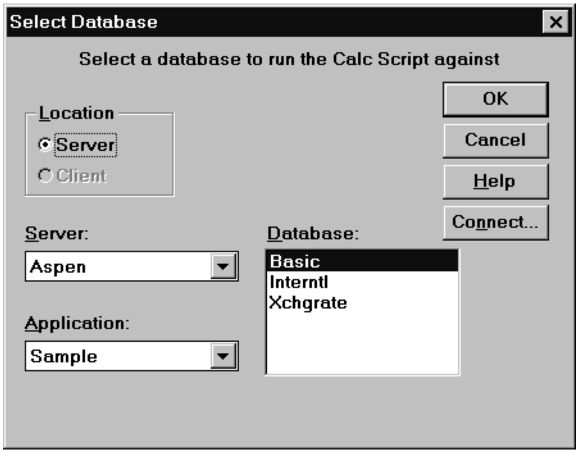
Calc Scripts: CalcOne

☑ Lock file

4. In the **Calc Scripts** list, select the calculation script that you want to run, and click **Run**.

   Essbase displays the **Select Database** dialog box as shown in Figure 452.

   Figure 452: Select Database Dialog Box



Select Database

Select a database to run the Calc Script against

Location

● Server

○ Client

Server: Aspen

Application: Sample

Database: Basic, Interntl, Xchgrate

5. Select the Essbase server, application, and database against which you want to run the calculation script.

   If you are not currently connected to the server, click **Connect**.

6. Click **OK**.

   Essbase runs the calculation script against the database that you selected.

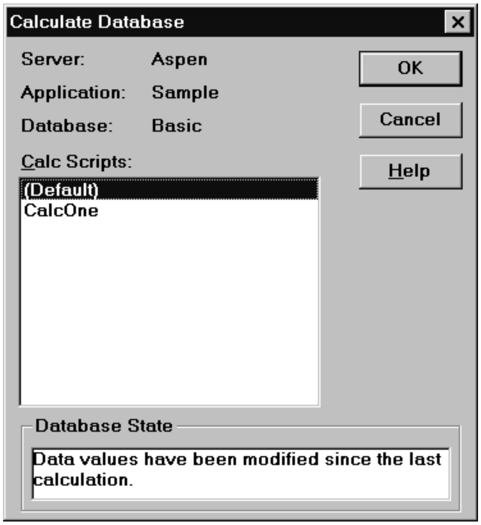
➤ To run a calculation script from the menu:

1. In the application desktop server window, select the application and database that contains the calculation script that you want to run.

2. From the Application Manager, select the **Database** > **Calculate** menu command.

   Essbase displays the **Calculate Database** dialog box as shown in Figure 453.

   Figure 453: Calculate Database Dialog Box



Calculate Database

Server: Aspen

Application: Sample

Database: Basic

Calc Scripts:

(Default)
CalcOne

Database State

Data values have been modified since the last calculation.

3. In the **Calc Scripts** list, select the calculation script that you want to run and click **OK**. Only scripts to which you have security access are displayed in the list.

   Essbase runs the calculation script against the database you selected in the application desktop server window.

Tip: You can execute a calculation script outside Application Manager.

Tool	Instructions	For More Information
Administration Services	Calculation Script Editor	*Essbase Application Manager Online Help*
MaxL	**execute calculation**	The *Technical Reference* in the `docs` directory
Essbase Spreadsheet Add-in	Essbase > Calculation	The *Essbase Spreadsheet Add-in User's Guide*
ESSCMD	RUNCALC	The *Technical Reference* in the `docs` directory

Printing a Calculation Script

You can print a calculation script from Calc Script Editor.

➤ To print a calculation script:

1. Open the Application Manager and connect to the OLAP Server.
2. From the application desktop, select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the **Calc Script Editor**.
4. Select the **File** > **Print** menu command, or click the **Print** button, 🖨.

   Essbase displays the **Print Calc Script** dialog box.
5. If you want to print page numbers on a calculation script, check **Page Numbers**.
6. Click **Print**.

Deleting a Calculation Script

How you delete a calculation script depends on where it is saved.

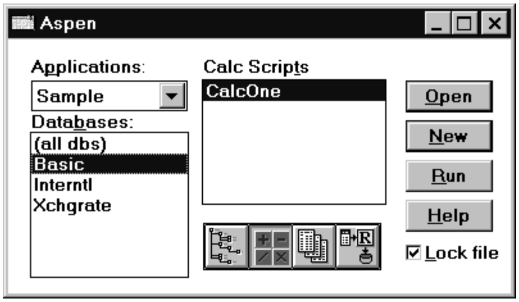
➤ To delete a calculation script saved as an object on the Essbase server or on a client computer:

1. Open the Application Manager and connect to the OLAP Server.
1. In the application desktop server or client window, select the application and database with which the calculation script is associated.
2. Click the **Calc Scripts** button, +−/×.

   Essbase displays a list of all the calculation scripts associated with the application and database that you chose.

   In Figure 454, the calculation script, CalcOne, is shown as associated with Sample Basic.

   Figure 454: Application Desktop Server Window



Aspen

Applications: Sample

Databases: (all dbs), Basic, Interntl, Xchgrate

Calc Scripts: CalcOne

☑ Lock file

3. In the **Calc Scripts** list, select the calculation script that you want to delete.
4. From the Application Manager menu, select **File** > **Delete**.

   Essbase displays a **Confirm Delete** message box.
5. Click **Yes** to delete the calculation script.

➤ To delete a calculation script saved in the file system of a client computer, delete the calculation script file by using the client computer's file system. You cannot delete the file using Application Manager.

➤ To undo the last action: in Calc Script Editor, select Edit > Undo, or click the button.

Using Formulas in a Calculation Script

You can place member formulas in a calculation script. When you place formulas in a calculation script, they override any conflicting formulas that are applied to members in the database outline.

In a calculation script, you can do both of the following:

- Calculate a member formula on the database outline
- Define a formula

To calculate a formula that is applied to a member in the database outline, simply use the member name followed by a semicolon (;). For example:

```
Variance;
```

calculates the formula applied to the Variance member in the database outline.

To define a formula in a calculation script, use Calc Script Editor. For example:

```
Expenses = Payroll + Marketing + Misc;
```

cycles through the database, adding the values in the members Payroll, Marketing, and Misc and placing the result in the Expenses member. This formula overrides any formula placed on the Expenses member in the database outline.

Note: You cannot apply formulas to shared members or label only members.

Basic Equations

You can define basic equations in a calculation script as follows:

Member = mathematical operation;

where *Member* is a member name from the database outline, and *mathematical operation* is any valid mathematical operation.

For example, the following formula causes Essbase to cycle through the database, subtracting the values in COGS from the values in Sales and placing the result in Margin:

```
Margin = Sales - COGS;
```

The next formula cycles through the database subtracting the values in Cost from the values in Retail, calculating the resulting values as a percentage of the values in Retail, and placing the results in Markup:

```
Markup = (Retail - Cost) % Retail;
```

For more information on the nature of multidimensional calculations, see Chapter 2, “Multidimensional Concepts.”

Conditional Equations

When you use an IF statement as part of a member formula in a calculation script, you need to do both of the following:

- Associate the IF statement with a single member
- Enclose the IF statement in parentheses

For example:

```
Profit
(IF (Sales > 100)
   Profit = (Sales - COGS) * 2;
ELSE
  Profit = (Sales - COGS) * 1.5;
ENDIF;)
```

Essbase cycles through the database and performs the following calculations:

1. The IF statement checks to see if the value of Sales for the current member combination is greater than 100.
2. If Sales is greater than 100, Essbase subtracts the value in COGS from the value in Sales, multiplies the difference by 2, and places the result in Profit.
3. If Sales is less than or equal to 100, Essbase subtracts the value in COGS from the value in Sales, multiplies the difference by 1.5, and places the result in Profit.

The whole of the IF … ENDIF statement is enclosed in parentheses and associated with the Profit member, `Profit (IF(...)...)`.

Interdependent Formulas

When you use an interdependent formula in a calculation script, the same rules apply as for the IF statement. You need to do both of the following:

- Associate the formula with a single member
- Enclose the formula in parentheses

Consider the interdependent formula discussed earlier. If you place the formula in a calculation script, you construct it as follows:

```
"Opening Inventory"
(IF(NOT @ISMBR (Jan))"Opening Inventory" =
    @PRIOR("Ending Inventory"));
    ENDIF;
"Ending Inventory" = "Opening Inventory" - Sales + Additions;)
```

The whole of the formula is enclosed in parentheses and is associated with the Opening Inventory member, as follows:

```
"Opening Inventory"(IF(...)...)
```

Inserting Text and Operators in a Calculation Script

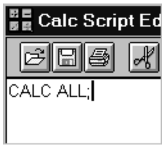
You can type text and operators directly into the text area of Calc Script Editor, or you can use the toolbar buttons to add the text and operators. You can also cut, copy, and search for text in Calc Script Editor.

➤ To type text in Calc Script Editor:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the **Calc Script Editor**.
4. In the **Calc Script Editor**, click in the text area below the toolbar.
5. Type the appropriate text.

   Text is displayed at the cursor position as you type as shown in Figure 455.

   Figure 455: Calc Script Editor Showing Calculation Script

   ```
   CALC ALL;
   ```

➤ To insert an equal (=) sign in Calc Script Editor:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the **Calc Script Editor**.
4. Place the cursor where you want to insert the equal sign (=).
5. Type **=** or click the = button.

➤ To insert a mathematical operator (+, -, X, /, %) in Calc Script Editor:

1. Place the cursor where you want to insert the mathematical operator.
2. Type the appropriate operator or click one of the following toolbar buttons:

For example, to insert an addition operator (+), place the cursor where you want to insert the addition (+) operator, and type **+** or click the + button.

➤ To insert the cross-dimensional operator (->) in Calc Script Editor:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the Calc Script Editor.
4. Place the cursor where you want to insert the cross-dimensional operator.
5. Type a **-** (hyphen) followed by a **>** (greater than symbol), or click the → button.

For more information on the cross-dimensional operator, see Chapter 25, "Developing Formulas."

➤ To insert the semicolon formula end-of-line character (;) in Calc Script Editor:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the Calc Script Editor.
4. Place the cursor at the end of the formula.
5. Type a **;** (semicolon), or click the ; button.

➤ To insert a function or operator in Calc Script Editor:

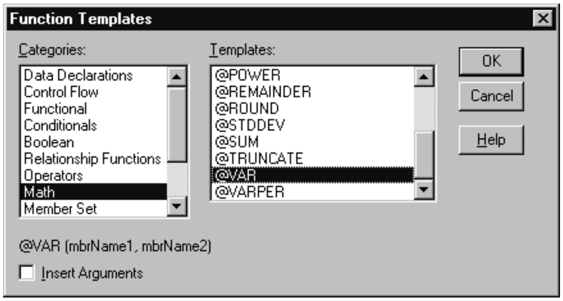
1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the Calc Script Editor.
4. Place the cursor where you want to insert the function.
5. Select **Formula** > **Paste Function**, or click the f(x) button.

   Essbase displays the **Function Templates** dialog box.
6. In the **Categories** list, select the function category.

   For example, to insert the @VAR function, select **Math**.
7. In the **Templates** list, select the required function or operator.

   In the example shown in Figure 456, scroll down the list and select **@VAR**. Essbase displays the function or operator and the default arguments below the **Categories** list.

Figure 456: Function Templates Dialog Box



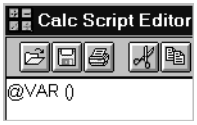
@VAR (mbrName1, mbrName2)

□ Insert Arguments

8. If required, check **Insert Arguments** to insert default, temporary arguments for the function.
9. Click **OK**.

   Essbase inserts `@VAR` at the cursor position as shown in Figure 457.

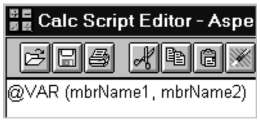
   Figure 457: Calc Script Editor With @VAR Function Inserted

   ```
   @VAR ()
   ```

   If you checked Insert Arguments, Essbase inserts `@VAR` and default, temporary arguments as shown in Figure 458. You can then type over the temporary arguments with the correct arguments.

   Figure 458: Calc Script Editor With @VAR Function and Arguments Inserted

   ```
   @VAR (mbrName1, mbrName2)
   ```

➤ To cut text in Calc Script Editor:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the **Calc Script Editor**.
4. Select the text that you want to cut.
5. Select **Edit** > **Cut**, click the ✂ button, or press Ctrl + X.

➤ To copy text in Calc Script Editor:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the **Calc Script Editor**.
4. Select the text that you want to copy.
5. Select **Edit** > **Copy**, click the ⎘ button, or press Ctrl + C.

➤ To paste text in Calc Script Editor:

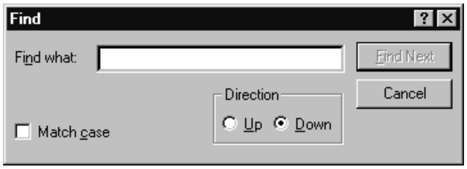
1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the **Calc Script Editor**.
4. Select the text that you want to paste.
5. Select **Edit** > **Paste**, click the 📋 button, or press Ctrl + V.

➤ To find and replace text in Calc Script Editor:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the Calc Script Editor.
4. Select **Edit** > **Find**.

   Essbase displays the **Find** dialog box as shown in Figure 459.

   Figure 459: Calc Script Editor Find Dialog Box

5. In the **Find what** text box, type the characters you want to search for, and click **Find Next**.

➤ To do a case-sensitive search:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the **Calc Script Editor**.
4. In the **Find** dialog box, check **Match case**.

   For example, to search for Margin but not margin, type **Margin** in the **Find** what text box and check **Match case**.
5. Click **Find Next**.

Associating a Calculation Script with a Database

If you want to insert member names in a calculation script by selecting them within Calc Script Editor, you need to associate the calculation script with the database outline that contains the members.

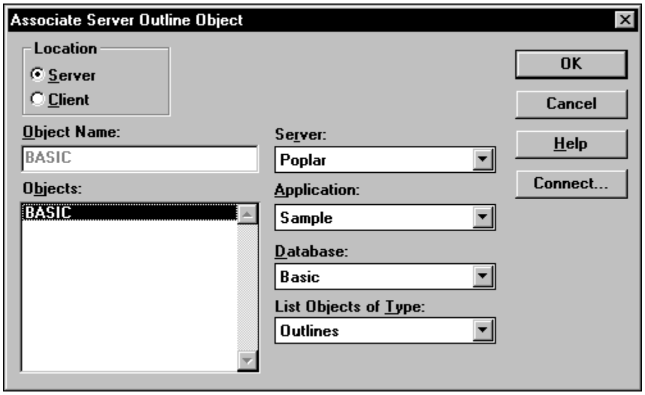
➤ To associate a calculation script with a database outline:

1. Open the Application Manager and connect to the OLAP Server.
2. Select the application and database that contains the calculation script.
3. Double-click the calculation script to open it in the **Calc Script Editor**.

4. Click the button or, from the **Calc Script Editor** menu of the Application Manager, select **Options** > **Associate Outline**.

   Essbase displays the **Associate Client Outline Object** or the **Associate Server Outline Object** dialog box. The latter is shown in Figure 460.

   Figure 460: Associate Server Outline Object Dialog Box



Associate Server Outline Object

Location
◉ Server
○ Client

Object Name: BASIC

Objects: BASIC

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Outlines

5. Do one of the following:

   - To associate an outline saved on your client computer, under **Location**, select **Client**.
   - To associate an outline saved on the Essbase server computer, under **Location**, select **Server**.

6. In the **Server**, **Application**, and **Database** lists, select the server, application, and database that contain the outline that you want to associate with a calculation script.

7. In the **Objects** list, select the database outline.

8. Click **OK**.

   Essbase displays the dimension names of the associated outline in the Dimensions list. You can now insert members from this list. See “Associating a Calculation Script with a Database” on page 857.

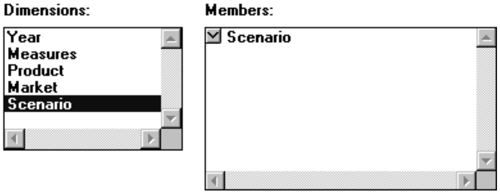
Essbase associates the calculation script with the database outline only while you are editing the calculation script. When you close Calc Script Editor, Essbase cancels the association. If you want to insert members from the database outline in the future, you need to re-associate the outline with the calculation script.

➤ To insert the name of a dimension in a calculation script:

1. Associate the database outline that contains the dimensions you want to insert. See "Associating a Calculation Script with a Database" on page 857.
2. In the **Calc Script Editor** of the Application Manager, place the cursor where you want to insert the member name.
3. In the **Dimensions** list, select the dimension that contains the member you want to insert in a formula.

   The name of the dimension is displayed in the **Members** list. If a ☑ button is displayed to the left of the dimension name, the dimension has children. Figure 461 shows the **Scenario** dimension in the **Sample Basic** database.

   Figure 461: Inserting Dimensions and Members In a Calculation Script



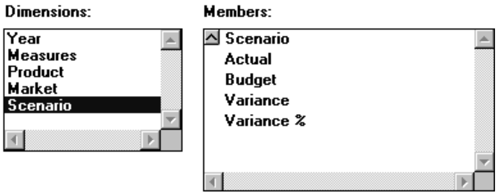
   If you want to insert the name of the dimension in a formula, click the dimension name in the Members list. Essbase inserts the dimension name at the cursor position.

➤ To expand and collapse a member branch:

1. To display a member's children, in the **Members** list, double-click the ⌄ button next to the member name.

   The ⌄ button changes to a ⌃ button as shown in Figure 462.

   Figure 462: Expanding a Member Branch

   Dimensions:

   Year
Measures
Product
Market
Scenario

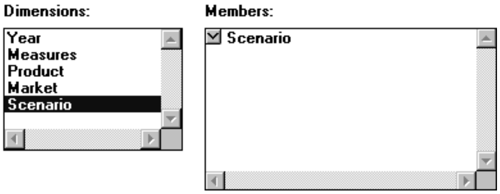
   Members:

   ⌃ Scenario
Actual
Budget
Variance
Variance %

2. To collapse the member branch, double-click the ⌃ button.

   Essbase does not display the member's children. The ⌃ button changes to a ⌄ button as shown in Figure 463.

   Figure 463: Collapsing a Member Branch

   Dimensions:

   Year
Measures
Product
Market
Scenario

   Members:

   ⌄ Scenario

➤ To search for a member:

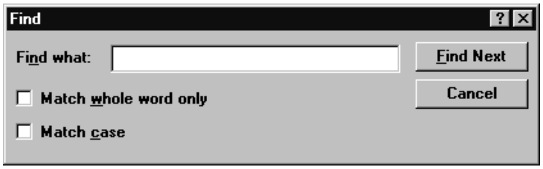
1. In the **Dimensions** list, select the dimension in which you want to search for a member.

   For example, select the **Measures** dimension from the **Sample Basic** database.

2. Click **Find Member**.

   Essbase displays the **Find** dialog box as shown in Figure 464.

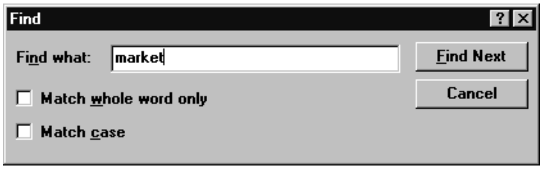
   Figure 464: Opening the Find Dialog Box

3. In the **Find what** text box, type the characters that you want to search for.

   For example, to search for the Marketing member in the Measures dimension, enter **market** as shown in Figure 465.

   Figure 465: Searching For Members

   
   Find what: market

   a. To make the search case-sensitive, check **Match case**.

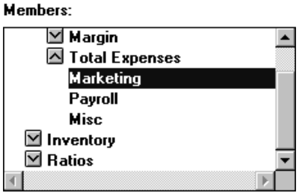
   b. To search for whole words only, check **Match whole word only**.

      For example, to search for Margin, but not Margin % in the Sample Basic database, type **margin**, and check **Match whole word only**.

4. Click **Find Next**.

   Essbase finds and selects the appropriate member as shown in Figure 466.

   Figure 466: Finding A Specific Member

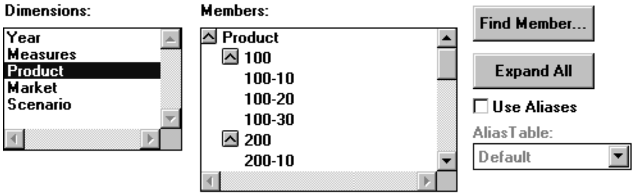
   

➤ To expand a dimension to display all members in the Calc Script Editor of the Application Manager:

1. In the **Dimensions** list, select the dimension for which you want to display all the members.

   For example, select the **Product** dimension in the **Sample Basic** database. Figure 467 shows all of the members under the Product dimension.

   Figure 467: Expanding a Dimension



Dimensions:
Year
Measures
Product
Market
Scenario

Members:
Product
100
100-10
100-20
100-30
200
200-10

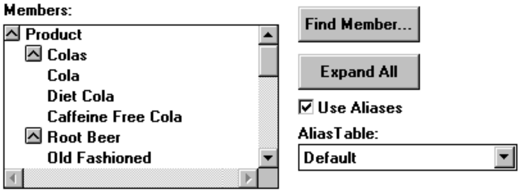
□ Use Aliases

AliasTable: Default

2. Click **Expand All**.

   In the **Members** list, Essbase displays all members in the dimension, an example of which is shown in Figure 468.

   Figure 468: Expanding a Dimension



Members:
Product
Colas
Cola
Diet Cola
Caffeine Free Cola
Root Beer
Old Fashioned

☑ Use Aliases

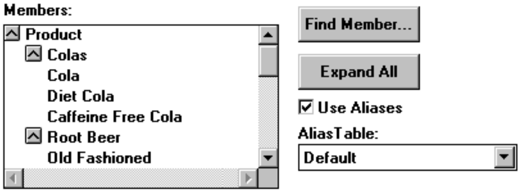
AliasTable: Default

➤ To display and insert alias names in the Calc Script Editor of the Application Manager:

1. Check **Use Aliases**.

   Essbase displays the alias names for the members. Figure 469 shows the Product dimension from the Sample Basic database.

   Figure 469: Displaying and Inserting Alias Names

   Members:
   - Product
     - Colas
       - Cola
       - Diet Cola
       - Caffeine Free Cola
     - Root Beer
       - Old Fashioned

   ☑ Use Aliases

   AliasTable: Default

2. To select a different alias table, from the **Alias Table** list box, select the table.

   When you select a member from the Members list, Essbase inserts the alias name at the cursor position. If required, Essbase automatically encloses the alias name in double quotation marks (" ").

Checking Syntax

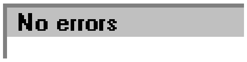
Essbase includes a syntax checker that tells you about any syntax errors in a calculation script. For example, Essbase tells you if you have typed a function name incorrectly.

The syntax checker cannot tell you about semantic errors in a calculation script. Semantic errors occur when a calculation script does not work as you expect. To find semantic errors, always run the calculation, and check the results to ensure they are as you expect.

➤ To check the syntax of a calculation script in Calc Script Editor of the Application Manager:

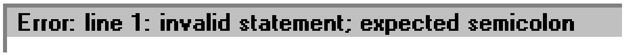
Select **Syntax** > **Check Syntax** or click the button.

Essbase displays the syntax checker results at the bottom of the Calc Script Editor window. If Essbase finds no syntax errors, it displays the following message:



```
No errors
```

If Essbase finds one or more syntax errors, it displays the number of the line that includes the error and a brief description of the error. For example, if you do not include a semicolon end-of-line character at the end of a calculation script command, Essbase displays a message similar to the following:

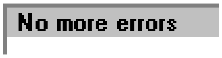


```
Error: line 1: invalid statement; expected semicolon
```

➤ To step through syntax errors in Calc Script Editor of the Application Manager:

Select **Syntax** > **Next Error or Syntax** > **Previous Error**.

When you reach the first or last error, Essbase displays the message:



```
No more errors
```

Essbase maintains the list of error messages until you check the syntax again.

Using a Calculation Script to Control Intelligent Calculation

Assume that you have a formula on a sparse dimension member and the formula contains either of the following:

- A relationship function (for example, @PRIOR or @NEXT)
- A financial function (for example, @NPV or @INTEREST)

Essbase always recalculates the data block that contains the formula, even if the data block is marked as clean for the purposes of Intelligent Calculation. For more information, see Chapter 52, “Optimizing with Intelligent Calculation.”

Grouping Formulas and Calculations

You may achieve significant calculation performance improvements by carefully grouping formulas and dimensions in a calculation script. For more information and examples, see “Calculating a Series of Member Formulas” on page 865 and “Calculating a Series of Dimensions” on page 866.

When you run a calculation script, Essbase automatically displays the calculation order of the dimensions for each pass through the database. Thus, you can tell how many times Essbase has cycled through the database during the calculation.

Essbase displays these information messages in the ESSCMD window and in the application log. To display the application log, select Application > View Event Log from the Application Manager menu.

Calculating a Series of Member Formulas

When you calculate formulas, avoid using parentheses unnecessarily. The following formulas cause Essbase to cycle through the database once, calculating both formulas in one pass:

```
Profit = (Sales - COGS) * 1.5;
Market = East + West;
```

Similarly, the following configurations cause Essbase to cycle through the database only once, calculating the formulas on the members Qtr1, Qtr2, and Qtr3:

```
Qtr1;
Qtr2;
Qtr3;
```

or

```
(Qtr1;
Qtr2;
Qtr3;)
```

However, the inappropriately placed parentheses in the following example cause Essbase to cycle through the database twice, once calculating the formulas on the members Qtr1 and Qtr2 and once calculating the formula on Qtr3:

```
(Qtr1;
Qtr2;)
Qtr3;
```

Calculating a Series of Dimensions

When you calculate a series of dimensions, you can optimize performance by grouping the dimensions wherever possible.

For example, the following formula causes Essbase to cycle through the database only once:

```
CALC DIM(Year, Measures);
```

However, the following syntax causes Essbase to cycle through the database twice. It cycles through once for each CALC DIM command:

```
CALC DIM(Year);
CALC DIM(Measures);
```

Using Substitution Variables

You can use substitution variables in calculation scripts. Substitution variables are useful, for example, when you reference information or lists of members that change frequently.

When you include a substitution variable in a calculation script, Essbase replaces the substitution variable with the value you specified for the substitution variable.

You create and specify values for substitution values in Essbase Application Manager. For more information, see Chapter 6, “Creating Applications and Databases.”

You can create variables at the server, application, and database levels. When you use a substitution variable in a calculation script, it must be available to the calculation script. For example, if you create a substitution variable at the database level, it is only available to calculation scripts within the database. However, if you create a variable at the server level, it is available to any calculation script on the server.

The ampersand (&) character prefaces a substitution variable in a calculation script. Essbase treats any string that begins with a leading ampersand as a substitution variable, replacing the variable with its value before parsing the calculation script.

For example, `&CurQtr;` becomes `Qtr1;` if you have given the substitution variable `&CurQtr` the value `Qtr1`.

Consider an example in which you want to calculate Sample Basic data for the current quarter. You can use the following calculation script:

```
FIX(&CurQtr)
CALC DIM(Measures, Product);
ENDFIX
```

You then define the substitution variable `CurQtr` as the current quarter; for example, Qtr3. Essbase replaces the variable CurQtr with the value Qtr3 when it runs the calculation script.

Clearing Data

You can use the CLEARDATA and CLEARBLOCK calculation commands to remove data values and data blocks from a database. You can use the CLEARBLOCK DYNAMIC command to remove blocks for Dynamic Calc And Store member combinations. For more information, see Chapter 28, "Dynamically Calculating Data Values."

When you use the CLEARBLOCK command, Essbase removes the entire contents of a block, including all the dense dimension members. Essbase removes the entire block, unless CLEARBLOCK is inside a FIX command on members within the block.

The following examples are based on the Sample Basic database. If the Scenario dimension is dense, the following example removes all the data cells that do not contain input data values and intersect with member Actual from the Scenario dimension

```
FIX(Actual)
CLEARBLOCK NONINPUT;
ENDFIX
```

If the Scenario dimension is sparse, the following formula removes only the blocks whose Scenario dimension member is Actual. The other blocks remain:

```
FIX(Actual)
CLEARBLOCK NONINPUT;
ENDFIX
```

When you use the CLEARDATA command, Essbase changes the values of the cells you specify to `#MISSING`. The data blocks are not removed.

For example, the following formula clears all the Actual data values for Colas:

```
CLEARDATA Actual -> Colas;
```

You can use the FIX command with the CLEARDATA command to clear a subset of a database. If you want to clear an entire database, you can select the Clear Data command from the Database menu in Application Manager.

For more information on the CLEARBLOCK and CLEARDATA calculation commands, see the *Technical Reference* in the `docs` directory.

Copying Data

You can use the DATACOPY calculation command to copy data cells from one range to another range in a database. The two ranges must be the same size.

For example, in the Sample Basic database, the following formula copies Actual values to Budget values:

```
DATACOPY Actual TO Budget;
```

You can use the FIX command to copy a subset of values.

For more information on the FIX command, see the *Technical Reference* in the `docs` directory.

Calculating a Subset of a Database

You can calculate a subset of a database, which means that you can use different formulas to calculate separate sections of a database.

To calculate a subset of a database, you can use either of the following:

- Member set functions to calculate lists of members
- The FIX … ENDFIX commands to calculate a range of values

For more information, see "Calculating Lists of Members" on page 869 and "Using the FIX Command" on page 870.

Note: When you have Intelligent Calculation turned on, the newly calculated data blocks are not marked as clean after a partial calculation of a database. When you calculate a subset of a database, you can use the SET CLEARUPDATESTATUS AFTER command to ensure that the newly calculated blocks are marked as clean. Using this command ensures that Essbase recalculates the database as efficiently as possible using Intelligent Calculation. For more information on Intelligent Calculation, see Chapter 52, "Optimizing with Intelligent Calculation." For more information on the SET CLEARUPDATESTATUS command, see the *Technical Reference* in the `docs` directory.

Calculating Lists of Members

You can use a member set function to generate a list of members that is based on a member you specify. For example, you can use the @IDESCENDANTS function to generate a list of all the descendants of a specified member.

In the Sample Basic database, `@IDESCENDANTS("Total Expenses");` generates the following list of members: Total Expenses, Marketing, Payroll, and Misc.

When you use a member set function in a formula, Essbase generates a list of members before calculating the formula.

For detailed information on these and other member set functions, see the *Technical Reference* in the `docs` directory.

Using the FIX Command

The FIX … ENDFIX commands are particularly useful to calculate a carefully defined subset of the values in a database. For example, the following calculation script calculates only the Budget values for only the descendants of East (New York, Massachusetts, Florida, Connecticut, and New Hampshire) in the Sample Basic database:

```
FIX(Budget,@DESCENDANTS(East))
CALC DIM(Year, Measures, Product);
ENDFIX
```

The next example fixes on member combinations for the children of East that have a user-defined attribute (UDA) of New Mkt. For information on defining UDAs, see Chapter 7, “Creating and Changing Database Outlines.”

```
FIX(@CHILDREN(East) AND @UDA(Market,"New Mkt"))
Marketing = Marketing * 1.1;
ENDFIX
```

The next example uses a wildcard match to fix on member names that end in the characters -10. In Sample Basic, this example fixes on the members 100-10, 200-10, 300-10, and 400-10.

```
FIX(@MATCH(Product, "???-10"))
Price = Price * 1.1;
ENDFIX
```

When you use the FIX command *only* on a dense dimension, Essbase retrieves the entire block that contains the required value or values for the member or members that you specify. Thus, I/O is not affected, and the calculation performance time is improved.

When you use the FIX command on a sparse dimension, Essbase retrieves the block for the specified sparse dimension member or members. Thus, I/O may be greatly reduced.

Essbase cycles through the database once for each FIX command that you use on dense dimension members. When possible, combine FIX blocks to improve calculation performance. For example, the following calculation script causes Essbase to cycle through the database only once, calculating both the Actual and the Budget values:

```
FIX(Actual,Budget)
CALC DIM(Year, Measures);
ENDFIX
```

However, this calculation script causes Essbase to cycle through the database twice, once calculating the Actual data values and once calculating the data values for Budget:

```
FIX(Actual)
CALC DIM(Year, Measures);
ENDFIX
FIX(Budget)
CALC DIM(Year, Measures);
ENDFIX
```

You cannot FIX on a subset of a dimension that you calculate within a FIX statement. For example, the following calculation script returns an error message because the CALC DIM operation calculates the entire Market dimension, although the FIX above it fixes on specific members of the Market dimension.

```
FIX(@CHILDREN(East) AND @UDA(Market,"New Mkt"))
CALC DIM(Year, Measures, Product, Market);
ENDFIX
```

For detailed information on using the FIX command, see the *Technical Reference* in the `docs` directory.

Writing Calculation Scripts for Partitions

A Essbase OLAP Server partitioned application can span multiple servers, processors, or computers. For more information on partitioning, see Chapter 13, “Designing Partitioned Applications” and Chapter 14, “Building and Maintaining Partitions.”

You can achieve significant calculation performance improvements by partitioning applications and running separate calculations on each partition.

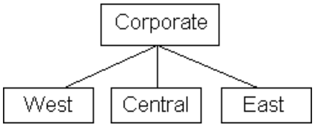
However, when you use partitioning, you need to do both of the following:

- Consider carefully the performance impact on the overall database calculation. You might choose to do any of the following:
  - Redesign the overall calculation to avoid referencing remote values that are in a transparent partition in a remote database.
  - Dynamically calculate a value in a remote database. See Chapter 28, “Dynamically Calculating Data Values.”
  - Replicate a value in the database that contains the applicable formula. See Chapter 13, “Designing Partitioned Applications.” Ensure that you replicate only the required values. For example, if you are replicating quarterly data for the Eastern region, replicate only the values for Qtr1, Qtr2, Qtr3, and Qtr4, and calculate the parent Year values locally.
- Ensure that a referenced value is up-to-date when Essbase retrieves it. Choose one of the options previously discussed (redesign, dynamically calculate, or replicate) or calculate the referenced database before calculating the formula.

Controlling Calculation Order for Partitions

You need to calculate databases in a specific order to ensure that Essbase calculates the required results. For example, consider the following partitions in which you view information from the West, Central, and East databases transparently from the Corporate database.

Figure 470: Calculating Partitions



West, Central, and East contain only actual values. Corporate contains actual and budgeted values. Although you can view the West, Central, and East data in the Corporate database, the data exists only in the West, Central, and East databases; it is not duplicated in the Corporate database.

Therefore, when Essbase calculates Corporate, it needs to take the latest values from West, Central, and East. To obtain the required results, you need to calculate West, Central, and East before you calculate Corporate.

Chapter 31 Examples of Calculation Scripts

The examples in this chapter illustrate different types of calculation scripts, which you may want to adapt for your own use.

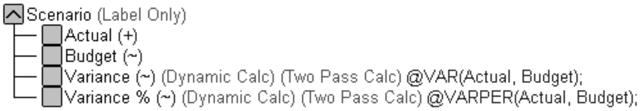
This chapter includes the following examples:

For more examples that use the Intelligent Calculation commands SET UPDATECALC and SET CLEARUPDATESTATUS in calculation scripts, see Chapter 52, "Optimizing with Intelligent Calculation."

Calculating Variance

The Sample Basic database includes a calculation of the percentage of variance between Budget and Actual values.

Figure 471: Calculating Variance and Variance %



```
Scenario (Label Only)
├── Actual (+)
├── Budget (~)
├── Variance (~) (Dynamic Calc) (Two Pass Calc) @VAR(Actual, Budget);
└── Variance % (~) (Dynamic Calc) (Two Pass Calc) @VARPER(Actual, Budget);
```

During a default calculation of the Sample Basic database, Hyperion Essbase aggregates the values on the Market and Product dimensions. Aggregating percentage values does not produce the correct result. Therefore, the Variance % formula needs to be recalculated after the default calculation.

In the Sample Basic outline, Variance % is tagged as a Dynamic Calc, two-pass member, which means that Hyperion Essbase dynamically calculates Variance % values when you retrieve them. This calculation overwrites the incorrect values with the correctly calculated percentages. If you choose not to tag Variance % as a Dynamic Calc, two-pass member, you could use the following calculation script to recalculate the percentages. For more information on dynamic calc members, see Chapter 28, “Dynamically Calculating Data Values.”

Assuming that Intelligent Calculation is turned on (the default), this calculation script performs a default calculation and then recalculates the formula on Variance %:

```
CALC ALL;
SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
"Variance %";
```

Hyperion Essbase performs the following calculations:

1. Hyperion Essbase uses the CALC ALL command to perform a default calculation of the database. Alternatively, you could run a default calculation of the database outline without using a calculation script.
2. The SET UPDATECALC OFF command turns off Intelligent Calculation.

3. The CLEARUPDATESTATUS AFTER command tells Hyperion Essbase to mark the calculated blocks as clean, even though this is a partial calculation of the database (by default, data blocks are marked as clean only after a full calculation of the database).
4. Hyperion Essbase cycles through the database calculating the formula for Variance %.

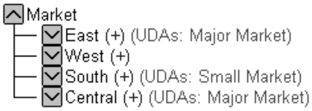
For information on calculating *statistical* variance, see the *Technical Reference* in the `docs` directory.

For more information on using a calculation script for two-pass calculations, see Chapter 51, “Optimizing Calculations.” For more information on developing formulas, see Chapter 25, “Developing Formulas.”

Calculating a Subset of a Database

This example is based on the Sample Basic database. The Marketing managers of each of the regions East, West, South, and Central need to calculate their corresponding areas of the database.

Figure 472: Market Dimension from the Sample Basic Database



```
Market
├─ East (+) (UDAs: Major Market)
├─ West (+)
├─ South (+) (UDAs: Small Market)
└─ Central (+) (UDAs: Major Market)
```

The marketing manager of the region East uses this calculation script to calculate the data values for East:

```
/* 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
```

This script calculates the Year, Measures, and Products dimensions for each child of East.

Hyperion Essbase performs the following calculations:

1. Hyperion Essbase fixes on the Budget values for the descendants of East.
2. The Year, Measures, and Products dimensions are calculated in one pass of the database for each of the Budget values of the descendants of East.
3. Hyperion Essbase fixes on the Budget values for all members on the other dimensions.
4. Hyperion Essbase aggregates the descendants of East and places the result in East.

The following three calculation scripts are used by the marketing managers of the other regions:

```
/* Calculate the Budget data values for the descendants of West */
FIX(Budget, @DESCENDANTS(West))
CALC DIM(Year, Measures, Product);
ENDFIX
/* Consolidate West */
FIX(Budget)
@DESCENDANTS(West);
ENDFIX
```

```
/* Calculate the Budget data values for the descendants of South
*/
FIX(Budget, @DESCENDANTS(South))
CALC DIM(Year, Measures, Product);
ENDFIX
/* Consolidate South */
FIX(Budget)
@DESCENDANTS(South);
ENDFIX
```

```
/* Calculate the Budget data values for the descendants of
Central */
FIX(Budget, @DESCENDANTS(Central))
CALC DIM(Year, Measures, Product);
ENDFIX
/* Consolidate Central */
FIX(Budget)
@DESCENDANTS(Central);
ENDFIX
```

Loading New Budget Values

This example loads budget values into the Sample Basic database and recalculates the database:

```
/* Recalculate all Budget values */
FIX(Budget)
CALC DIM(Year, Product, Market, Measures);
ENDFIX

/* Recalculate the Variance and Variance % formulas, which
      require two passes */
Variance;
"Variance %";
```

Hyperion Essbase performs these calculations:

1. Hyperion Essbase fixes on the Budget values.
2. Hyperion Essbase calculates all Budget values. The CALC DIM command is used to calculate all the dimensions except for the Scenario dimension, which contains Budget.
3. Hyperion Essbase calculates the formula applied to Variance in the database outline.
4. Hyperion Essbase calculates the formula applied to Variance % in the database outline.

Calculating Product and Market Share Values

The following example is based on the Sample Basic database. It calculates product share and market share values for each market and each product.

The product and market share values are calculated based on:

- Each member as a percentage of the total.
- Each member as a percentage of its parent.

Assume that you add four members to the Measures dimension: Market Share, Product Share, Market %, and Product %.

```
/* 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);
```

Hyperion Essbase performs these calculations:

1. Hyperion Essbase fixes on the Sales values and consolidates all the Sales values. The CALC DIM command is used to calculate the Year, Market, and Product dimensions. The Measures dimension contains the Sales member and therefore is not consolidated. The Scenario dimension is label only and therefore does not need to be consolidated.
2. Hyperion Essbase cycles through the database and calculates Market Share. It takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of total Sales in all markets for each product (Sales -> Market).
3. Hyperion Essbase calculates Product Share. It takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of total Sales of all products in each market (Sales -> Product).

4. Hyperion Essbase calculates Market %. It takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of the Sales value of the parent of the current member on the Market dimension. It uses the @PARENTVAL function to obtain the Sales value of the parent on the Market dimension.
5. Hyperion Essbase calculates Market %. It takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of the Sales value of the parent of the current member on the Product dimension. It uses the @PARENTVAL function to obtain the Sales value of the parent on the Product dimension.

Allocating Costs Across Products

The following example is based on the Sample Basic database. It allocates overhead costs to each product in each market for each month.

The overhead costs are allocated based on each product's Sales value as a percentage of the total Sales for all products.

Assume that you add two members to the Measures dimension: OH_Costs for the allocated overhead costs and OH_TotalCost for the total overhead costs.

```
/* 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
```

Hyperion Essbase performs these calculations:

1. Hyperion Essbase creates a one-dimensional array called ALLOCQ. The size of ALLOCQ is based on the number of members in the Year dimension. Hyperion Essbase uses ALLOCQ to store the value of Sales as a percentage of total Sales temporarily for each member combination.

2. The `SET AGGMISSG OFF;` command means that `#MISSING` values are not aggregated to their parents. Data values stored at parent levels are not overwritten. If this is your system default, you can omit this line. For more information on setting the default for aggregating `#MISSING` values, see Chapter 51, “Optimizing Calculations.”

   - Hyperion Essbase fixes on the Actual values.
   - Hyperion Essbase cycles through the member combinations for Actual and calculates OH_Costs.
   - It then takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of total Sales for all products in each market (Sales -> Product). It places the result in ALLOCQ.
   - It then takes the total overhead costs for all products (OH_TotalCost -> Product) and multiplies it by the value it has just placed in ALLOCQ. It places the result in OH_Costs.

   Notice that both of the equations are enclosed in parentheses () and associated with the OH_Costs member, OH_Costs(equation1; equation2;). For more information, see Chapter 30, “Developing Calculation Scripts.”

3. Hyperion Essbase calculates and consolidates the Measures dimension.

Allocating Values Within or Across Dimensions

Using the @ALLOCATE and @MDALLOCATE functions, you can allocate values to members in the same dimension or to members in multiple dimensions.

Allocating Within a Dimension

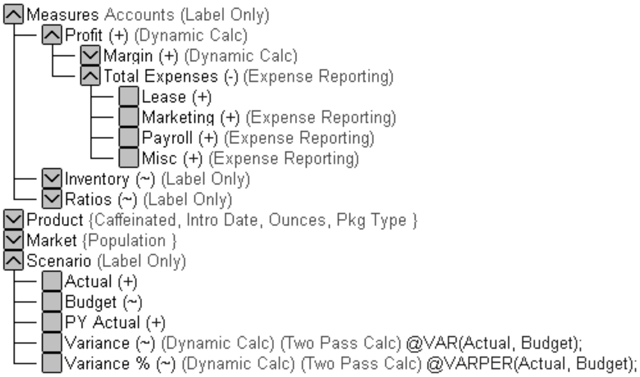
The following example uses the @ALLOCATE function to allocate budgeted total expenses across expense categories for two products. The budgeted total expenses are allocated based on the actual values for the prior year.

Note: For more information on the @ALLOCATE function, see the *Technical Reference* in the `docs` directory.

The following example is based on the Sample Basic database. Assume that you have made the following changes to Sample Basic:

- Added a child, Lease, under Total Expenses in the Measures dimension
- Added a child, PY Actual, to the Scenario dimension
- Removed the Dynamic Calc tag from the Total Expenses member

Figure 473: Modified Measures and Scenario Dimensions from the Sample Basic Database



```
Measures Accounts (Label Only)
    Profit (+) (Dynamic Calc)
        Margin (+) (Dynamic Calc)
        Total Expenses (-) (Expense Reporting)
            Lease (+)
            Marketing (+) (Expense Reporting)
            Payroll (+) (Expense Reporting)
            Misc (+) (Expense Reporting)
    Inventory (~) (Label Only)
    Ratios (~) (Label Only)
Product {Caffeinated, Intro Date, Ounces, Pkg Type }
Market {Population }
Scenario (Label Only)
    Actual (+)
    Budget (~)
    PY Actual (+)
    Variance (~) (Dynamic Calc) (Two Pass Calc) @VAR(Actual, Budget);
    Variance % (~) (Dynamic Calc) (Two Pass Calc) @VARPER(Actual, Budget);
```

For this example, assume that data values of 1000 and 2000 are loaded into Budget -> Total Expenses for Colas and Root Beer, respectively. These values need to be allocated to each expense category, evenly spreading the values based on the non-missing children of Total Expenses from PY Actual. The allocated values need to be rounded to the nearest dollar.

This calculation script defines the allocation:

```
/* Allocate budgeted total expenses based on prior year */
```

```
/* 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
```

This table shows the results:

		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**	**1100**

* Rounding errors are added to this value. See Step 5 for more information.

Hyperion Essbase cycles through the database, performing the following calculations:

1. Hyperion Essbase fixes on the children of Total Expenses. Using a FIX statement with @ALLOCATE may improve calculation performance.
2. For Budget -> Colas -> Marketing, Hyperion Essbase divides 1 by the count of non-missing values for each expense category in PY Actual -> Colas for each month. In this case, 1 is divided by 3, because there are 3 non-missing expense values for Budget -> Colas.
3. Hyperion Essbase takes the value from step 2 (.333), multiplies it by the value for Budget -> Colas -> Total Expenses (1000), and then rounds to the nearest dollar (333). This value is placed in Budget -> Colas -> Marketing.
4. Hyperion Essbase repeats steps 2–3 for each expense category for Budget -> Colas and then for Budget -> Root Beer.
5. As specified in the calculation script, the allocated values are rounded to the nearest whole dollar. Hyperion Essbase makes a second pass through the block to make the sum of the rounded values equal to the allocation value (for example, 1000 for Budget -> Colas -> Total Expenses). In this example, there is a rounding error of 1 for Budget -> Colas -> Total Expenses, because the expense categories add up to 999, not 1000, which is the allocation value. Because all the allocated values are identical (333), the rounding error of 1 is added to the first value in the allocation range, Budget -> Colas -> Marketing (thus a value of 334).

Note: For another example of the @ALLOCATE function, see the *Technical Reference* in the `docs` directory.

Allocating Across Multiple Dimensions

The following example uses the @MDALLOCATE function to allocate a loaded value for budgeted total expenses across three dimensions. The budgeted total expenses are allocated based on the prior year's actual values.

Note: For complete information on the @MDALLOCATE function, see the *Technical Reference* in the `docs` directory.

The following example is based on the Sample Basic database. Assume that you have made the following modifications:

- Added a child, PY Actual, to the Scenario dimension
- Copied data from Actual into PY Actual
- Cleared data from Budget

For this example, a value of 750 (for Budget -> Total Expenses -> Product -> East -> Jan) needs to be allocated to each expense category for the children of product 100 across the states in the East. The allocation uses values from PY Actual to determine the percentage share that each category should receive.

This calculation script defines the allocation:

```
/* 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
```

This table shows the values for PY Actual:

		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

Hyperion Essbase cycles through the database, performing these calculations:

1. Hyperion Essbase fixes on East, the children of 100, and Total Expenses. Using a FIX statement with @MDALLOCATE may improve calculation performance.
2. Before performing the allocation, Hyperion Essbase needs to determine what share of 750 (the value to be allocated) each expense category should receive, for each product-state combination. To do this, Hyperion Essbase uses the shares of each expense category from PY Actual. Starting with PY Actual -> 100-10 -> New York, Hyperion Essbase divides the value for the first expense category, Marketing, by the value for PY Actual-> 100-10 -> East -> Total Expenses to calculate the percentage share of that category. For example, Hyperion Essbase divides the value for PY Actual -> 100-10 -> New York -> Marketing (94) by the value for PY Actual -> 100-10 -> East -> Total Expenses (460), which yields a percentage share of approximately 20.4% for the Marketing category.
3. Hyperion Essbase repeats step 2 for each expense category, for each product-state combination.
4. During the allocation, Hyperion Essbase uses the percentage shares calculated in steps 2–3 to determine what share of 750 should be allocated to each child of Total Expenses from Budget, for each product-state combination. For example, for Marketing, Hyperion Essbase uses the 20.4% figure calculated in step 2, takes 20.4% of 750 (approximately 153), and places the allocated value in Budget -> 100-10 -> New York -> Marketing (see table below).
5. Hyperion Essbase repeats step 4 for each expense category and for each product-state combination, using the percentage shares from PY Actual calculated in steps 2–3.
6. Hyperion Essbase consolidates the expense categories to yield the values for Total Expenses.

This table shows the results of the allocation for Budget:

		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	24.46	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

Note: For another example of the @MDALLOCATE function, see the *Technical Reference* in the `docs` directory.

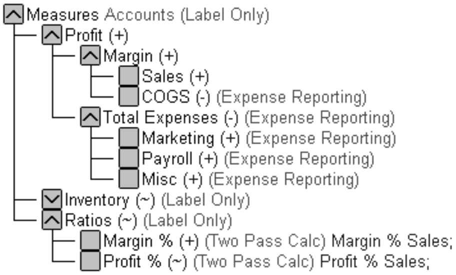
Goal Seeking Using the LOOP Command

The following example is based on the Sample Basic database. However, the example assumes that no members are tagged as Dynamic Calc, and that the Profit per Ounce member (under Ratios in the Scenario dimension) is not included in the calculation. For more information on Dynamic Calc members, see Chapter 28, “Dynamically Calculating Data Values.”

You want to know what sales value you have to reach in order to obtain a certain profit on a specific product.

This example adjusts the Budget value of Sales to reach a goal of 15,000 Profit for Jan. The results are shown for product 100-10.

Figure 474: Measures Dimension from the Sample Basic Database



```
Measures Accounts (Label Only)
    Profit (+)
        Margin (+)
            Sales (+)
            COGS (-) (Expense Reporting)
        Total Expenses (-) (Expense Reporting)
            Marketing (+) (Expense Reporting)
            Payroll (+) (Expense Reporting)
            Misc (+) (Expense Reporting)
    Inventory (~) (Label Only)
    Ratios (~) (Label Only)
        Margin % (+) (Two Pass Calc) Margin % Sales;
        Profit % (~) (Two Pass Calc) Profit % Sales;
```

Assume that the data values before running the goal-seeking calculation script are as follows:

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

This calculation script produces the goal-seeking results:

```
/* 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 and base it 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
```

Hyperion Essbase performs the following calculations:

1. It declares the required temporary variables using the VAR command. Where appropriate, the initial values are set.
2. Hyperion Essbase declares a one-dimensional array called Rollback. The size of Rollback is based on the number of members in the Measures dimension. Hyperion Essbase uses Rollback to store the Budget values.
3. Hyperion Essbase fixes on the Jan -> Budget values for all Product and Market members.
4. The LOOP command ensures that the commands between LOOP and ENDLOOP are cycled through 35 times *for each member combination*. However, if the Quit variable is set to 1, then the LOOP is broken and the calculation continues after the ENDLOOP command.

5. Hyperion Essbase cycles through the member combinations, performing the following calculations:

   a. Hyperion Essbase places the Budget -> Sales value in the Rollback temporary array variable.

   b. It calculates the acceptable error. It multiplies the Target value (15000) by the AcceptableErrorPercent value (0.001) and places the result in the AcceptableError variable.

   c. It retains the current Sales value. It places the Sales value for the current member combination in the PriorVar temporary variable.

   d. It retains the current Profit value. It places the Profit value for the current member combination in the PriorTar temporary variable.

   e. It calculates a new Sales value. It multiplies the PctNewVarChange value (0.1) by the current Sales value, adds the current Sales value, and places the result in Sales.

   f. Hyperion Essbase calculates and consolidates the Measures dimension.

   g. It subtracts the PriorTar value from the PriorVar value and places the result in the DependencyCheck temporary variable.

   h. The IF command checks that DependencyCheck is not 0 (zero).

      - If DependencyCheck is not 0, then Hyperion Essbase subtracts the Target value (15000) from the current Profit and places the result in the CurTarDiff temporary variable.

      The IF command checks to see if the absolute value (irrespective of the + or – sign) of CurTarDiff is greater than the absolute value of the acceptable error (AcceptableError). If it is, Hyperion Essbase calculates the Slope, NxtVar, and PctNewVarChange temporary variables.

      If it is not greater than AcceptableError, Hyperion Essbase breaks the LOOP command by setting the value of Quit to 1. The calculation continues after the ENDLOOP command.

      - If DependencyCheck is 0, Hyperion Essbase places the value in the Rollback array into Budget. Hyperion Essbase breaks the LOOP command by setting the value of Quit to 1. The calculation continues after the ENDLOOP command.

6. Hyperion Essbase calculates and consolidates the Measures dimension.

The results are shown in this table:

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

Forecasting Future Values

The following example uses the @TREND function to forecast sales data for June through December, assuming that data currently exists only up to May. Using the linear regression forecasting method, this example produces a trend, or line, that starts with the known data values from selected previous months and continues with forecasted values based on the known values. In addition, this example demonstrates how to check the results of the trend for “goodness of fit” to the known data values.

Note: For more information about @TREND, see the *Technical Reference* in the `docs` directory.

The following example is based on the Sample Basic database. Assume that the Measures dimension contains an additional child, ErrorLR. The goodness-of-fit results are placed in this member. This calculation script defines the forecasting:

```
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););
```

This table explains each parameter:

@LIST(Jan,Mar,Apr)	Represents the *Ylist*, or the members that contain the known data values. The @LIST function is needed to group the three members as a comma-delimited list and to keep the list separate from other parameters.
@LIST(1,3,4)	Represents the *Xlist*, or the underlying variable values. Since Feb and May are skipped, we need to number the *Ylist* values accordingly (1,3,4).
,	The extra comma after the *Xlist* parameter indicates that a parameter has been skipped, in this case, the *weightList* parameter. The default weight of 1 is used for this example.
@RANGE(ErrorLR, @LIST(Jan,Mar,Apr)	Represents the *errorList*, or the member list where results of the goodness of fit of the trend line to *Ylist* are placed. The values placed in *errorList* are the differences between the data points in *Ylist* and the data points on the trend line produced. The @RANGE function combines the ErrorLR member with *Ylist* (Jan, Mar, Apr) to produce a member list.
@LIST(6,7,8,9,10,11,12)	Represents the *XforecastList*, or the underlying variable values for which the forecast is sought. This example forecasts values consecutively for Jun through Dec, so the values are simply 6,7,8,9,10,11,12.

Jun:Dec	Represents the *YforecastList*, or the member list into which the forecast values are placed. In this example, values are forecast for Jun through Dec based on the values for Jan, Mar, and Apr.
LR	Specifies the Linear Regression method.

Note: For more information on the @LIST and @RANGE functions, see the *Technical Reference* in the `docs` directory.

This table shows the results of the calculation script:

	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

Hyperion Essbase cycles through the database, performing the following calculations:

1. Hyperion Essbase finds the known data values on which to base the trend (Sales for Jan, Mar, Apr), as specified for the *Ylist* and *Xlist* parameters in the calculation script.
2. Hyperion Essbase calculates the trend line using Linear Regression and places the results in Sales for Jun through Dec, as specified for the *YforecastList* parameter in the calculation script.
3. Hyperion Essbase calculates the goodness of fit of the trend line to the data values for Jan, Mar, and Apr and places the results in ErrorLR for those months. For example, the value in ErrorLR for Jan (4.57) means that after Hyperion Essbase calculates the trend line, the difference between the Sales value for Jan (2339) and the Jan value on the trend line is 4.57. The ErrorLR values for Feb and May are `#MISSING` since these months were not part of *Ylist*.

Note: For another example of the @TREND function, see the *Technical Reference* in the `docs` directory.

Chapter 32

Developing Custom-Defined Calculation Macros

This chapter explains how to develop custom-defined macros and how to use them in calculation scripts and formulas. Custom-defined macros are written with calculator functions and special macro functions. Macros enable you to combine multiple calculation functions into a single function.

For more details about the macro language syntax, rules, and examples of its use, see the *Technical Reference* in the `docs` directory.

This chapter includes the following sections:

Understanding Custom-Defined Macros

Custom-defined macros use an internal macro language that enables you to combine calculation functions and operate on multiple input parameters.

When developing and testing custom-defined macros, make sure to create and test new macros locally within a test application. You should register custom-defined macros globally only after you have tested them and are ready to use them as part of a production environment.

Essbase requires that you have a security level of database designer or higher to create and manage custom-defined macros.

Viewing Custom-Defined Macros

You may want to determine whether a macro has been successfully created or whether a custom-defined macro is local or global. You can find out information about existing custom-defined macros using either MaxL or Application Manager.

➤ To view information about custom-defined macros using Application Manager, use this procedure:

1. Start OLAP Server.
2. Start Application Manager and connect to the server.
3. Select Server > Custom-Defined Macros. Essbase displays the Custom Defined Macros Manager, where all the macros you have created are listed.

➤ To view information about custom-defined macros using MaxL, use this procedure:

1. Start OLAP Server.
2. Start MaxL Shell and use the **display macro** statement. For example, to view information about all the custom-defined macros on the Sample application, you would type this statement:

```
MAXL> display macro on application Sample;
```

This statement displays a list of all macros registered for the named application (Sample) and any registered global macros. The **display macro** statement lists global macros without an application name to indicate that they are global. If the application contains a macro with the same name as a global macro, only the local macro is listed.

For more information about similar statements in MaxL, see the *Technical Reference* in the `docs` directory.

Creating Custom-Defined Macros

When you create a custom-defined macro, records the macro definition and stores it for future use. Create the macro once, and then you can use it in formulas and calculation scripts until the macro is updated or removed from the catalog of macros.

Use these sections to understand more about creating custom-defined macros and to find instructions for creating them using MaxL or Application Manager:

- “Understanding Scope” on page 901
- “Naming Custom-Defined Macros” on page 902
- “Creating Macros Using MaxL” on page 902
- “Creating Macros Using Application Manager” on page 903

Understanding Scope

You can create custom-defined macros in two ways: locally or globally. When you create a local custom-defined macro, the macro is only available in the application in which it was created. When you create a global custom-defined macro, the macro is available to all applications on the server where it was created.

When you create a global custom-defined macro, all of your Essbase applications can use it. Be sure you test custom-defined macros in a single application (and create them only in that application) before making them global macros.

CAUTION: Do not create macros as global (using a macro name without the *AppName.* prefix) when testing them, because this makes it more difficult to update them if you find problems. For more information about updating custom-defined macros, see “Changing Custom-Defined Macros” on page 907.

For more detailed information on the MaxL **create macro** grammar, see the *Technical Reference* in the `docs` directory. For rules about naming custom-defined macros, see “Naming Custom-Defined Macros” on page 902.

Naming Custom-Defined Macros

Remember these requirements when you create macro names:

- The names of custom-defined macro must start with an “@” symbol; for example, @MYMACRO. The rest of a macro name can contain letters, numbers, and the following symbols: “@”, “#”, “$”, and “_”. Macro names can not contain spaces.
- The names of custom-defined macros which are only called by other macros should start with “@_” to distinguish them from general use macros and functions.
- Macros must have unique names. Macro names must be different from each other, from the names of custom-defined functions, and from the names of existing calculation functions. If a application contains a local macro that has the same name as a global macro, the local macro takes precedence and is used for calculation.

Creating Macros Using MaxL

➤ To create a custom-defined macro, use this procedure:

1. Start OLAP Server.
2. Start MaxL Shell at the operating system command prompt and log on to the server.
3. Use the **create macro** statement to create the custom-defined macro. The name of the macro is different, depending on whether it is a global macro, or if it is local.
    - For example, you would type this set of statements at the MaxL Shell command prompt to create a macro @COUNTRANGE which is used only in the Sample database:

```
MAXL>create macro Sample.'@COUNTRANGE'(Any) AS
   2>'@COUNT(SKIPMISSING, @RANGE(@@S))'
   3>spec '@COUNTRANGE(MemberRange)'
   4>comment 'counts all non-missing values';
```

- For example, you would type this set of statements at the MaxL Shell command prompt to create a global macro @COUNTRANGE which is used only in the Sample database:

  ```
  MAXL>create macro'@COUNTRANGE'(Any) AS
     2>'@COUNT(SKIPMISSING, @RANGE(@@S))'
     3>spec '@COUNTRANGE(MemberRange)'
     4>comment 'counts all non-missing values';
  ```

Be sure to add the application name plus a period (.) as a prefix before the name of the local macro. In this example, Sample is the prefix for the local macro name. This prefix assigns the macro to an application, so the macro will only be available within that application.

4. Refresh the catalog of custom-defined macros for a single application by issuing this statement:

   ```
   MAXL> refresh custom definition on application sample;
   ```

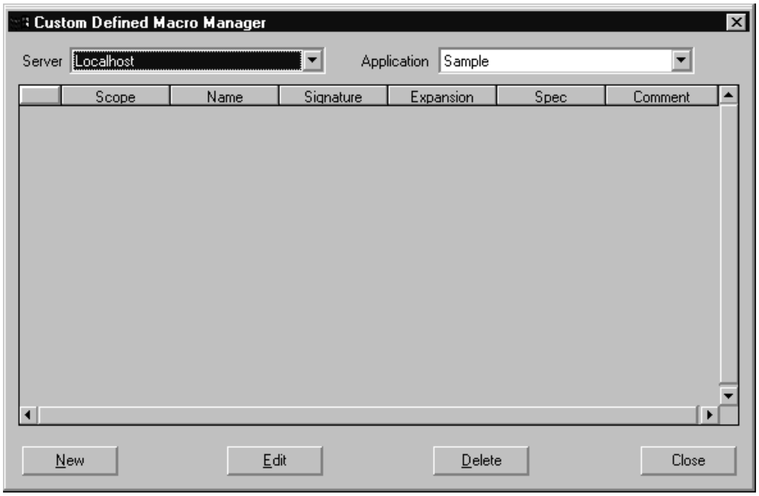
   You can refresh the catalog for all applications on a server by restarting the server.

Creating Macros Using Application Manager

➤ To create a custom-defined macro, global or local, use this procedure:

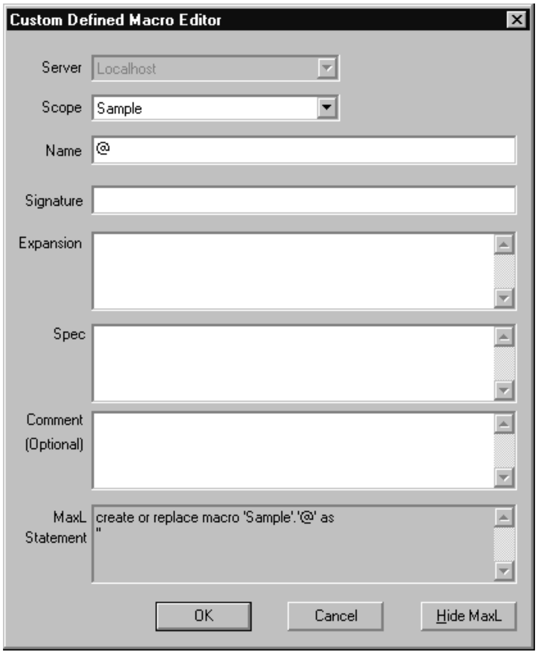
1. Start the OLAP Server.
2. Start the Application Manager and connect to the server.

3. Click Server > Custom-Defined Macros. Essbase displays the Custom Defined Macro Manager:



4. Select a server to which you are connected, where the custom-defined macro will be used. If the custom-defined macro is global, disregard the Application selection box. If the custom-defined macro is not global, select the application where the custom-defined macro will be used.

5. Click New to create a new custom-defined macro. Essbase displays the Custom Defined Macro Editor:



Server Localhost

Scope Sample

Name @

Signature

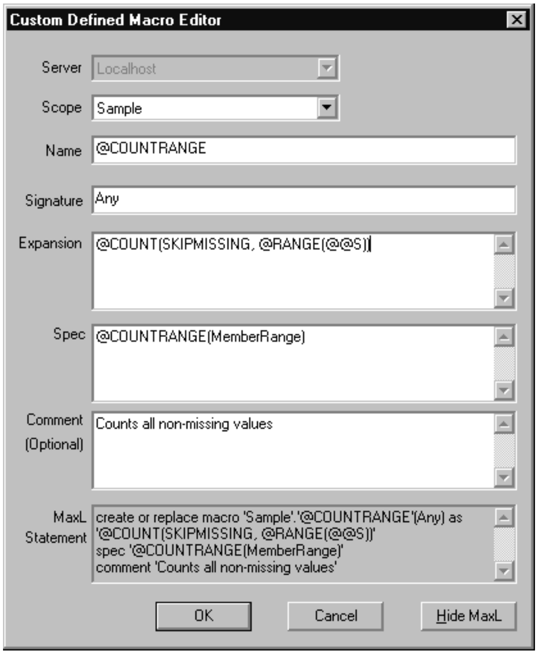
MaxL Statement

```
create or replace macro 'Sample'.'@' as
''
```

6. Select the appropriate scope for the custom-defined macro: <Global> if the custom-defined macro will be available for all applications on the server, or select an application name, such as Sample in this example, if the custom-defined macro will be available for only one application.

7. In the Name text box, leave the “@” symbol as the first character of the custom-defined macro name, and then add the rest of the name, for example COUNTRANGE. Notice that the corresponding MaxL statements which Application Manager uses to create the custom-defined macro are displayed in the MaxL Statement box. Review the contents of this box as you add entries in the editor to ensure accuracy.

8. Add the correct information to the Signature, Expansion, Spec, and optional Comments box:



Server: Localhost

Scope: Sample

Name: @COUNTRANGE

Signature: Any

Expansion: @COUNT(SKIPMISSING, @RANGE(@@S))

Spec: @COUNTRANGE(MemberRange)

Comment (Optional): Counts all non-missing values

MaxL Statement:

```
create or replace macro 'Sample'.'@COUNTRANGE'(Any) as
'@COUNT(SKIPMISSING, @RANGE(@@S))'
spec '@COUNTRANGE(MemberRange)'
comment 'Counts all non-missing values'
```

9. Review the information and when it is correct, click OK to create the macro. Essbase displays the macro in the Custom Defined Macro Manager.

Using Custom-Defined Macros

After creating custom-defined macros, you can use them like native calculation commands. Macros you created locally—using the *AppName.* prefix on the macro name—are only available for use in calculation scripts or formulas within the application in which they were created. If you created the custom-defined macros globally—without the *AppName.* prefix—the macros are available to all calculation scripts and formulas on the server where they were created.

For more information about creating custom-defined macros, see "Creating Custom-Defined Macros" on page 901.

➤ To use a custom-defined macro:

1. Create a new calculation script or formula, or open an existing calculation script or formula.
   - If the custom-defined macro was registered locally—within a specific application—you must use a calculation script or formula within that application.
   - If the custom-defined macro was registered globally, you can use any calculation script or formula on the server.
2. Add the custom-defined macro to a new or existing calculation script or formula. To use the custom-defined macro shown earlier in this chapter, type the following calculation script:

   ```
   CountMbr = @COUNTRANGE(Sales, Jan:Dec);
   ```

   Use this calculation script with the Sample Basic database, or replace "Sales, Jan:Dec" with a range of members in a test database.
3. Save the calculation script or formula, and then run it as usual.

For more information about creating and running calculation scripts, see Chapter 30, "Developing Calculation Scripts." For more information about creating and running formulas, see Chapter 25, "Developing Formulas."

Changing Custom-Defined Macros

When you update a custom-defined macro, you must determine whether the macro is registered locally or globally. There are different procedures for updating macros, depending on whether the macro is local or global, and also depending on which client you use:

- "Changing Custom-Defined Macros Using MaxL" on page 908
- "Changing a Custom-Defined Macro Using Application Manager" on page 909

For information on determining whether a custom-defined macro is local or global, see "Viewing Custom-Defined Macros" on page 900.

Changing Custom-Defined Macros Using MaxL

Local custom-defined macros are created using an *AppName*. prefix in the macro name and can be used only within the application where they were created.

➤ To change a local custom-defined macro using MaxL, use this procedure:

1. Start the server.
2. Start MaxL Shell at the operating system command prompt and log on to the server.
3. Use the **create or replace macro** statement to replace the custom-defined macro. The syntax is slightly different, depending on whether the macro is local or global.

   - For example, you would type this set of statements at the MaxL Shell command prompt to change a macro @COUNTRANGE which is used only in the Sample database:

     ```
     MAXL> create or replace macro Sample.'@COUNTRANGE'(Any)
        2> as '@COUNT(SKIPMISSING, @RANGE(@@S))';
     ```

   - For example, you would type this set of statements at the MaxL Shell command prompt to change a macro @COUNTRANGE which is used in any application on the server:

     ```
     MAXL> create or replace macro '@COUNTRANGE'(Any) as
        2> '@COUNT(SKIPMISSING, @RANGE(@@S))';
     ```

4. Refresh the catalog of custom-defined macros for a single application by issuing this statement:

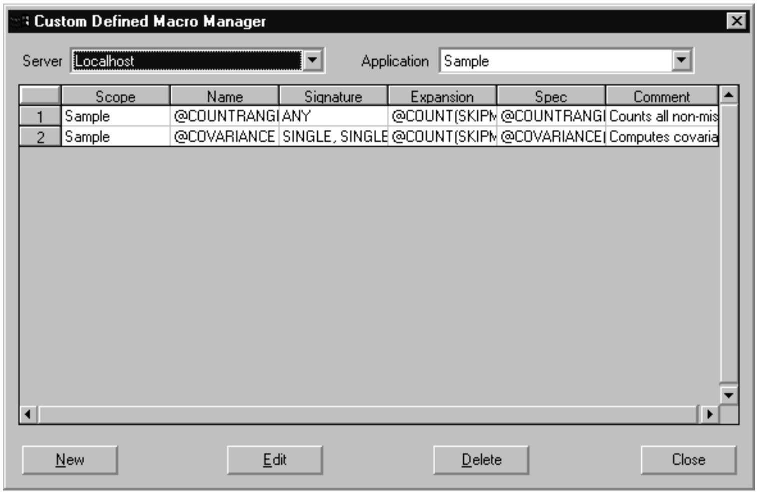
   ```
   MAXL> refresh custom definition on application sample;
   ```

   You can refresh the catalog for all applications on a server by restarting the server.

Changing a Custom-Defined Macro Using Application Manager

➤ To change local or global custom-defined macros using Application Manager, use this procedure:

1. Start the OLAP Server.
2. Start the Application Manager and connect to the server.
3. Click Server > Custom-Defined Macros. Essbase displays the Custom Defined Macro Manager:



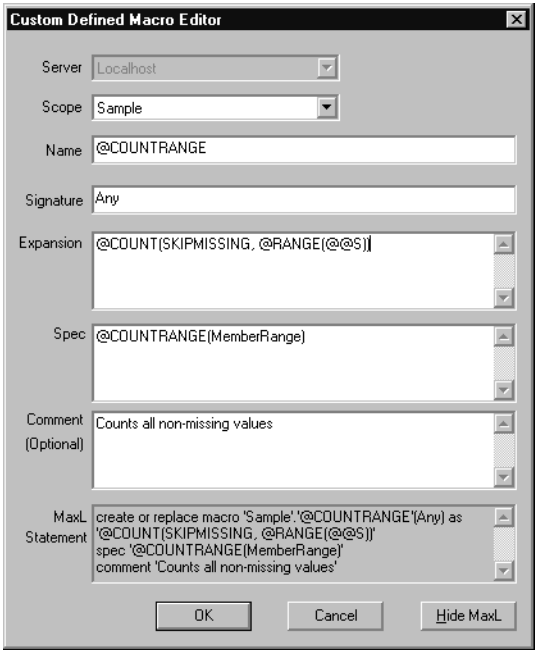
Server: Localhost

Application: Sample

	Scope	Name	Signature	Expansion	Spec	Comment
1	Sample	@COUNTRANGI	ANY	@COUNT(SKIPM	@COUNTRANGI	Counts all non-mis
2	Sample	@COVARIANCE	SINGLE, SINGLE	@COUNT(SKIPM	@COVARIANCE(	Computes covaria

4. Select a server to which you are connected, where the custom-defined macro will be used. If the custom-defined macro is global, disregard the Application selection box. If the custom-defined macro is not global, select the application where the custom-defined macro will be used.
5. Select the macro you want to change.

6. Click Edit to change the custom-defined macro. Essbase displays the Custom Defined Macro Editor:



Server Localhost

Scope Sample

Name @COUNTRANGE

Signature Any

Expansion @COUNT(SKIPMISSING, @RANGE(@@S))

Spec @COUNTRANGE(MemberRange)

Comment (Optional) Counts all non-missing values

MaxL Statement

```
create or replace macro 'Sample'.'@COUNTRANGE'(Any) as
'@COUNT(SKIPMISSING, @RANGE(@@S))'
spec '@COUNTRANGE(MemberRange)'
comment 'Counts all non-missing values'
```

7. Change the macro as needed.
8. Review the information and when it is correct, click OK to save your changes the macro. Essbase displays the macro in the Custom Defined Macro Manager.

Removing Custom-Defined Macros

When removing a custom-defined macro, you must first determine whether the macro is registered locally or globally. The procedure for removing global custom-defined macros is more complex than that for removing local custom-defined macros and should only be performed by a database administrator.

Before removing custom-defined macros, you should verify that no calculation scripts or formulas are using them. Global custom-defined macros can be used in calculation scripts and formulas across a server, so you must verify that no calculation scripts or formulas on the server are using a global custom-defined macro before removing it.

You can use MaxL or Application Manager to remove custom-defined macros:

- "Removing a Custom-Defined Macros Using MaxL" on page 911
- "Removing A Custom-Defined Macro Using Application Manager" on page 912

For information on determining whether a custom-defined macro is local or global, see "Viewing Custom-Defined Macros" on page 900.

Removing a Custom-Defined Macros Using MaxL

➤ To remove a custom-defined macro, use this procedure:

1. Start the server.
2. Start MaxL Shell at the operating system command prompt and log on to the server.
3. Use the **drop macro** statement to remove the custom-defined macro. The syntax is slightly different, depending on whether the macro you are removing is local or global.
   - For example, you would type this set of statements at the MaxL Shell command prompt to remove a macro @COUNTRANGE which is used only in the Sample database:

     ```
     MAXL> drop macro Sample.'@COUNTRANGE';
     ```

   - For example, you would type this set of statements at the MaxL Shell command prompt to remove a macro @COUNTRANGE which is used in any application on the server:

     ```
     MAXL> drop macro '@COUNTRANGE';
     ```

4. Refresh the catalog of custom-defined macros for a single application by issuing this statement:

```
MAXL> refresh custom definition on application sample;
```

You can refresh the catalog for all applications on a server by restarting the server.

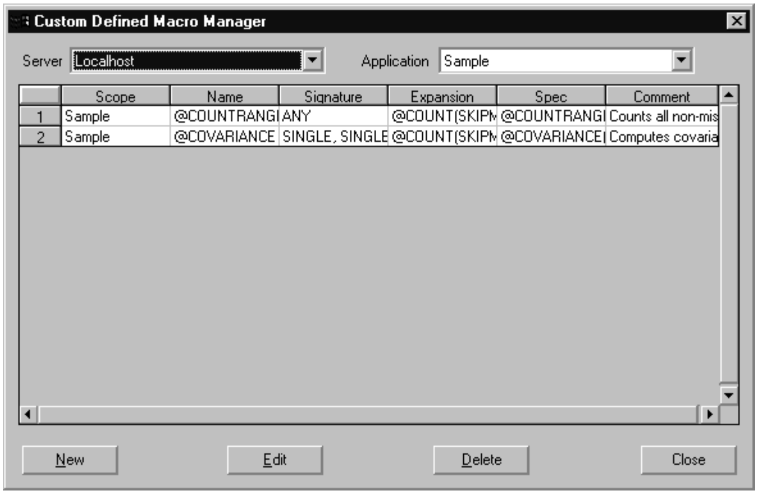
CAUTION: Only a database administrator should remove global custom-defined macros. Removal of global custom-defined macros should only be performed when no users are accessing databases and no calculation routines are being performed.

Removing A Custom-Defined Macro Using Application Manager

➤ To change local or global custom-defined macros using Application Manager, use this procedure:

1. Start the OLAP Server.
2. Start the Application Manager and connect to the server.

3. Click Server > Custom-Defined Macros. Essbase displays the Custom Defined Macro Manager:



Server Localhost

Application Sample

	Scope	Name	Signature	Expansion	Spec	Comment
1	Sample	@COUNTRANGI	ANY	@COUNT(SKIPM	@COUNTRANGI	Counts all non-mis
2	Sample	@COVARIANCE	SINGLE, SINGLE	@COUNT(SKIPM	@COVARIANCE(	Computes covaria

4. Select a server to which you are connected, where the custom-defined macro will be used. If the custom-defined macro is global, disregard the Application selection box. If the custom-defined macro is not global, select the application where the custom-defined macro will be used.
5. Select the macro you want to change.
6. Essbase displays the Custom Defined Macro Manager. The macro you deleted should not be listed.

Chapter 33

Developing Custom-Defined Calculation Functions

This chapter explains how to develop custom-defined functions and use them in Essbase formulas and calculation scripts. Custom-defined functions are written in the Java™ programming language and enable you to create calculation functions not otherwise supported by the Essbase calculation scripting language.

Essbase does not provide tools for creating Java classes and archives. This chapter assumes that you have a compatible version of the Java Development Kit (JDK) and a text editor installed on the computer you will use to develop custom-defined functions. For more information on compatible versions of Java, see the *Essbase Installation Guide*.

For more examples of custom-defined functions, see the *Technical Reference* in the `docs` directory.

Use these sections to create and use custom-defined functions:

Viewing Custom-Defined Functions

You can find out information about existing custom-defined functions using Application Manager:

1. Start OLAP Server.
2. Start Application Manager and connect to OLAP Server.
3. Select Server > Custom-Defined Functions. Essbase displays the Custom Defined Function Manager. You can see a list of all the functions that have been created for the selected server.

No custom-defined functions are displayed until they have been created and registered. Essbase does not supply sample custom-defined functions.

Tip: You can also view information about custom-defined functions using MaxL. For instructions, see "Verifying Custom-Defined Functions" on page 925.

Creating Custom-Defined Functions

There are several steps required to create a custom-defined function:

1. Learn the requirements for custom-defined functions:
   - "Understanding Java Requirements for Custom-Defined Functions" on page 917
   - "Understanding Method Requirements for Custom-Defined Functions" on page 917
   - "Understanding Security and Custom-Defined Functions" on page 918
   - "Understanding Scope and Custom-Defined Functions" on page 918
   - "Naming Custom-Defined Functions" on page 919
2. Write a public Java class that contains at least one public, static method to be used as a custom-defined function. For instructions, see "Creating and Compiling the Java Class" on page 919
3. Install the Java class created in step 2. For instructions, see "Installing Java Classes on OLAP Server" on page 920
4. Register the custom-defined function as a local or global function. For instructions, see "Registering Custom-Defined Functions" on page 921

Understanding Java Requirements for Custom-Defined Functions

The basis of a custom-defined function is a Java class and method created by a database administrator or Java programmer to perform a particular type of calculation. Creating and testing these Java classes and methods is the first step toward creating a custom-defined function.

You can create more than one method in a class for use as a custom-defined function. In general, it is recommended that you create all the methods you want to use as custom-defined functions in a single class. However, if you want to add new custom-defined functions that are not going to be used across all applications on OLAP Server, create them in a new class and add them to OLAP Server in a separate `.jar` file.

When creating multiple Java classes that contain methods for use as custom-defined functions, verify that each class name is unique. Duplicate class names will cause methods in the duplicate class not to be recognized, and you will be unable to register those methods as custom-defined functions.

After creating the Java classes and methods for custom-defined functions, test them using test programs in Java. When you are satisfied with the output of the methods, install them on OLAP Server and register them in a single test application. Do not register functions globally for testing, because this makes it more difficult to update them if you find problems.

Understanding Method Requirements for Custom-Defined Functions

Methods in custom-defined functions can have any combination of the following supported data types as input parameters:

Data Types Allowed As Input Parameters	
boolean	byte
char	java.lang.String
short, int, long	com.hyperion.essbase.calculator.CalcBoolean
float, double	arrays of any of these types

CalcBoolean is an Essbase-specific data type that can include three values: TRUE, FALSE, and #MISSING. For more information about the other listed data types, see the documentation for the Java Development Kit.

The method return data type can be void or any of the preceding data types. Returned data types are converted to Essbase-specific data types. Strings are mapped to a string type. Boolean values are mapped to the CalcBoolean data type. All other values are mapped to a double type.

Note: Double variables returned with infinite or Not-a-Number values are not supported by Essbase. If these values are returned from a Java program, they may not be recorded or displayed correctly in Essbase. Double variables should be checked for infinite or Not-a-Number values and set to finite values before being returned to Essbase. For more information, see the entry for the class, Double, in the documentation for the Java Development Kit.

For more examples of Java custom-defined functions, see the MaxL statement **create function** in the *Technical Reference* in the `docs` directory.

Understanding Security and Custom-Defined Functions

Essbase requires that you have these security permissions:

- To create, delete, and manage local (application) custom-defined functions, you must have security permission of application designer or higher.
- To create, delete, and manage global (server-wide) custom-defined functions, you must have security permission of supervisor.

Understanding Scope and Custom-Defined Functions

Custom-defined functions are registered with one of two scopes: local (application) or global (OLAP Server). When you register a local custom-defined function, it is available only in the application in which it was registered. When you register a global custom-defined function, it is available to all applications on the OLAP Server on which it was registered.

When developing and testing custom-defined functions, make sure to register and test new functions locally within a test application. You should never register custom-defined functions globally until you have thoroughly tested them and are ready to use them as part of a production environment.

Naming Custom-Defined Functions

When you register a custom-defined function in Essbase, you give the function a name. This name is used in calculation scripts and formulas and is distinct from the Java class and method name used by the function.

Remember these requirements when you create function names:

- Custom-defined function names must start with an @ symbol; for example, @MYFUNCTION. The rest of a function name can contain letters, numbers, and the following symbols: @, #, $, and _. Function names can not contain spaces.
- The names of custom-defined functions which are called only by custom-defined macros should start with "@_" to distinguish them from general use functions and macros.
- Custom-defined functions must have unique names. Function names must be different from each other, from the names of custom-defined macros, and from the names of existing calculation functions.
- If an Essbase application contains a local function that has the same name as a global function, the local function is used for calculation.

For more information about registering custom-defined functions, see "Registering Custom-Defined Functions" on page 921.

Creating and Compiling the Java Class

➤ To create a Java class for a custom-defined function, use this procedure:

1. Start a text editor and create a Java class. For example, open a text editor and create this class:

```
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. Save the Java class as a .java file; for example, `CalcFunc.java`.
3. Compile the Java class using the JDK javac tool. At the operating system command prompt, move to the directory containing the class you want to compile and use the javac tool. For example, to compile the `CalcFunc.java` class, type this command:

   ```
   >javac CalcFunc.java
   ```
4. Resolve any compiling errors and repeat steps 2 and 3 until the compiler creates a new .class file; for example, `CalcFunc.class`.

Installing Java Classes on OLAP Server

When you install Java classes on OLAP Server, you must first compile the classes into a Java Archive (`.jar`) file, and then copy the `.jar` file to a specific location on the computer running OLAP Server.

Note: You must either stop and restart Essbase applications or stop and restart OLAP Server when you install new `.jar` files.

➤ To create a `.jar` file from a `.class` file and install it on an OLAP Server:

1. At the operating system command prompt, move to the directory containing the class you want to add to a `.jar` file and use the JDK jar tool. To add `CalcFunc.class` to a `.jar` file, type this command:

   ```
   >jar cf CalcFunc.jar CalcFunc.class
   ```
2. Copy the `.jar` file to one of the following locations on the computer running OLAP Server:
   - The *ARBORPATH*`\java\udf\` directory. This location is recommended for `.jar` files containing custom-defined functions. If this directory does not exist, create the directory.
   - The *ARBORPATH*\app\AppName\udf\ directory where *AppName* is the name of the application where the custom-defined function will be used. If this directory does not exist, create the directory. Use this location if you want the code in the `.jar` file to only be used with a specific application.

   If the `.jar` file is placed in another location, you must modify the CLASSPATH variable to include the full path and file name for the `.jar` file.
3. If the functions will only be used by specific applications, restart those applications in Essbase. Otherwise, restart OLAP Server.

Registering Custom-Defined Functions

After you have compiled the Java classes for custom-defined functions into `.jar` files and installed the `.jar` files on OLAP Server, you must register the custom-defined function before you can use them in calculation scripts and formulas.

When you register a global custom-defined function, all of your Essbase applications can use it. Be sure you test custom-defined functions in a single application (and register them only in that application) before making them global functions.

CAUTION: Do not register global functions for testing, because this makes it more difficult to change them if you find problems.

Registering Custom-Defined Functions Using MaxL

➤ To register a custom-defined function using MaxL, use this procedure:

1. Start OLAP Server.
2. Start MaxL Shell at the operating system command prompt and log on to OLAP Server.
3. Use the **create function** statement to register the custom-defined function. The exact syntax is different, depending on whether the custom-defined function is global or local:

   - To register a custom-defined function with local scope, include the application name as a prefix. For example, to register the custom-defined function in the class CalcFunc from "Creating and Compiling the Java Class" on page 919, as a local function, type this statement:

     ```
     MAXL> create function Sample.'@JSUM'
        2> as 'CalcFunc.sum'
        3> spec '@JSUM(memberRange)'
        4> comment 'adds list of input members';
     ```

     The prefix `Sample.` before the name of the function assigns the custom-defined function to the application Sample, so the function will be available only within that application.

- To register a custom-defined function with global scope, do not include the application name as a prefix. For example, to register the custom-defined function in the class CalcFunc from “Creating and Compiling the Java Class” on page 919, as a global function, type this statement:

  ```
  MAXL> create function '@JSUM'
     2> as 'CalcFunc.sum';
     3> spec '@JSUM(memberRange)'
     4> comment 'adds list of input members';
  ```

  The *AppName*. prefix is not included in the name of the function. The lack of a prefix makes a function global.

Note: Specifying input parameters for the Java method is optional. If you do not specify input parameters, Essbase reads them from the method definition in the Java code. However, if you are registering two or more custom-defined functions with the same method name but with different parameter sets, you must register each version of the function separately, specifying the parameters for each version of the function.

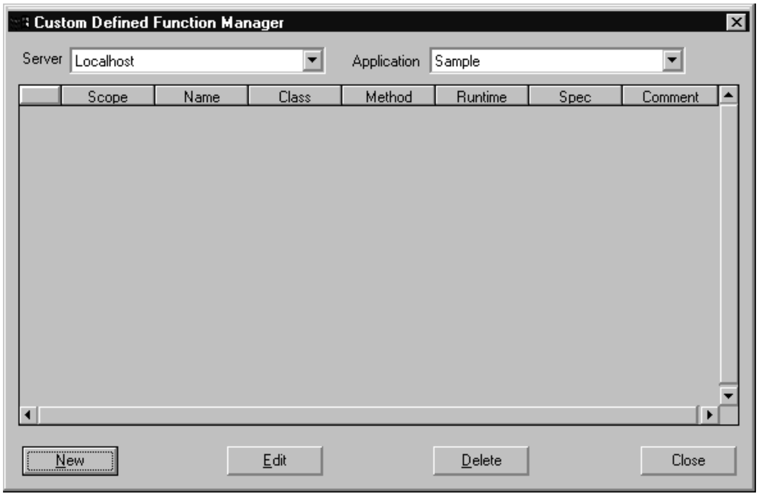
4. Refresh the custom-defined function and macro catalog by using the **refresh custom definitions** statement in MaxL, or by stopping and restarting the application.

For more detailed information on the **create function** statement grammar, see the MaxL section of the *Technical Reference* in the `docs` directory. For rules about naming custom-defined functions, see “Naming Custom-Defined Functions” on page 919.

Registering Custom-Defined Functions Using Application Manager

➤ To register custom-defined functions using Application Manager, use this procedure:

1. Start the OLAP Server.
2. Start the Application Manager and connect to the server.
3. Click Server > Custom-Defined Functions. Essbase displays the Custom Defined Function Manager:



Custom Defined Function Manager

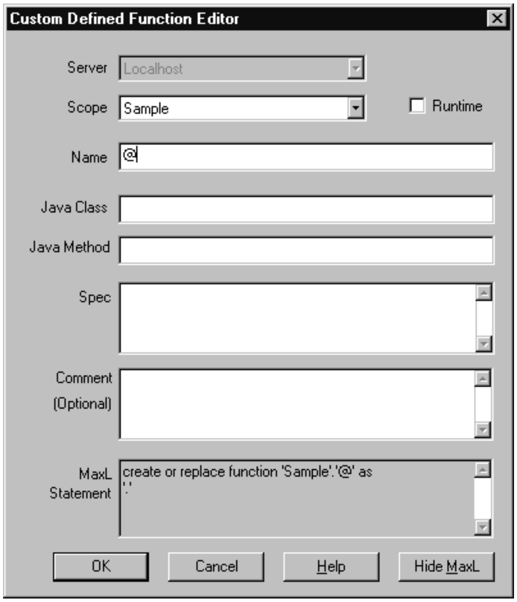
Server: Localhost

Application: Sample

	Scope	Name	Class	Method	Runtime	Spec	Comment

4. Select a server to which you are connected, where the custom-defined function will be used. If the custom-defined function is global, disregard the Application selection box. If the custom-defined function is not global, select the application where the custom-defined function will be used.

5. Click New to register a new custom-defined function. Essbase displays the Custom Defined function Editor:



Custom Defined Function Editor

Server: Localhost

Scope: Sample

□ Runtime

Name: @

Java Class:

Java Method:

Spec:

Comment (Optional):

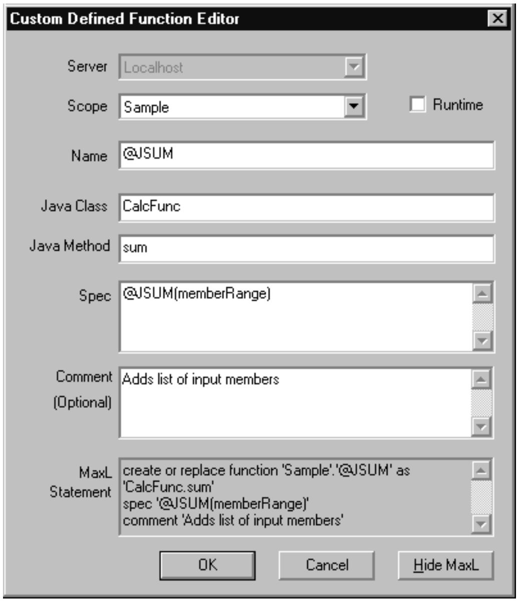
MaxL Statement:

```
create or replace function 'Sample'.'@' as
''
```

6. Select the appropriate scope for the custom-defined function: <Global> if the custom-defined function will be available for all applications on the server, or select an application name, such as Sample in this example, if the custom-defined function will be available for only one application.

7. In the Name text box, leave the “@” symbol as the first character of the custom-defined function name, and then add the rest of the name, for example JSUM. Notice that the corresponding MaxL statements which Application Manager uses to register the custom-defined function are displayed in the MaxL Statement box. Review the contents of this box as you add entries in the editor to ensure accuracy.

8. Add the correct information to the fields in the Custom Defined Function Editor:



Custom Defined Function Editor

Server: Localhost

Scope: Sample

□ Runtime

Name: @JSUM

Java Class: CalcFunc

Java Method: sum

Spec: @JSUM(memberRange)

Comment (Optional): Adds list of input members

MaxL Statement:

```
create or replace function 'Sample'.'@JSUM' as
'CalcFunc.sum'
spec '@JSUM(memberRange)'
comment 'Adds list of input members'
```

This example uses information about the sample class created in “Registering Custom-Defined Functions Using MaxL” on page 921. If you need help filling in the values, review that section.

9. When the information is correct, click OK to register the function. Essbase displays the function in the Custom Defined Function Manager.

Verifying Custom-Defined Functions

After registering custom-defined functions, you can determine whether a function has been registered successfully and whether it is registered locally or globally. You can use MaxL or Application Manager.

➤ To check the registration of custom-defined function on OLAP Server using MaxL, use this procedure:

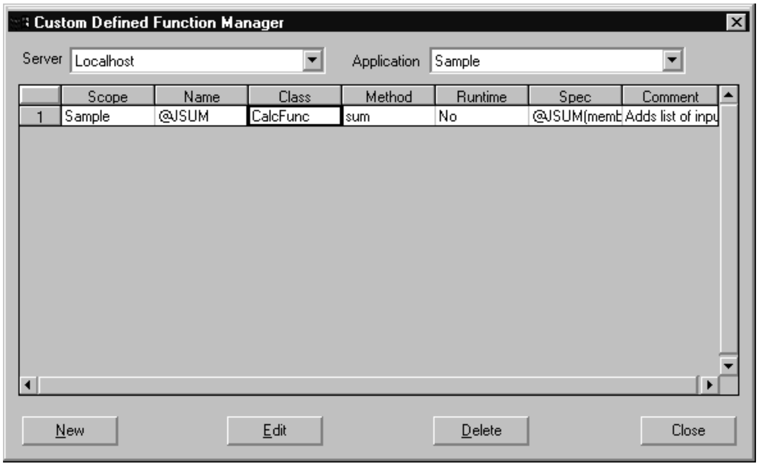
1. Start OLAP Server.
2. Start MaxL Shell at the operating system command prompt and log on to OLAP Server.
3. Use the **display function** statement to check the registration of the custom-defined function; for example, type this statement:

   ```
   MAXL> display function Sample;
   ```

   This statement displays a list of all functions registered for the named application (Sample) and any registered global functions. The **display function** statement lists global functions without an application name to indicate that they are global. If the application contains a function with the same name as a global function, only the local function is listed.

➤ To check the registration of custom-defined functions on OLAP Server using Application Manager, use this procedure:

1. Start OLAP Server.
2. Start the Application Manager and connect to the server.
3. Click Server > Custom-Defined Functions. Essbase displays the Custom Defined Function Manager:



Server: Localhost
Application: Sample

	Scope	Name	Class	Method	Runtime	Spec	Comment
1	Sample	@JSUM	CalcFunc	sum	No	@JSUM(memb	Adds list of inpu

If your function has been successfully registered, it will appear in this list.

Using Registered Custom-Defined Functions

After registering custom-defined functions, you can use them like native Essbase calculation commands. Functions you registered locally—using the *AppName*. prefix on the function name—are only available for use in calculation scripts or formulas within the application in which they were registered. If you registered the custom-defined functions globally, then the functions are available to all calculation scripts and formulas on the OLAP Server where the functions are registered.

For more information about registering custom-defined functions, see "Registering Custom-Defined Functions" on page 921.

➤ To use a registered custom-defined function:

1. Create a new calculation script or formula, or open an existing calculation script or formula.
   - If the custom-defined function was registered locally—within a specific application—then you must use a calculation script or formula within that application.
   - If the custom-defined function was registered globally, then you can use any calculation script or formula on OLAP Server.
2. Add the custom-defined function to a new or existing calculation script or formula. To use the custom-defined function shown earlier in this chapter, use this calculation script:

   ```
   "New York" = @JSUM(@LIST(2.3, 4.5, 6.6, 1000.34));
   ```

   Use this calculation script with the Sample Basic sample database, or replace "New York" with the name of a member in a test database.
3. Save the calculation script or formula, and then run it as usual.

For more information about creating and running calculation scripts, see Chapter 30, "Developing Calculation Scripts." For more information about creating and running formulas, see Chapter 25, "Developing Formulas."

Updating Custom-Defined Functions

When you update a custom-defined function, there are two major issues to consider:

- Is the function registered locally or globally?
- Have the class name, method name, or input parameters changed in the Java code for the custom-defined function?

The answers to these questions determine the procedure for updating the custom-defined function.

For information on determining whether a custom-defined function is local or global, see "Verifying Custom-Defined Functions" on page 925.

To update a custom-defined function, in most cases, you must replace the `.jar` file that contains the code for the function, and then re-register the function. However, if the signature of the custom-defined function—the Java class name, method name and input parameters—have not changed and the function has only one set of input parameters (it is not an overloaded method), you can simply replace the `.jar` file that contains the function. In either situation, you must stop and restart the Essbase application or OLAP Server where the custom-defined function is registered, depending on whether it is a local or global function.

Note: The following procedure is effective only if the signature of the custom-defined function—the Java class name, method name, and input parameters for the custom-defined function—has not changed.

➤ To update a custom-defined function whose signature has not changed:

1. Make the changes to the Java class for the custom-defined function and use Java test programs to test its output.
2. Compile the Java classes and encapsulate them in a `.jar` file, using the same name as the previous `.jar` file. Make sure to include any other classes and methods for custom-defined functions that were included in the previous `.jar` file.
3. Perform one of the following steps:
   - If the function is registered locally, shut down the application in which it was registered.
   - If the function is registered globally, shut down all running applications.

4. Copy the new `.jar` file to OLAP Server over the existing `.jar` file with the same name.
5. Restart the applications you shut down in step 3.

Updating Local Custom-Defined Functions

Local custom-defined functions are registered with an *AppName.* prefix on the function name and can be used only within the application where they were registered. To update a local custom-defined function, you must stop and restart the Essbase application where the function is registered.

➤ To update a local custom-defined function:

1. Make the changes to the Java class for the custom-defined function and use Java test programs to test its output.
2. Compile the Java classes and encapsulate them in a `.jar` file, using the same name as the previous `.jar` file. Make sure to include any other classes and methods for custom-defined functions that were included in the previous `.jar` file.
3. Perform one of the following steps:
   - If the `.jar` file contains local custom-defined functions only, shut down any Essbase applications that use these functions.
   - If the `.jar` file contains any global custom-defined functions, shut down all Essbase applications.
   - If you are unsure as to which Essbase applications use which functions in the `.jar` file you are replacing, shut down all Essbase applications.
4. Copy the new `.jar` file to OLAP Server over the existing `.jar` file with the same name.

5. Start MaxL Shell at the operating system command prompt and log on to OLAP Server.
6. Use the **create or replace function** statement to replace the custom-defined function; for example, type this statement:

```
MAXL> create or replace function sample.'@JSUM'
   2> as 'CalcFunc.sum';
```

Note: This step can also be performed by using the Application Manager Custom Defined Function Manager. Start the server, Application Manager and application, and select File > Custom Defined Function Manager from the menu. Select a function from the list displayed and click Edit to make changes.

Updating Global Custom-Defined Functions

Global custom-defined functions are registered without an *AppName*. prefix on the function name and are available in any application running on the OLAP Server where they are registered. To update a global custom-defined function, you must stop and restart all applications on OLAP Server.

Global custom-defined functions can be used in calculation scripts and formulas across OLAP Server, so you should verify that no calculation scripts or formulas are using a global custom-defined function before updating it.

CAUTION: Only a database administrator should update global custom-defined functions.

➤ To update a global custom-defined function:

1. Make the changes to the Java class for the custom-defined function and use Java test programs to test its output.
2. Compile the Java classes and encapsulate them in a `.jar` file, using the same name as the previous `.jar` file. Make sure to include any other classes and methods for custom-defined functions that were included in the previous `.jar` file.
3. Shut down all running Essbase applications.
4. Copy the new `.jar` file to OLAP Server over the existing `.jar` file with the same name.

5. Start MaxL Shell at the operating system command prompt and log on to OLAP Server.
6. Use the **create or replace function** statement to replace the custom-defined function; for example, type this statement:

```
MAXL> create or replace function '@JSUM'
   2> as 'CalcFunc.sum';
```

Note: This step can also be performed by using the Application Manager Custom Defined Function Manager. Start the server and Application Manager, and select File > Custom Defined Function Manager from the menu. Select a function from the list displayed and click Edit to make changes.

Removing Custom-Defined Functions

When removing a custom-defined function, you must first determine whether the function is registered locally or globally to identify the security permissions required:

- To remove a custom-defined function with global scope, you must have supervisor permission.
- To remove a custom-defined function with local scope, you must have application designer permission for the application, or any wider permission.

Before removing custom-defined functions, you should verify that no calculation scripts or formulas are using them. Global custom-defined functions can be used in calculation scripts and formulas across OLAP Server, so you must verify that no calculation scripts or formulas on OLAP Server are using a global custom-defined function before removing it.

For information on determining whether a custom-defined function is local or global, see "Verifying Custom-Defined Functions" on page 925.

Removing Local Custom-Defined Functions

Local custom-defined functions are registered with an *AppName.* prefix on the function name and can only be used within the application where they are registered. When you remove a local custom-defined function, the Essbase application where the function is registered must be stopped and restarted.

➤ To remove a local custom-defined function:

1. Start OLAP Server.
2. Start MaxL Shell at the operating system command prompt and log on to OLAP Server.
3. Use the **drop function** statement to remove the custom-defined function; for example, type the following statement:

   ```
   MAXL> drop function Sample.'@JSUM';
   ```

   Note: This step can also be performed by using the Application Manager Custom Defined Function Manager. Start the server and Application Manager, and select File > Custom Defined Function Manager from the menu. Select a function from the list displayed and click Delete to remove the custom-defined function.

4. Restart the application where the function was registered. This refreshes the custom-defined function and macro catalog.

Removing Global Custom-Defined Functions

Global custom-defined functions are registered without an *AppName.* prefix on the function name and are available in any application running on OLAP Server where they are registered. When you remove a global custom-defined function, all running Essbase applications must be stopped and restarted.

Global custom-defined functions can be used in calculation scripts and formulas across OLAP Server, so you should verify that no calculation scripts or formulas are using a global custom-defined function before removing it.

CAUTION: Only a database administrator with supervisor permission can remove global custom-defined functions. Removal of global custom-defined functions should only be performed when no users are accessing Essbase databases and no calculation routines are being performed.

➤ To remove a global custom-defined function:

1. Start OLAP Server.
2. Start MaxL Shell at the operating system command prompt and log on to OLAP Server.
3. Use the **drop function** statement to remove the custom-defined function; for example, type the following statement:

   ```
   MAXL> drop function '@JSUM';
   ```

   Note: This step can also be performed by using the Application Manager Custom Defined Function Manager. Start the server and Application Manager, and select File > Custom Defined Function Manager from the menu. Select a function from the list displayed and click Delete to remove the custom-defined function.

4. Restart the application where the function was registered. This refreshes the custom-defined function and macro catalog.

Performance and Memory Considerations for Custom-Defined Functions

The ability to create and run custom-defined functions is provided as an extension to the Essbase calculator framework. When you use custom-defined functions, consider how their use affects memory resources and calculator performance.

Performance Considerations

Because custom-defined functions are implemented as an extension of the Essbase calculator framework, you can expect custom-defined functions to operate less efficiently than functions that are native to the Essbase calculator framework. In tests using a simple addition function running in the Java Runtime Environment 1.3 on the Windows NT 4.0 platform, the Java function ran 1.8 times (about 80%) slower than a similar addition function performed by native Essbase calculation commands.

To optimize performance, limit use of custom-defined functions to calculations that you cannot perform with native Essbase calculation commands, particularly in applications where calculation speed is a critical consideration.

Memory Considerations

Use of the Java Virtual Machine (JVM) and Java API for XML Parsing has an initial effect on the memory required to run Essbase. The memory required to run these additional components is documented in the memory requirements for Essbase. For more information about memory requirements, see the *Essbase Installation Guide*.

Beyond these start-up memory requirements, the Java programs you develop for custom-defined functions sometimes require additional memory. When started, the JVM for Win32 operating systems immediately allocates 2 MB of memory for programs. This allocation is increased according to the requirements of the programs that are then run by the JVM. The default upper limit of memory allocation for the JVM on Win32 operating systems is 64 MB. If the execution of a Java program exceeds the default upper limit of memory allocation for the JVM, the JVM generates an error. For more information about JVM memory management and memory allocation details for other operating systems, see the Java Development Kit documentation.

Considering the default memory requirements of the JVM and the limitations of the hardware on which you run servers, carefully monitor your use of memory. In particular, developers of custom-defined functions should be careful not to exceed memory limits of the JVM when creating large objects within custom-defined functions.

Part

VII

Retrieving Data

Part VI describes how to create reports on the data in OLAP Server databases, including how to create reports quickly, develop report scripts, optimize reports, and export data to other programs:

- Chapter 34, “Quick Start to Report Scripts,” introduces you to the basic concepts behind reports and describes how to create them using the Report Writer.
- Chapter 35, “Developing Report Scripts,” describes how to create complex report scripts, including page layout and formatting information, how to select and sort members, how to restrict and order data values, how to convert data to a different currency, and how to generate reports using the C, Visual Basic, and Grid APIs.
- Chapter 36, “Examples of Report Scripts,” contains detailed examples of report scripts.
- Chapter 37, “Copying Data Subsets and Exporting Data to Other Programs,” describes how to move data from OLAP Server databases to other programs by extracting an output file of the data to move using the Report Writer.

Note: For optimization information, see Chapter 53, “Optimizing Reports and Other Types of Retrieval.”

Chapter 34 Quick Start to Report Scripts

An Essbase report lets you retrieve formatted summaries from a database.

There are several ways that you can report on the data in your database:

- Use the Application Manager's Report Writer to create a report script and run your report, as explained in this chapter.
- Use the Administration Services Report Script Editor to create a report script and run your report. For information about using this Report Script Editor, see *Essbase Administration Services Online Help*.
- Generate database reports through a spreadsheet. You can use report commands in a spreadsheet in Free-Form mode or in Template Retrieval mode. For information on reporting through your spreadsheet interface, see your *Essbase Spreadsheet Add-in User's Guide*.
- Use Essbase APIs. See Chapter 35, "Developing Report Scripts," for more information about using report APIs or the *API Reference* in your `docs` directory for syntax and technical descriptions.
- Use third-party reporting tools such as Crystal Info for Hyperion Essbase (sold separately from Essbase).

Use Report Writer when you need to create large-scale reports consisting of many pages of multidimensional data. Reports of this scale often can exceed the capabilities of even the most robust spreadsheet. You can use Application Manager or Administration Services to create report scripts and run reports, or you can schedule report scripts to run in batch mode, using either the MaxL language interface or the ESSCMD command-line interface. See Chapter 45, "Automating the Production Environment" for information about ESSCMD. See also the *Technical Reference* in the `docs` directory for information about MaxL or ESSCMD.

Report Writer commands let you define formatted reports, export data subsets from an Essbase database, and produce free-form reports.

This chapter provides fundamental information about reports and report scripts, including:

- “Creating a Simple Report Script” on page 938
- “Parts of Report Scripts and Reports” on page 942
- “Planning a Report” on page 946
- “Security and Multiple-User Issues” on page 947
- “Creating and Editing Report Scripts” on page 948
- “Running Report Scripts” on page 956
- “Developing Free-Form Reports” on page 963

For information about creating complex report scripts, see Chapter 35, “Developing Report Scripts.”

Creating a Simple Report Script

When you combine report commands that include page, row, and column dimension declarations with selected members, you have all the elements of a simple report script.

The following step-by-step example of a report script specifies these elements, dimensions, and member selection commands. It includes comments, which document the behavior of the script, and the **!** output command. This example is based on the Sample Basic database, which is supplied with your OLAP Server installation.

➤ To create a simple report script using Application Manager:

1. Start Application Manager, and connect to your OLAP Server.
2. Select the **Sample** application and the **Basic** database, and click the **Report Scripts** button.

   If you do not have the Sample Basic database installed, contact your Essbase administrator.

3. Click the **New** button to open the Report Editor.
4. Type the following information, with the exception of the commented (//) lines, which are for your reference:

```
// 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
    !
```

5. Choose **File > Save**, and type **Myrept1** for the report script object name, and save it on the server (the default).
6. Choose **Report > Run**.

When you run this report against the Sample Basic database, the script produces the following report:

```
                                          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     1,925     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    6,500    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
```

For information about Report Writer command syntax and definitions, see the *Technical Reference* in the `docs` directory. If you are using the Report Editor, choose the Help menu item to consult the *Essbase Application Manager Online Help*.

Tool	Description
MaxL	To run existing report scripts in MaxL, use the **export database** statement. For example, `export database sample.basic using report file ‘C:\Hyperion\Essbase\App\Sample\Basic\asym.rep’ to data file ‘c:\month2.rpt’;`

ESSCMD	To create and run reports in ESSCMD, use the REPORTLINE command to execute a single report line.
	For example,
	`REPORTLINE "<DESCENDANTS Year !"`
	Use the REPORT command to execute one or more report lines.
	For example,
	`REPORT <DESCENDANTS  Year <ICHILDREN Market !`

See the *Technical Reference* in the `docs` directory for information about the MaxL statement and REPORTLINE and REPORT commands. See Chapter 45, “Automating the Production Environment” for information about ESSCMD.

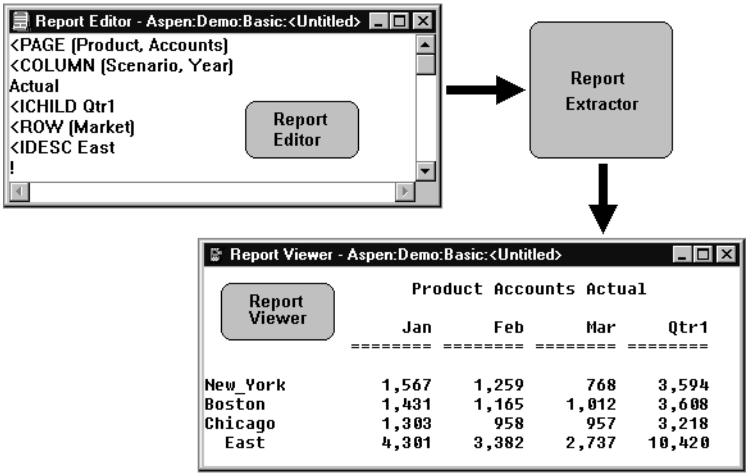
For information about using the Report Script Writer in Administration Services, see the *Essbase Administration Services Online Help*.

Parts of Report Scripts and Reports

The Report Writer consists of three main components:

- *Report Editor* is an ASCII text editor that you use to write the report script. The Report Editor features a text editing window and customized menus. Saved report scripts have the file extension `.REP`.
- *Report Extractor* retrieves the data information from the Essbase database when you run a report script.
- *Report Viewer* displays the complete report. Saved reports have the file extension `.RPT`.

Figure 475: Report Writer Components



Report Editor - Aspen:Demo:Basic:<Untitled>
Report Editor
Report Extractor
Report Viewer - Aspen:Demo:Basic:<Untitled>
Report Viewer

```
<PAGE (Product, Accounts)
<COLUMN (Scenario, Year)
Actual
<ICHILD Qtr1
<ROW (Market)
<IDESC East
!
```

```
                  Product Accounts Actual

                     Jan       Feb       Mar      Qtr1
                ======== ======== ======== ========

New_York           1,567     1,259       768     3,594
Boston             1,431     1,165     1,012     3,608
Chicago            1,303       958       957     3,218
  East             4,301     3,382     2,737    10,420
```

How the Report Extractor Retrieves Data

The Report Extractor processes the report script and retrieves data in the following order:

1. Composes the member list, based on all possible member combinations. For example, the following command retrieves member East and all of its descendants:

   ```
   <IDESCENDANTS East
   ```

2. Applies member restrictions. For example, the following command refines the member selection:

   ```
   <LINK
   ```

3. Orders the member output. For example, the following command determines the order in which members are sorted:

   ```
   <SORT
   ```

4. Extracts data from the following areas:

   - Local regions
   - Partitioned regions
   - Dynamically calculated data

5. Restricts data. For example, the following command suppresses the display of all rows that contain only missing values:

   ```
   {SUPMISSINGROWS}
   ```

6. Sorts data. For example, the following command returns rows with the highest values of a specified data column:

   ```
   <TOP
   ```

7. Formats output. For example, the following command skips one or more lines in the final output report:

   ```
   {SKIP}
   ```

The order in which the Report Extractor retrieves data is important when using complex extraction and formatting commands. For example, because the Extractor restricts data (step 5) before sorting data (step 6), if you place conditional retrieval commands in the wrong order, the report output results could be unexpected. Be aware of the data retrieval process when designing your report scripts.

Parts of a Report

Understanding the parts of a report is essential as you plan and design your own reports. A typical report is composed of the following parts:

- Page Headings list dimensions represented on the current page. All data values on the page have the dimensions in the page heading as a common property.

  ```
  <PAGE (Market, Measures)
  ```

- Column Headings list members across a page. You can define columns that report on data from more than one dimension, which results in *nested column headings*.

  ```
  <COLUMN (Year, Scenario)
  ```

- Row Headings list members down a page. You can define a member list that includes rows from more than one level within a dimension or from more than one dimension. The rows are indented below the dimension name.

  ```
  <ROW (Product)
  ```

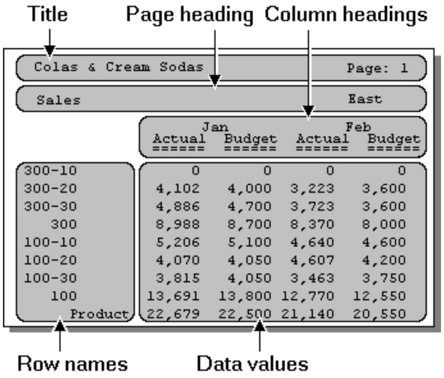
- Titles contain user-defined text, date and time stamp, the user ID of the person running the report, page numbers, the name of the source database, or any other descriptive information. Titles are user-generated and optional, whereas page, column, and row headings are automatically generated, because they are necessary to clearly describe the data on the report page.

  ```
  { STARTHEADING
  TEXT   1 "Prepared by:"
        14 "*USERNAME"
         C "The Electronics Club"
        65 "*PAGESTRING"
  TEXT  65 "*DATE"
  SKIP
  ENDHEADING }
  ```

- Data values are the values contained in the database cells; they are the lookup results of member combinations or the results of calculations when the report is run through the Report Extractor. Each data value is the combination of the members in the page heading, column heading, and row name.

  All data values in a row share the properties of the row names of that row. A report can have zero or more row name dimensions, each of which produces column of row names, with the innermost row name column cycling the fastest.

Figure 476: Elements of a Typical Report



```
 Colas & Cream Sodas                  Page: 1

 Sales                                  East

                     Jan                 Feb
              Actual   Budget   Actual   Budget
              ======   ======   ======   ======
300-10             0        0        0        0
300-20         4,102    4,000    3,223    3,600
300-30         4,886    4,700    3,723    3,600
   300         8,988    8,700    8,370    8,000
100-10         5,206    5,100    4,640    4,600
100-20         4,070    4,050    4,607    4,200
100-30         3,815    4,050    3,463    3,750
   100        13,691   13,800   12,770   12,550
       Product 22,679   22,500   21,140   20,550
```

Parts of a Report Script

A report script consists of a series of Report Writer commands, terminated by the bang (**!**) report output command.

You can enter one or more report scripts in a report script file. A report script file is an ASCII text file that you create with the Report Editor or any text editor.

To build a report script, enter commands that define the layout, member selection, and format in the Report Editor.

The commands in Report Writer perform two functions, data extraction and formatting:

- Extraction commands deal with the selection, orientation, grouping, and ordering of raw data extracted from the database. These commands begin with less than signs (<).
- Formatting commands allow for customization of the report format and appearance, the creation of new columns, and calculation of columns and rows. These commands are generally contained within braces ({}), although some begin with less than signs (<).

Additionally, the *bang* character (**!**) terminates a series of commands and requests information from the database. You must terminate a report script with a bang character, or you can use several bang characters within the script. See Chapter 53, "Optimizing Reports and Other Types of Retrieval," for more information about the **!** character.

See the *Technical Reference* in the `docs` directory for detailed information about the various report commands that you can use.

Planning a Report

Report design is an important part of presenting your information. Designing a report is easy if you include the proper elements and arrange information in an attractive, easy-to-read layout.

➤ To plan your report:

1. Consider your reporting needs and the time it will take to generate the report.
2. Make a rough sketch of the report that includes:
    - Report layout
    - Number of columns
    - Members to include
    - Titles, if applicable
    - Format of the data values

3. Review the sketch; if you need to add additional data or formatting to the report, this is often apparent at this stage.
4. Determine ways to optimize the run time of the report. See Chapter 53, "Optimizing Reports and Other Types of Retrieval," for suggestions about optimizing your report script.

Note: As you plan your report, minimize your use of numeric row names. To avoid ambiguity, give the rows names that describe their content.

Security and Multiple-User Issues

Because you run the Report Editor from the Application Manager menu, you must have access to the Application Manager in order to use the Report Editor to create or modify a report script. You can also use any text editor to create script files. If you use the Application Manager's Report Editor, it lets you create and modify report scripts stored on your desktop machine, as well as the OLAP Server. To modify report scripts stored on the server, you must have Application Designer or Database Designer access.

Essbase supports concurrent, multiple-user database access. As in most multiple-user environments, Essbase protects your critical data with a security system. Users can read or update data only if they have the correct privileges.

When you execute a report script, the Essbase security system verifies that you have Read or higher access level to all data members specified in the report. In a filtering process identical to the one for retrieving members into a spreadsheet, Essbase filters any member from the output for which you have insufficient privileges.

To users who are only reporting data, locks placed by other users are transparent. Even if a user has locked and is updating part of the data required by your report, the lock does not interfere with your report in any way. The data in the report reflects the data in the database at the time you run the report. Running the same report later reflects any changes made after your last report ran.

See Chapter 15, "Managing Security for Users and Applications" for more information about the Essbase security system.

Creating and Editing Report Scripts

You can create your report script using the Report Editor in Application Manager, the Report Script Editor in Administration Services, or with any ASCII text editor. For more information about using the Report Script Editor in Administration Services, see *Essbase Administration Services Online Help*.

Once you create your script, you can choose to save it to either OLAP Server or your desktop machine.

You can modify your script using text editing features that let you cut, copy, paste, find, and replace. You can also undo most commands and changes with the Edit > Undo command.

The Report Editor uses familiar text editing commands, such as Edit > Cut, Edit > Copy, and Edit > Paste. These commands are available from the Application Manager menu or through accelerator key (hot key) combinations.

Creating New Report Scripts

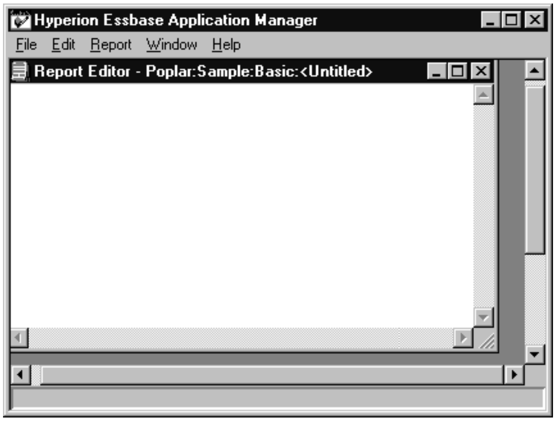
Before you can create a report script, you must connect to an OLAP Server and open a database.

➤ To create a new report script:

1. Choose **File > New > Report Script** from the Application Manager menu.

   The Report Editor displays.

   Figure 477: New Report Editor Window


Hyperion Essbase Application Manager
File
Edit
Report
Window
Help
Report Editor - Poplar:Sample:Basic:<Untitled>

2. Begin typing your report script.

   The name of the report script is Untitled when it is first displayed. When you save the script, the Application Manager prompts you for a file name.

Saving Report Scripts

You can save a report script as:

- A file on your client machine.
- An object on OLAP Server. If you want other users to have access to the report script, you need to save it on the OLAP Server. You can associate the script object with:
  - An application and all the databases within the application, which lets you run the script from any of the databases in the application.
  - A database, which lets you run the script from the specified database.

Report scripts have a `.REP` extension by default. If you run a report script from the Application Manager it must have a `.REP` extension.

➤ To save a report script:

1. Choose **File > Save**.
2. Choose to save the script on OLAP Server or to your desktop machine.

➤ To save the script on the OLAP Server:

1. Click the **Server** button in the **Location** box.
2. Type the name to give to the file in the **Object Name** box.
3. Choose the server where you want to save the script in the **Server** list box.
4. Choose the application where you want to save the script in the **Application** list box.
5. Choose the database where you want to save the script in the **Database** list box, or choose “(all dbs)” to make the report script available to all databases within the chosen application.
6. Choose **Report Scripts** in the **List Files of Type** box.
7. Click **OK**.

➤ To save the script on your desktop machine:

1. Type the file name in the **Object Name** box.
2. Choose the application where you want to save the script in the **Application** list box.
3. Choose the database where you want to save the script in the **Database** list box, or choose "(all dbs)" to make the report script available to all databases within the chosen application.
4. Choose **Report Scripts** in the **List Files of Type** box.
5. Click **OK**.

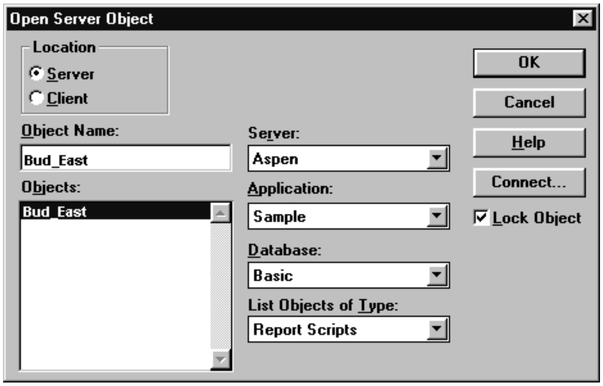
   By default, the file is saved to the `\ESSBASE\CLIENT\SAMPLE` directory on your desktop machine.

Opening Report Scripts from the Application Manager

➤ To open a report script from the Application Manager:

1. From the Application Manager menu, choose **File > Open**.

   Figure 478: Open Server Object Dialog Box



Open Server Object

Location
● Server
○ Client

Object Name: Bud_East

Objects: Bud_East

Server: Aspen

Application: Sample

Database: Basic

List Objects of Type: Report Scripts

☑ Lock Object

2. From the Location group, select the storage location for the report script.
   - Click the **Server** option to search the OLAP Server.
   - Click the **Client** option to search your desktop machine. The dialog box title changes to **Open Client Object**.

3. From the **List Objects of Type** list box, choose **Report Scripts**.

   The Objects list box displays the names of all available report scripts in the selected application and database.

   Note: To choose another application to search, select it from the Application list box. To choose another database to search, select it from the Database list box.

4. You can also locate a locally stored report script through the Windows file system. Click the **File System** button in the **Client Object** dialog box.

   The **Open Client File** dialog box is displayed. The file name mask `*.REP` is displayed in the **File Name** text box and all report script files in the current directory are displayed in the list box.

5. To search other directories, use the **Directories** list box.

6. When the report script you want to open is displayed in the **Objects** list of the **Open Server Object**, **Open Client Object**, or **Open Client File** dialog box, you can:

   - Click the report script name. When the name is displayed in the **Object Name** text box, click **OK**.
   - Enter the name in the **Object Name** text box and click **OK**.
   - Double-click on the report name in the **Objects** list.

   The Application Manager starts the Report Editor and loads the selected report script.

Opening Report Scripts from an Application or Database Directory

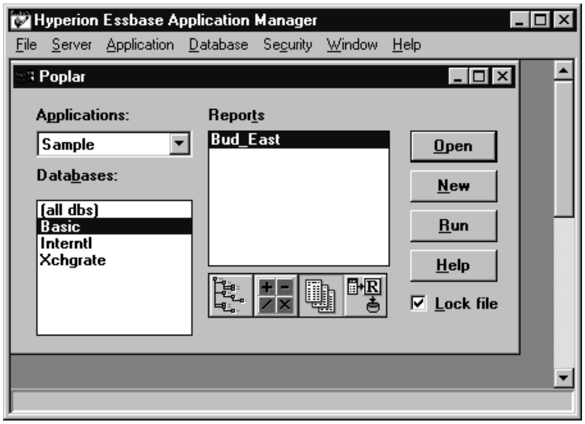
You can open report scripts stored in applications and databases from the application directory.

➤ To open a report script from an application folder:

1. Double-click the application name to open the application folder and select a database from the **Databases** list that contains a report script, or select “(all dbs)” from the Databases list if you want to open a report script stored at the application level.
2. Click the **Report Scripts** button.

3. From the **Reports** list, choose the report script you want to open.
4. Click **Open** to start the Report Editor and load the selected report script.

Figure 479: Database with a Report Script



Applications: Sample

Databases: (all dbs), Basic, Interntl, Xchgrate

Reports: Bud_East

☑ Lock file

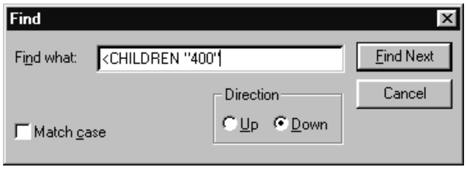
Finding Text

You can use the Find menu command to search for text in the Report Editor. Find locates all occurrences of the specified text in your report script.

➤ To find text in a report script:

1. Choose **Edit > Find** to open the **Find** dialog box.

Figure 480: Find Dialog Box



Find what: <CHILDREN "400"

□ Match case

Direction: ○ Up ● Down

2. Type the text you want to find in the **Find what** text box.

3. Use the **Direction** group box to choose the search direction.

   - Click the **Down** option button to search from the cursor position forward to the end of the report script.
   - Click the **Up** option button to search backward from the cursor position to the beginning of the report script.

   For example, Figure 480 shows a dialog box that finds the next occurrence of <CHILDREN “400” between the current cursor position and the end of the report script.

4. If you want the search to be case-sensitive, check the **Match case** check box. The search matches uppercase and lowercase letters exactly as they are displayed in the **Find wha**t text box.

   In the example, the Match case check box is checked, so the search finds <CHILDREN “400” but not <Children “400”.

5. Click **Find Next**.

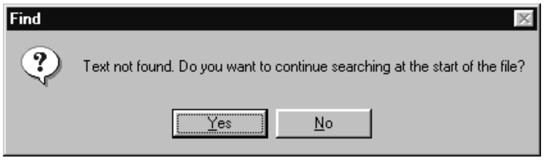
   If the search finds the text you entered in the **Find wha**t text box, it highlights the text in the Report Editor.

6. To edit the selection, click **Cancel** to close the **Find** dialog box.

7. To search for other occurrences of the text, click **Find Next** again.

   When Application Manager reaches the end of the report script, or if it cannot find the selected text to replace, the dialog box shown in Figure 481 is displayed:

   Figure 481: Continue Find Dialog Box



Text not found. Do you want to continue searching at the start of the file?

   - Click **Yes** to continue searching the file.
   - Click **No** to close the dialog box and return to the **Find** dialog box.

8. When you are finished, click **Cancel** to close the **Find** dialog box.

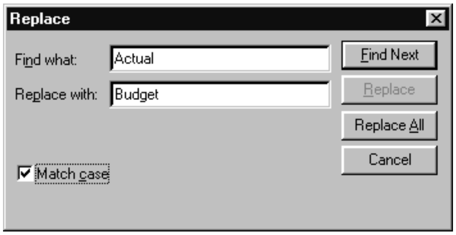
Replacing Text

The Replace menu command replaces one or all instances of specified text with different text.

➤ To replace text in a report script:

1. Choose **Edit > Replace** to open the **Replace** dialog box.

Figure 482: Replace Dialog Box



Find what: Actual

Replace with: Budget

☑ Match case

2. Enter the text you want to find in the **Find what** text box.

   If you want the search to be case-sensitive, check the **Match case** check box. The search matches uppercase and lowercase letters exactly as they are displayed in the **Find what** text box.

   In the example, the “Match case” check box is checked, so the search finds Actual but not ACTUAL.

3. Enter the text in the **Replace with** box that will replace one or all instances of the contents of the **Find wha**t box.

4. Click **Find Next**.

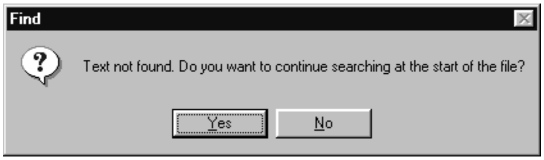
   If the search finds the text you entered in the “Find what” text box, it highlights the text in the Report Editor.

5. Click **Replace** to replace the highlighted text with the information in the **Replace with** box.

6. Click **Replace All** to replace all instances of the text in the **Find what** box with the information in the **Replace with** box.

   When the search reaches the end of the report script, or if it cannot find the selected text to replace, the dialog box shown in Figure 483 is displayed:

   Figure 483: Continue Find Dialog Box

   Find

   Text not found. Do you want to continue searching at the start of the file?

   - Click **Yes** to continue the searching the file.
   - Click **No** to close the dialog box and return to the **Replace** dialog box.

7. When you are finished, click **Cancel** to close the **Replace** dialog box.

Cutting, Copying, and Pasting Text

Cutting, copying, and pasting are basic text editing features that you use to modify your report script. These features work with the desktop Clipboard, allowing you to take information from the Report Editor (by copying or cutting) and then bring it back into the Report Editor (by pasting). You can copy, cut, and paste between different scripts in the Report Editor window. For example, you can copy text from Script1, open Script2, and paste the text into Script2.

➤ To copy text:

1. In the Report Editor window, select the text to copy.
2. Choose **Edit > Copy**.

   The text is still displayed in the Report Editor window, and a copy of the information is stored in the desktop Clipboard.

3. In the **Report Editor** window, place your cursor where you want the text to be displayed.
4. Choose **Edit > Paste**.

The Paste command is disabled if there is nothing stored in the Clipboard.

➤ To cut text:

1. In the **Report Editor** window, select the text to cut.
2. Choose **Edit > Cut**.

   The text is removed from the Report Editor window, and is stored in the desktop Clipboard.
3. In the **Report Editor** window, place your cursor where you want the text to be displayed.
4. Choose **Edit > Paste**.

The Paste command is disabled if there is nothing stored in the Clipboard.

Deleting Text

➤ To remove text from the Report Editor:

1. In the **Report Editor** window, select the text to delete.
2. Choose **Edit > Delete**, or use the **Delete** key on your keyboard.

Running Report Scripts

The Report menu in Report Editor features commands that you can use to choose a destination for the report, select a database against which to run the report script, and run the report script to generate a final report. The following sections describe these commands.

Choosing the Report Output

The Output Options menu command opens the Report Output Options dialog box that lets you choose three output destinations for your report:

- Window displays your report in a window using the default system font.
- Printer sends your report to the currently selected printer and lets you choose the font to use for the report. If Window is also selected, window output is displayed in the same font.
- File sends your report to a `.RPT` file.

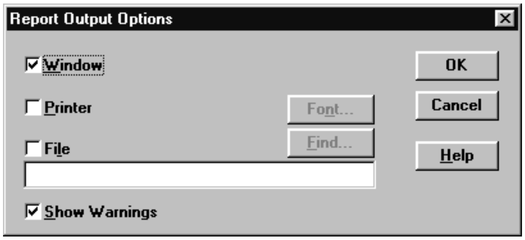
You can select as many output options as you like, but you must select at least one.

Sending the Report Output to a Window

➤ To send your report output to a window:

1. Choose **Report > Output Options** from the Report Editor.

   Figure 484: Report Output Options Dialog Box with Window Check Box Selected

   The **Window** check box is selected as the default.

   If you also check the **Printer** check box, you can select a printer font for Essbase to use as the screen font. See "Sending Report Output to a Printer" on page 958 for details.

2. Notice that the **Show Warnings** check box is selected as the default. This option displays any warning messages about report processing in a window on the screen. If you want to turn off this display, click the check box.

3. To display your report in a window, click **OK**.

   The **Report Output Options** dialog box closes. You are ready to select a database and run the report.

Note: You must check at least one output option to run a report, and you can check any combination of output options.

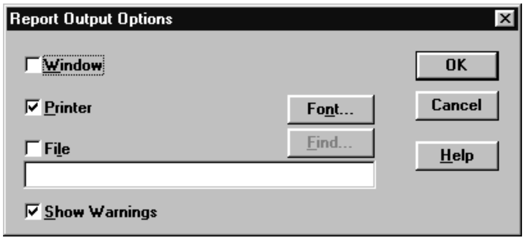
Sending Report Output to a Printer

➤ To send your report output to a printer:

1. Choose **Report > Output Options** from the **Report Editor** menu.
2. Click the **Window** check box to clear it if you do not want the report to be displayed in a window.
3. Click the **Printer** check box to select it.

   Application Manager enables the **Font** button as shown in Figure 485.

   Figure 485: Report Output Options Dialog Box with Printer Check Box Selected

   Report Output Options

   ☐ Window

   ☑ Printer

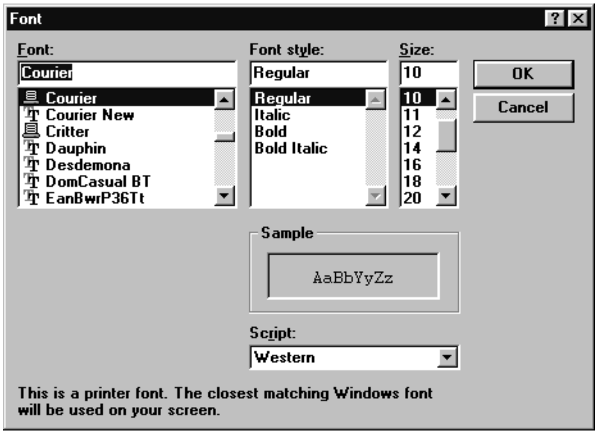
   ☐ File

   ☑ Show Warnings

4. Click the **Font** button to select a printer font for the report.

   The fonts, font styles, and font sizes available on your system are shown in the **Font** dialog box.

   Figure 486: Font Dialog Box

   Font

   Font: Courier

   Font style: Regular

   Size: 10

   Script: Western

   This is a printer font. The closest matching Windows font will be used on your screen.

5. From the **Font** list, select the font type to use in the report.
6. If you want to change the style of the font, choose a different style from the **Font Style** list.
7. From the **Size** list, choose the point size of the font. A sample of the font is displayed in the **Sample** box.
8. When you have finished configuring font options, click **OK**.

   The **Font** dialog box closes, and you return to the **Report Output Options** dialog box.

   In the **Report Output Options** dialog box, the **Show Warnings** check box is selected as the default. This option displays any warning messages about report processing in a window on the screen.
9. If you want to turn off this display, click the check box.
10. Click **OK** to prepare the report for the printer.

    The **Report Output Options** dialog box closes. You are ready to select a database and run the report.

Note: You must check at least one output option to run a report, and you can check any combination of output options.

Sending Report Output to a File

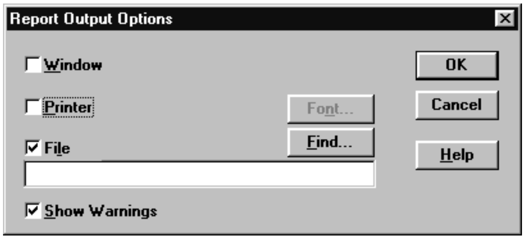
➤ To send your report output to a file:

1. Choose **Report > Output Options** from the **Report Editor**.
2. Click the **Window** check box to clear it if you do not want the report to be displayed in a window.

3. Click the **File** check box to select it.

   Application Manager enables the **Find** button.

   Figure 487: Report Output Options Dialog Box with File Check Box Selected

   Report Output Options

   ☐ Window

   ☐ Printer

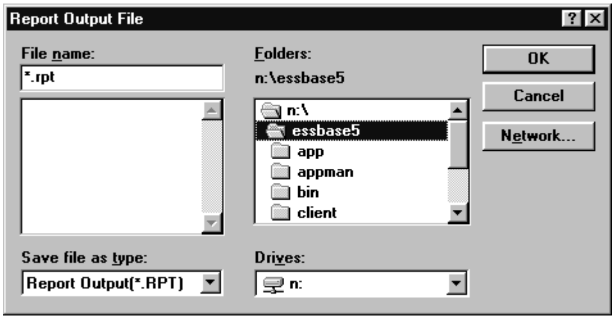
   ☑ File

   ☑ Show Warnings

4. If you want to save the report to your desktop file system, click Find to choose a directory location.

   The file name mask for report output, `*.RPT`, is displayed in the **File name** text box. Any report scripts already in the current directory are displayed in the list box.

   Figure 488: Report Output File Dialog Box

   Report Output File

   File name: *.rpt

   Folders: n:\essbase5

   n:\
   essbase5
   app
   appman
   bin
   client

   Save file as type: Report Output(*.RPT)

   Drives: n:

5. In the **File name** text box, enter a name for the report script, while retaining the `.RPT` extension.

   Report script files follow the same naming conventions as DOS files.

6. In the **Folders** list box, select the directory location for the report file.

7. Click **OK** to save the name and location.

   The **Report Output File** dialog box closes and you return to the **Report Options** dialog box. The **File** text box displays the file name and directory location of the report script.

8. Click **OK** to prepare the report script.

   The **Show Warnings** check box is selected as the default. This option displays any warning messages about report processing in a window on the screen.

9. If you want to turn off this display, click the check box.

   The R**eport Output Options** dialog box closes. You are ready to select a database and run the report.

   Note: You must check at least one output option to run a report, and you can check any combination of output options.

Choosing Report Databases

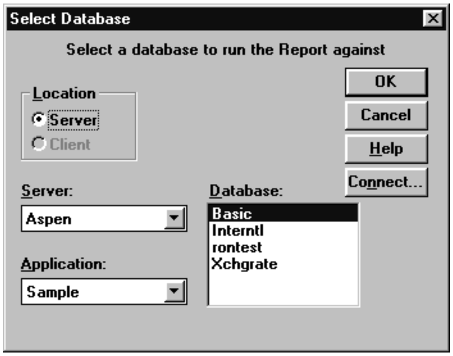
The Select Database dialog box lets you select a database against which to run your report script.

➤ To choose a report database:

1. Choose **Report > Database** from the **Report Editor**.

   The **Select Database** dialog box is displayed.

   Figure 489: Select Database Dialog Box



Select Database

Select a database to run the Report against

Location
● Server
○ Client

Server: Aspen

Application: Sample

Database:
Basic
Interntl
rontest
Xchgrate

2. Click the **Server** or **Client** option to specify the location of the database.

3. From the **Server** list box, select the server where the database resides.

   If there are no server names in the list, click the **Connect** button to log onto the server.

4. Select the application where the database resides from the **Application** list box.

   Application Manager lists the databases that reside on the selected server and in the selected application.

5. Select the database from the **Database** list.
6. Click **OK**.

When you run the report, it reports on the selected database.

Note: If the report is already associated with a database or application other than the one you specify, the Select > Database command does not change that association but simply overrides the default database for this Report Editor session only.

Running the Report

The Report > Run menu command sends your report to the screen and any other output destination you have selected. See "Choosing the Report Output" on page 956 to learn how to choose a report output.

To run the report script, choose **Report > Run**.

Running Multiple Report Script Files

You can use the RUNREPT command in ESSCMD to run one or more report script files. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Executing a Report String Without Creating a Report Script

You can use the following ESSCMDs to run one or more report strings.

To run a single report string, use the REPORTLINE command. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, “Automating the Production Environment” for information about ESSCMD.

To run multiple report strings, use the REPORT command. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, “Automating the Production Environment” for information about ESSCMD.

Developing Free-Form Reports

Free-form reports are often easier to create than structured reports. The free-form reporting style is ideal for ad hoc reporting in the Application Manager’s Report Editor window.

A free-form report does not include PAGE, COLUMN, or ROW commands and instead gathers this information from a series of internal rules that are applied to the report script by the Report Extractor when you run the report.

The following example script and report illustrate free-form reporting:

```
Sales Colas
Jan Feb Mar
Actual Budget
Illinois
Ohio
Wisconsin
Missouri
Iowa
Colorado
{UCHARACTERS}
Central
     !
```

This example produces the following report:

```
                              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
```

You can use formatting commands to add specific formats to a free-form report. The rest of the report is automatically produced in a format similar to that of any other report. When PAGE, COLUMN, and ROW commands are omitted, Essbase formats free-form reports according to the following rules:

1. The Report Extractor finds the last member or members of a single dimension defined in the report specification (before the report output operator !). This dimension becomes the ROW dimension for the report. All remaining selections become PAGE or COLUMN dimensions, as defined by rules 2 and 3.
2. The Report Extractor searches for any single-member selections. If a single member is found that does not satisfy rule 1, that dimension becomes a PAGE dimension.
3. The Report Extractor searches for all remaining dimension members that do not satisfy rules 1 or 2. If any remaining members are found, those dimensions become COLUMN dimensions. COLUMN dimensions are nested in the order of selection in the free-form script.
4. The Report Extractor searches the database outline for any dimensions not specified in the report specification. If any unspecified dimensions are found, they become PAGE dimensions (the default for single-member selections, as defined in rule 2).

5. A subsequent selection of one or more consecutive members from a given dimension overrides any previous selection for that dimension.

For example, the following report recognizes California, Oregon, Washington, Utah, Nevada, and West as members of Market.

```
Sales
Jan Feb Mar
Actual Budget
Apr May Jun
California
Oregon
Washington
Utah
Nevada
{UCHARACTERS}
West
    !
```

The Report Extractor applies free-form formatting rules to this report as follows:

1. Because California, Oregon, Washington, Utah, Nevada, and West are listed last, the Report Extractor treats them as if `ROW(Market)` had been specified (according to rule 1).
2. Sales is a single-member selection from dimension Measures. The Report Extractor treats this member as if `PAGE(Measures)` had been specified (according to rule 2).
3. After searching the remaining members, the Report Extractor finds members of dimensions Year and Scenario, which it treats as COLUMN(Year, Scenario), according to rule 3.
4. The Report Extractor searches the Database Outline and finds that dimension Product is not specified in the report specification. Since Product is a single-member selection, the Report Extractor treats this member as if PAGE (Product) had been specified (according to rule 4).
5. Finally, the Report Extractor finds that Apr May Jun is from the same dimension as Jan Feb Mar and is displayed on a subsequent line of the script. The Report Extractor discards the first specification (Jan Feb Mar) and uses the second (Apr May Jun).

As a result, the report example produces the following report:

```
                               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
```

Note: You cannot use substitution variables in free-form mode.

Chapter 35

Developing Report Scripts

Once you understand the basics of creating report scripts, you can create more complex reports.

You create a report using *extraction commands* which specify member combinations for pages, columns, and rows. You use *formatting commands* to determine the visual design of the report and to control some of the data values' display. Formatted data values are displayed in the report when you run the script, based on the combined extraction and report commands.

Extraction commands perform the following actions:

- Determine the selection, orientation, grouping, and ordering of raw data records extracted from the database. Extraction commands are based on either dimension or member names, or keywords. Their names begin with the greater-than symbol (>).
- Apply to the report from the line on which they occur until the end of the report. If another extraction command occurs on a subsequent line of the report, it overrides the previous command.

Formatting commands perform the following actions:

- Let you customize the format and appearance of a report and create report-time calculations. Formatting commands are generally enclosed inside left and right braces ({ }), although there are several formatting commands that begin with the less-than (<) character.
- Are either applied globally within the report script or are specific to a member.

This chapter provides information about creating complex report scripts, including:

- "Syntax Guidelines" on page 968
- "Designing the Page Layout" on page 970
- "Formatting" on page 976
- "Selecting and Sorting Members" on page 994
- "Restricting and Ordering Data Values" on page 1011
- "Converting Data to a Different Currency" on page 1017
- "Generating Reports Using the C, Visual Basic, and Grid APIs" on page 1018

For fundamental information about reports and report scripts, see Chapter 34, "Quick Start to Report Scripts."

Syntax Guidelines

To build a report, you enter commands that define the layout, member selection, and format you want in the Application Manager's Report Editor. When you write a report script, follow these guidelines:

- Separate commands with at least one space, tab, or new line. Report processing is not affected by extra blank lines, spaces, or tabs.
- Enter commands in either uppercase or lowercase. Commands are not case-sensitive. If the database outline is case-sensitive, the members in the report script must match the outline.
- To start report processing, enter the **!** (bang) report output command or one or more consecutive numeric values. You can place one or more report scripts, each terminated by its own **!** command, in the same report file.
- You can group more than one format command within a single set of braces. For example, these formats are synonymous:

  ```
  {UDATA SKIP}
  {UDATA} {SKIP}
  ```

- Enclose member names in quotation marks in the following cases:
  - Names beginning with an ampersand (for example, “&Product”).
  - Names containing spaces (for example, “Cost of Goods Sold”).
  - Names containing the word Default (for example, “Default”).
  - Names containing one or more numerals at the beginning of the name (for example, “100-Blue”)
  - Names containing any of the following characters:

    ```
    * (asterisks)
    @ (at signs)
    [] (brackets)
    : (colons)
    , (commas)
    {} (braces)
    - (dashes, hyphens, or minus signs)
    < (less than signs)
    () (parentheses)
    + (plus signs)
    / (slashes)
    ; (semicolons)
    ```

- If a formatting command is preceded by three or more underscore, equal sign, or hyphen characters, respectively, the Report Extractor assumes that the characters are extraneous underline characters and ignores them. For example, ==={SKIP 1}.
- Use // (double slash) to indicate a comment. Everything on the line following a comment is ignored by the Report Writer. Each line of a comment must start with a double slash, so you can include multi-line comments.
- Exercise caution if you abbreviate command names. Remember that many names begin with the same letters, and your results may be unexpected unless you use a completely unambiguous abbreviation.

Designing the Page Layout

Reports are two-dimensional views of multidimensional data. You can use page layout commands to incorporate additional dimensions that are defined as nested groups of columns or rows on a page, or additional pages in the report.

The page layout is composed of headings that make up the columns and rows of a page. You define the basic layout of a report using page, row, and column data extraction commands combined with specific member selections.

Each component of page layout has a different formatting command:

<PAGE <COLUMN <ROW

In addition, the <ASYM and <SYM commands override the default method of interpreting the column dimension member lists, and produce either an asymmetric or symmetric report format.

For information about formatting the page, column, or row headings, see "Formatting Page, Column, and Row Headings" on page 977.

Creating Page, Column, and Row Headings

➤ To design the report headings:

1. Type **<PAGE** `(`*`dimensionname`*`, `*`dimensionname`*`)`, where *dimensionname* lists dimensions represented on the current page. All data values on the page have the dimensions in the page heading as a common property. For example,

   ```
   <PAGE (Measures, Market)
   ```

2. Press Enter.
3. Type **<COLUMN** *`(dimensionname)`*, where *dimensionname* equals the name of each dimension to display across the page. For example,

   ```
   <COLUMN (Year)
   ```

   You can add dimension names to create nested column headings.

4. Press Enter.

5. Type **<ROW** `(dimensionname)`, where *dimensionname* equals the name of each dimension to display down the page. For example,

   ```
   <ROW (Market)
   ```

6. Press Enter.

Note: You can select additional members to associate with the heading commands. If you type a member name as a parameter for the PAGE, COLUMN, or ROW commands, Report Extractor automatically associates the member with the appropriate dimension.

The following report script is based on the Sample Basic database:

```
<PAGE (Product, Measures)
<COLUMN (Scenario, Year)
Actual
<ICHILDREN Qtr1
<ROW (Market)
<IDESCENDANTS East
    !
```

This script produces the following report:

```
               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
```

You can create page, column, and row headings with members of attribute dimensions. The following report script is based on the Sample Basic database:

```
<PAGE (Measures,Caffeinated)
Profit
<COLUMN (Year,Ounces)
Apr May
"12"
<ROW (Market,"Pkg Type")
Can
<ICHILDREN East
    !
```

This script produces the following report:

```
                          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
```

Modifying Headings

You can perform the following modifications to headings in the report:

Task	Report Command
Create a custom page heading in place of the default heading, which is displayed at the top of each page in the report or immediately following a HEADING command. Use the ENDHEADING command to specify the end of the custom heading.	STARTHEADING
Display the page heading, either the default heading or the heading as defined with the STARTHEADING and ENDHEADING commands. Use this command to re-enable the page heading display if the SUPHEADING command has been used.	HEADING
Force the immediate display of the heading without waiting for the next non-suppressed data row, when heading suppression commands are in use.	IMMHEADING
Automatically turn on the display of the column header.	COLHEADING
Display the page heading before the next data output row.	PAGEHEADING

For information about suppressing the display of headings in the report, see "Suppressing Page, Column, and Row Formatting" on page 981.

Creating Symmetric and Asymmetric Reports

Essbase reports can contain symmetric or asymmetric column groups. Essbase determines the symmetry of column groups automatically, based on the members you select.

A *symmetric report,* shown below, is characterized by repeating, identical groups of members.

```
           East                           West
   Budget       Actual             Budget      Actual
Q1  Q2  Q3     Q1  Q2  Q3        Q1  Q2  Q3   Q1  Q2  Q3
```

An *asymmetric report,* shown below, is characterized by groups of nested members that differ by at least one member in the nested group. There can be a difference in the number of members or the names of members.

```
             East                           West
   Budget        Actual                    Budget
Q1  Q2  Q3      Q1  Q2  Q3               Q1  Q2  Q3
```

By default, Essbase creates a symmetric report unless you select the same number of members for all column dimensions.

See Chapter 36, “Examples of Report Scripts” for an example of an asymmetric report.

The Essbase evaluation of symmetry versus asymmetry takes place prior to any ordering, restriction on columns, or application of the effects of calculated columns.

Overriding Default Column Groupings

You can override the default column grouping that Essbase selects for reports with the <SYM and <ASYM commands. <SYM and <ASYM affect the member selection commands that follow them in a report.

1. Use the <SYM command when the selection of column members meets the requirements of the rule for asymmetry, but you want to produce a symmetric report. The <SYM command always produces a symmetric report, creating all combinations of each column dimension.
2. Turn off the symmetric format and restore the rules for asymmetric reports with the <ASYM command.

Changing Column Headings

If you only need to change the column headings rather than the symmetry of the report, the <PYRAMIDHEADERS and <BLOCKHEADERS formatting commands are useful.

- Use the <BLOCKHEADERS formatting command to change the pyramid-style headers used in symmetric reports to block-style headers like those used in asymmetric reports. A symmetric report uses the <PYRAMIDHEADERS mode of column layout by default.
- Use the <PYRAMIDHEADERS formatting command to change the block-style headers used in asymmetric reports to pyramid-style headers like those used in symmetric reports. An asymmetric report uses the <BLOCKHEADERS mode of column layout.

Formatting

Formatting commands define the format of data and labels in the final report. These commands are generally enclosed in left and right braces ({ }).

The two types of formatting commands are *global* and *member-specific* commands.

- Global commands are executed when they occur in the report script file, and stay in effect until the end of the report file or until another global command replaces them.

  For example, the {SUPMISSINGROWS} command suppresses all rows in the report script file that contain only missing values.

- Member-specific commands are executed as they are encountered in the report script, usually the next member in the report script, and affect only that member. A format attached to a member is executed before that member is processed.

  For example, the {SKIP} command skips the specified number of rows between row dimensions in a report script. If you want a additional rows to skip lines, you must use the SKIP command again.

Formatting Report Pages

There are a number of formatting commands that you can use to design the look of your final report pages.

See Chapter 36, “Examples of Report Scripts” for report formatting examples.

Setting the Page Length, Width, and Centering

You can set the following page specifications in the report script:

Task	Report Command
Specify the column widths in a report.	WIDTH
Set the left margin of the report.	LMARGIN
Set the center of the page.	SETCENTER

Inserting Page Breaks

You can set the following types of page breaks in the report script:

Task	Report Command
Set the number of lines for each page.	PAGELENGTH
Force a page break regardless of how many lines have been generated for the current page.	NEWPAGE
Insert a page break whenever a member from the same dimension as the member in the command changes from one line in the report to the next. Use the NOPAGEONDIMENSION command to turn off this function.	PAGEONDIMENSION
Enable page breaks in a report when the number of lines on a page is greater than the current PAGELENGTH setting.	FEEDON

Formatting Page, Column, and Row Headings

Column and row formatting commands make up a special type of format setting commands.

Specifying Column Formats

Specifications for column formatting commands can precede or follow the columns to which they apply, depending on the desired format.

For example, in the following script, based on the Sample Basic database, the first {DECIMAL} command is processed after only two columns are set up, Actual and Budget. The {DECIMAL} command, however, refers to a column three, which does not yet exist. Essbase responds to this command by dynamically expanding the report to three columns. When the report specification expands to six columns, the {DECIMAL} formatting applies to columns three *and* six (and all multiples of three).

Essbase performs this pattern extension on the assumption that when another dimension is added, causing repetitions of previous column dimension groups, the formatting should repeat as well. The second {DECIMAL} formatting command is then applied to columns 1 and 4 only, as it occurs after the creation of six columns.

```
<PAGE (Measures, Market)
Texas Sales
     <COLUMN (Scenario, Year)
     Actual Budget
{DECIMAL 2 3 }
     Jan Feb Mar
{DECIMAL 1 1 4 }
<ROW (Product)
<DESCENDANTS "100"
    !
```

This script produces the following report:

```
                            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
```

The following scripts demonstrate two approaches to column formatting that produce identical results. In the first script, the first two {DECIMAL} commands are positioned to format every first and third column by distributing the formats when Jan Feb displays *after* processing the {DECIMAL} command. These examples are based on the Sample Basic database.

```
//Script One: Format Columns by Distributing the Formats

<PAGE (Measures, Market)
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"
    !
```

The two {DECIMAL} commands are positioned to format the individual columns 1, 3, 4, and 6.

```
//  Script Two: Format Columns by Direct Assignment

<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"
    !
```

Both scripts produce the following report:

```
                           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)
```

Accommodating Long Column and Row Names

Member names that are too long to fit into the column are automatically truncated; the tilde character (~) signifies that part of the name is missing. Long member names are common when using aliases in the report.

There are a several ways to modify your columns to display the entire member name.

Task	Report Command
Define the column width for all row members in the column.	NAMEWIDTH
Change where the row member column is displayed, and to shift the remaining columns left or right to make room fro the long member names.	NAMESCOL

Suppressing Page, Column, and Row Formatting

You can suppress the display of page heading, columns, and rows in your report by using various SUPPRESS commands.

Suppress	Report Command
The default column heading in the report.	SUPCOLHEADING
Rows that have only zero or missing values.	SUPEMPTYROWS
All rows that contain missing values. Use INCMISSINGROWS, INCZEROROWS, or INCEMPTYROWS to display rows that are empty, or have missing data or zeros.	SUPMISSINGROWS
The page member heading whenever a heading is generated.	SUPPAGEHEADING
The page and column headings, all member names, page breaks, commas, and brackets in the final report. To turn on the display of columns of row member names, use the NAMESON command. To turn on the use of commas and brackets, use the COMMAS and BRACKETS commands.	SUPALL
The default heading (page header and column headers) or custom header, if defined, at the top of each page.	SUPHEADING
Row member names in the final report. Use the NAMESON command to include row member names in the report.	SUPNAMES
All output while continuing to process all operations, such as calculations, format settings, and so forth. Use the OUTPUT command to reverse the actions of SUPOUTPUT.	SUPOUTPUT
Currency information when you use the CURRENCY command to convert the data values in your report to a specified currency.	SUPCURHEADING

Repeating Row Names

To repeat the row member names on every line of the report, use the <ROWREPEAT command. Use the <NOROWREPEAT command to prevent row member names from being repeated on each line of the report if the row member name does not change on the next line. `NOROWREPEAT` is enabled by default.

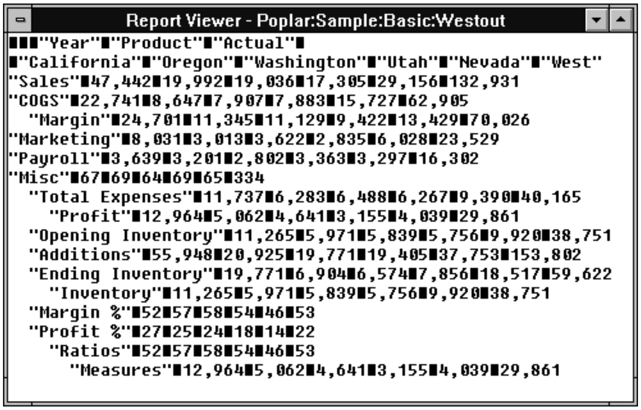
Formatting Reports with Tab Delimiters

You can place tabs between columns rather than spaces in your report scripts. This is useful when you want to export report output into another form.

To replace spaces with tab delimiters, type {TABDELIMIT} anywhere in the report script.

When you save the report script, Essbase automatically replaces the spaces with tabs. When you view the report in the Report Viewer, black squares indicate tab marks.

Figure 490: Tab-Delimited Report



```
■■■"Year"■"Product"■"Actual"■
■"California"■"Oregon"■"Washington"■"Utah"■"Nevada"■"West"
"Sales"■47,442■19,992■19,036■17,305■29,156■132,931
"COGS"■22,741■8,647■7,907■7,883■15,727■62,905
  "Margin"■24,701■11,345■11,129■9,422■13,429■70,026
"Marketing"■8,031■3,013■3,622■2,835■6,028■23,529
"Payroll"■3,639■3,201■2,802■3,363■3,297■16,302
"Misc"■67■69■64■69■65■334
  "Total Expenses"■11,737■6,283■6,488■6,267■9,390■40,165
    "Profit"■12,964■5,062■4,641■3,155■4,039■29,861
  "Opening Inventory"■11,265■5,971■5,839■5,756■9,920■38,751
  "Additions"■55,948■20,925■19,771■19,405■37,753■153,802
  "Ending Inventory"■19,771■6,904■6,574■7,856■18,517■59,622
    "Inventory"■11,265■5,971■5,839■5,756■9,920■38,751
  "Margin %"■52■57■58■54■46■53
  "Profit %"■27■25■24■18■14■22
    "Ratios"■52■57■58■54■46■53
      "Measures"■12,964■5,062■4,641■3,155■4,039■29,861
```

Adding Totals and Subtotals

Column and row calculations let you create additional calculations that are not defined as part of the database outline. For example, you can use column and row calculations to create extra columns or rows in a report, based upon selected data members, and perform calculations on these or existing columns and rows.

For examples of report scripts that contain column and row calculations, see Chapter 36, “Examples of Report Scripts.”

Totaling Columns

The CALCULATE COLUMN command lets you create a new report column, perform on-the-fly calculations, and display the calculation results in the newly created column.

The following table summarizes column calculation commands:

Task	Report Command
Create a new report column, perform Dynamic Calculations, and display the calculation results in the newly created column. Use the OFFCOLCALCS command to temporarily disable column calculations in the report, and ONCOLCALCS to re-enable calculations.	CALCULATE COLUMN
Remove all column calculation definitions from the report.	REMOVECOLCALCS

CALCULATE COLUMN adds up to 499 ad hoc column calculations to a report. Each new calculated column is appended to the right of the existing columns in the order in which it is created, and given the next available column number. These columns calculate the sum of data across a range of columns or an arithmetic expression composed of simple mathematical operators.

The CALCULATE COLUMN command supports the standard mathematical operations. For syntax and parameter descriptions, see the *Technical Reference* in the `docs` directory.

If you use the same name for more than one column, Essbase creates only the last column specified in the CALCULATE COLUMN command. Use a leading space with the second name (and two leading spaces with the third name, and so on) to create a unique column name.

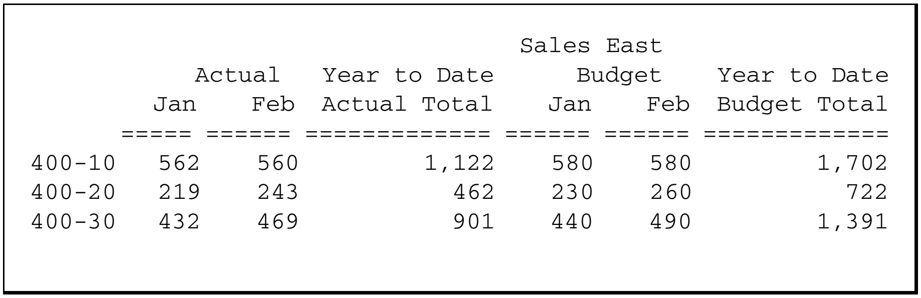
Alternately, you can add descriptive text far enough to the right that it is truncated to the column width. You could, for instance, use the names Q1 Actual and Q1 Budget to distinguish similar column names without affecting the appearance of the report. Column names are printed with right justification until the column header space is filled. Excess characters are then truncated to the right.

Divide lengthy column name labels into two or more lines. The maximum number of lines across which you can divide a label is equal to the number of column dimensions designated in the report specification. To do this, insert the tilde character (~) in the name at the point where you want the break. You must also specify at least two members for each column dimension to use the maximum number of lines.

This example is based on the Sample Basic database.

```
{CALCULATE COLUMN "Year to Date~Actual Total" = 1 : 2}
{CALCULATE COLUMN "Year to Date~Budget Total" = 3 : 4}
```

The example produces the following report:



```
                                   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
```

As a rule, in symmetric reports, if a calculated column name has fewer levels than the number of column dimensions, the previous member (to the left) of each of the column dimensions, above the top level supplied in the calculated column name, is attributed to the calculated column. If normal PYRAMIDHEADERS mode is in use, the centering of those higher-level column members shifts to the right to include the calculated column or columns. Column header members on the same level as calculated column names are not applied to the new calculated column or columns, and their centering does not shift.

If BLOCKHEADERS mode is in use, that is, if every member applying to a column is repeated above that column, the same rules apply, except that instead of shifting column header member centering, they are repeated in the appropriate higher levels of the calculated column name.

Asymmetric reports do not have groups of columns that share a member property. These reports still allow multiple-level column names up to the number of column levels defined, but member properties from preceding columns are not automatically shared and used for those levels that are not defined.

In cases where there are fewer column header dimensions than the number of levels that you want, you can create multi-line column labels. In this case, use TEXT, STARTHEADING, ENDHEADING, and other formatting commands to create a custom heading.

For the syntax and definitions of column calculation commands, see the *Technical Reference* in the `docs` directory.

Numbering Columns

If the number of regular (noncalculated) columns varies in the report because multiple sections in the report have different numbers of columns, the column numbers used to identify the calculated columns shift accordingly. For example:

- If the first section of a report has 12 columns (including row name columns), and 3 calculated columns are declared, column numbers 0–11 are the regular columns, and columns 12–14 are the calculated columns.
- If a second section of the report reduces the number of regular columns to 6, then the regular columns are columns 0–5, and the same calculated columns are columns 6–8.
- Likewise, if the number of regular columns is increased, the numbering of the calculated columns starts at a higher number.

In the example, CC1, CC2, and CC3 represent the names of three calculated column names. The column numbering for a report with two different sections with varying numbers of regular columns would look like this:

```
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
```

If you do not want the calculated columns in the second section, or if you need a different set of column calculation, use the command REMOVECOLCALCS to clear the old ones out. You can then define new column calculations.

This example assumes that all three column calculations had no references to regular columns other than columns 1 and 2. If CC3's calculation were = 1 + 3 + 6, when the second section of the report starts, an error would occur stating that the column calculation referred to a nonexistent column (6).

Totaling Rows

Row calculations create summary rows in a report. You can use *summary rows* to calculate the sum of data across a range of rows or to calculate an arithmetic expression composed of simple mathematical operators.

The following table summarizes row calculation commands:

Task	Report Command
Create a new row and associate it with a row name or label. This is similar to declaring a variable. You can also perform simple calculations with CALCULATE ROW. For more complex calculations, use SETROWOP. See also OFFROWCALCS and ONROWCALCS.	CALCULATE ROW
Temporarily disable row calculations in the report. See also CALCULATE ROW and ONROWCALCS.	OFFROWCALCS
Re-enable calculations after using OFFROWCALCS. See also CALCULATE ROW and OFFROWCALCS.	ONROWCALCS
Define complex calculations for the row specified in CALCULATE ROW. SETROWOP defines a calculation operator to be applied to all subsequent output data rows. You can display the calculation results in the newly created row with the PRINTROW command.	SETROWOP
Immediately display the row specified in CALCULATE ROW to the report.	PRINTROW
Reset the value of the calculated row to #Missing. See also CLEARALLROWCALC.	CLEARROWCALC
Reset the value of all calculated rows after using the CLEARROWCALC command.	CLEARALLROWCALC
Create a new calculated row with captured data. See also SAVEANDOUTPUT.	SAVEROW
Capture data and output the result after using the SAVEROW command.	SAVEANDOUTPUT

For the syntax and definitions of row calculation commands, see the *Technical Reference* in the `docs` directory.

Commands that designate columns must use valid data column numbers, as determined by the *original* order of the columns.

- Precede and follow all operators in an expression with a single space.
- Essbase does not support nested (parenthetical) expressions.
- Essbase supports integer and floating-point constants in expressions as single entries or members of an array.

The CALCULATE ROW command can specify an operation (+, -, *, /, or OFF) as an equation consisting of constants, other calculated rows, and operators. Equations are evaluated at the time of declaration.

If you specify an operator, it applies to subsequent output rows and stores the result in the calculated row. This is useful for aggregating a series of rows to obtain a subtotal or total. To reset the operator, use SETROWOP. If the CALCULATE ROW command does not specify either an equation or an operator, the + operator is assumed.

The CALCULATE ROW command supports the standard mathematical operations. For syntax and parameter descriptions, see the *Technical Reference* in the `docs` directory.

This example is based on the Sample Basic database.

```
{ CALC ROW "Total Sales" = "Sales..Group1"
     + "Sales..Group2" }
```

The example creates “Total Sales” based on two other calculated rows.

Changing How Data is Displayed

You can use a variety of formatting commands to customize how data displays in your final report.

Underlining

Use underlining as a visual aid to break up blocks of information in a report.

Task	Report Command
Set the default underline character that will display in the report.	UNDERLINECHAR
Underline all nonblank characters in the preceding row.	UCHARACTERS
Underline all the columns in the preceding row.	UCOLUMNS
Underline all the data columns for a row, while not underlining the row name columns.	UDATA
Underline all the row name columns in the preceding row while not underlining the data columns.	UNAME
Underline the row member names in a row whenever a member from the same dimension as the member in the command changes. Use the NOUNAMEONDIM command to turn off underlining for new rows.	UNAMEONDIMENSION

Suppressing Data Formatting

You can suppress data that you do not want to be displayed in your final report by using various SUPPRESS commands.

Suppress	Report Command
Brackets around negative numbers. Use the BRACKETS command to re-enable brackets.	SUPBRACKETS
Commas in numbers greater than 999. Use the COMMAS command to re-enable commas.	SUPCOMMAS
Rows that have only zero data values. Use INCZEROROWS or INCEMPTYROWS to re-enable the display.	SUPZEROROWS
The European method for displaying numbers (2.000,01 whereas the non-European equivalent is 2,000.01). Use the EUROPEAN command to re-enable European number display.	SUPEUROPEAN
The automatic insertion of a page break whenever the number of lines on a page exceeds the current PAGELENGTH setting.	SUPFEED
Formats that produce output such as underlines and skips. Use INCFORMATS to re-enable the display.	SUPFORMATS
Text masks that were defined in the report using the MASK command. Use INCMASK to re-enable the display.	SUPMASK

See "Suppressing Page, Column, and Row Formatting" on page 981 for information about suppressing the display of page, column, and row formats.

Indenting

Use indenting to provide visual clues to row levels of the script.

Task	Report Command
Shift the first row names column in column output order by a specified number of characters.	INDENT
Indent subsequent row members in the row names column based on the generation in the database outline. Use the NOINDENTGEN command to left-justify row member names based on generation name.	INDENTGEN

Inserting Custom Titles

Titles are user-generated and optional, in contrast to the automatically generated page and column headings and row names, which describe the data on the report page.

Titles repeat at the top of each report page, and provide such valuable information about a report as:

- A date and time stamp
- The user ID of the person running the report
- Page numbers
- The name of the source database
- Any other descriptive information

To add a title to the report, use the TEXT command, combined with:

- Any of the predefined keywords that automatically display information in the report
- A text string that you define

Note: You can also use the TEXT command at the bottom of the report to provide summary information.

See the *Technical Reference* in the `docs` directory for the syntax and definitions of Report Writer commands.

Replacing Missing Text or Zeros with Labels

When you run a report, there are often many empty data cells where no data was applicable to the retrieval, or cells where the value is zero.

The report displays the default `#MISSING` label in the data cell when no data values are found.

To replace the #MISSING label with a text label:

At the point in the script where you want to replace the `#MISSING` label with a text label, type

{MISSINGTEXT ["*text*"]}

where *text* is any text string that you want to display in the data cells.

You can place the MISSINGTEXT command at any point in the report script; the command applies throughout the script.

Note: You can also suppress `#MISSING` labels from appearing in the report. See "Suppressing Data Formatting" on page 990 for information about suppressing labels, or see the *Technical Reference* in the `docs` directory for the syntax and definitions of Report Writer commands.

To replace zeros with a text label:

At the point in the script where you want to replace zeros with a text label, type

{ZEROTEXT ["*text*"]}

where *text* is any text string that you want to display in the data cells.

Note: If a value is equal to #MISSING the string being inserted after that value will not print. This is also true if you replace #MISSING with some other value (such as 0).

Adding Blank Spaces

Adding blank spaces in a report draws the reader to key information, such as totals.

Task	Report Command
Add one or more blank lines in the final report.	SKIP
Add a blank line when a member from the same dimension as the specified member in the command changes on the next line in the report. Use the NOSKIPONDIMENSION command to turn off insertion of a new line.	SKIPONDIMENSION

Changing How Data Values Display

You can use the following commands to change how data values display in your final report.

Task	Report Command
Turn on the display of commas for numbers greater than 999 after commas have been suppressed with either a SUPCOMMAS or SUPALL command.	COMMAS
Turn on the display of brackets around negative numbers instead of negative signs, after using the SUPBRACKETS command earlier in the script.	BRACKETS
Include a percentage sign or other character after the data values.	AFTER
Include a dollar sign or other character before the data values.	BEFORE
Use the European method for displaying numbers where decimal points are used as the thousands separator character while commas separate the decimal portion of the number from the integer portion (2.000,01, whereas the non-European equivalent is 2,000.01).	EUROPEAN
Overwrite text in each output row with a specified characters and position.	MASK

Selecting and Sorting Members

The data that is displayed in the final report is based upon the members that you select and the order in which you display them. In addition, you can use conditional retrievals to further refine the selection and sorting of members.

Selecting Members

Member selection commands are extraction commands that select ranges of members based on database outline relationships, such as sibling, generation, and level. Using member selection commands ensures that any changes to the outline are automatically reflected in your report, unless you change the member name on which the member selection command is based. Attribute dimensions can be included in member selection commands.

Task	Report Command
Select all the members from the same dimension as the dimension member.	ALLINSAMEDIM
Include all the siblings of the specified member.	ALLSIBLINGS
Include all the ancestors of the specified member.	ANCESTORS
Select a base dimension member based on its attributes.	ATTRIBUTE
Select all members in the level immediately below the specified member.	CHILDREN
Include the descendants of the specified member to the report, excluding the dimension top.	DESCENDANTS
Select the Level 0 members at the bottom of the dimension.	DIMBOTTOM
Include the top member of the dimension.	DIMTOP
Include a member and its ancestors.	IANCESTORS
Select the specified member and all members in the level immediately below it.	ICHILDREN
Include the specified member and its descendants.	IDESCENDANTS
Include the specified member and its parent.	IPARENT

Task	Report Command
Include all members from the same dimension and generation as the specified member.	OFSAMEGEN
Include all members from the same dimension and on the same level as the specified member.	ONSAMELEVELAS
Include the parent of the specified member to the report.	PARENT
Extract data for a specified date or for a time period before or after a specific date.	TODATE
Include base dimension members associated with the specified attribute dimension.	WITHATTR

Selecting Members Using Generation and Level Names

Generation and level name selection commands identify a specific level or generation of members based on either:

- The default generation or level name in the outline
- The user-defined generation or level name in the outline

When you use generation and level names, changes to the outline are automatically reflected in your report. You can either define your own generation and level names, or you can use the default names provided by Essbase. For information on generations and levels, see Chapter 18, “Introducing Dynamic Dimension Building.”

Using generation or level names whenever possible makes your report easier to maintain. Because you do not have to specify a member name in your report, you do not need to change the report if the member name is changed or deleted from the database outline.

Note: Generation and level names are stand-alone commands. You cannot use them in place of member names in report extraction or formatting commands. For example, you *cannot* use them with the <DESCENDANTS or <CHILDREN commands.

To use default level names:

At the point in the script where you want to select a member by the default level name, use the following format:

```
Levn,DimName
```

where *n* is the level number.

DimName is the name of the dimension from which you want to select the members.

Note: Do *not* put a space after the comma.

For example, Lev1,Year selects all the level 1 members of the Year dimension.

To use default generation names:

At the point in the script where you want to select a member by the default generation name, use the following format:

```
Genn,DimName
```

where *n* is the generation number.

DimName is the name of the dimension from which you want to select the members.

Note: Do *not* put a space after the comma.

For example, Gen2,Year selects all the generation 2 members of the Year dimension.

Note: These default generation and level names are not displayed in the Outline Editor.

The following example is based on the Sample Basic database. It uses the default generation name Gen2,Year to generate a report that includes the members Qtr1, Qtr2, Qtr3, and Qtr4 from the Year dimension.

```
<PAGE(Product)
<COLUMN(Year)
<ROW (Measures)
{OUTALTNAMES}
Cola
Gen2,Year
Sales Profit
    !
```

The report script produces the following report:

```
                             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
```

Selecting Dynamic Time Series Members

You create and identify dynamic members in the database outline; they are members that are calculated only during user retrieval requests, such as generating a report script. The Time dimension contains a special Dynamic Time Series tag which has reserved generation names that you can define in the outline alias table.

Generation Name	Reserved Names	Explanation
History	H-T-D	History-to-Date
Year	Y-T-D	Year-to-Date
Season	S-T-D	Season-to-Date
Period	P-T-D	Period-to-Date
Quarter	Q-T-D	Quarter-to-Date
Month	M-T-D	Month-to-Date
Week	W-T-D	Week-to-Date
Day	D-T-D	Day-to-Date

See Chapter 6, “Creating Applications and Databases” for information about creating and maintaining Dynamic Time Series generation names in the database outline.

Note: The outline’s database header message identifies the number of dynamic members that are enabled in the current outline.

To select a Dynamic Time Series member:

At the point in the script where you want to select a Dynamic Time Series member, use either of the following formats:

```
<LATEST memberName
```

where *memberName* is the name of the member in the Time dimension.

The <LATEST command is a global command that is applied to the entire report script, and is an aggregation based on the lowest level member within the dimension.

```
reservedName(memberName)
```

where *reservedName* is the reserved Dynamic Time Series generation name, and the *memberName* is the name of the member in the Time dimension.

If you use this syntax to specify a Dynamic Time Series, the time series name is associated only to the member listed in the argument.

When you run the report script, the members are dynamically updated, and the information is incorporated into the final report.

Note: You must type the Dynamic Time Series string exactly as it is displayed in the database outline; you cannot create your own string and incorporate it into the final report.

You can create an alias table for the Dynamic Time Series members in the database outline, and use the aliases instead of the predefined generation names.

Selecting Members Using Boolean Operators

Boolean operators let you specify precise member combinations within your report, which is particularly useful when dealing with large outlines. Use the AND, OR, and NOT Boolean operators, combined with extraction commands, to refine your member selections within the report script.

- Use the AND operator when all conditions must be met.
- Use the OR operator when one condition of several must be met.
- Use the NOT operator to choose the inverse of the selected condition.

To create a Boolean expression using operators:

At the point in the script where you want to use linking, enter the following format:

```
<LINK (extractionCommand [operator extractionCommand])
```

where *extractionCommand* is the member selection command to retrieve data from, and *operator* is either the AND or OR operator.

Note: You must select members from the same dimension, and all extraction command arguments must be enclosed in parentheses, as in the example above. NOT can only be associated with an extraction command, and does not apply to the entire expression.

You can use Boolean operators with member selection commands, such as UDA and wildcards. See the *Technical Reference* in the `docs` directory for a list of all valid extraction commands that can be used in conjunction with the LINK command.

Examples:

```
<LINK ((<IDESCENDANTS("100") AND <UDA(Product,Sweet)) OR
ONSAMELEVELAS "100"-10")
```

selects sweet products from the “100” subtree, plus all products on the same level as “100-10.”

```
<LINK ((<IDESCENDANTS("100") AND NOT <UDA (Product,Sweet)) OR
ONSAMELEVELAS "100"-10")
```

selects non-sweet products from the “100” subtree, plus all products on the same level as “100-10.

See Chapter 36, “Examples of Report Scripts” for other examples of narrowing member selection criteria.

Selecting Members Using Substitution Variables

Substitution variables act as global placeholders for information that changes regularly; you set the substitution variables on the server through Application Manager, MaxL, or ESSCMD, and assign a value to each variable. You can then change the value at any time, reducing manual changes to a report script. You must have the role of at least Database Designer to set substitution variables.

For example, many reports are dependent on reporting periods; if you generate a report based on the current month, you would have to manually update the report script every month. With a substitution variable set on the server, such as CurMnth, you can change the assigned value each month to the appropriate time period. Essbase dynamically updates the information when you run the final report.

See Chapter 6, “Creating Applications and Databases” for information about creating and changing substitution variables in the database outline. See the *Technical Reference* in the `docs` directory, for information about the leading & character.

You can set substitution variables at the following levels:

- Server, providing access to the variable from all applications and databases on the server
- Application, providing access to the variable from all databases within the application
- Database, providing access to the specified database

To use a substitution variable:

The substitution variable must be accessible from the application and database against which you are running the report.

At the point in the script where you want to use the variable, use the following format:

```
&variablename
```

where *variablename* is the same as the substitution variable set on the server.

For example,

```
<ICHILDREN &CurQtr
```

becomes

```
<ICHILDREN Qtr1
```

Note: The variable name can be an alphanumeric combination whose maximum size is specified in Appendix A, “Limits.” You cannot use spaces or punctuation in the variable name.

When you run the report script, Essbase replaces the variable name with the substitution value and that information is incorporated into the final report.

Selecting Members Using Attributes

Using attributes, you can select and report on data based on one or more characteristics of base members. You can group and analyze members of base dimensions according to their attributes. You can also perform crosstab reporting based on two or more attributes. Using the <ATTRIBUTE command, you can select all the base dimension members associated with an attribute. For example, you can query the Sample Basic database on how many 12-ounce units of grape flavored juice and orange flavored juice were sold in New York during the first quarter.

To select a member based on a specific attribute:

At the point in the script where you want to select members based on a specific attribute, use the following format:

```
<ATTRIBUTE mbrName
```

where *mbrName* is the name of an attribute-dimension member; for example:

```
<ATTRIBUTE Bottle
```

returns all products packaged in bottles.

There can be cases where attribute dimensions have members with the same name. For instance, the attribute dimension Ounces and the attribute dimension Age can each have a member named 24. To ensure that a query returns correct results, you must specify the full attribute-dimension member name, as set in the Attribute Member Name dialog box in the Essbase Application Manager. The following format:

```
<ATTRIBUTE Ounces_24
```

returns all products that are packaged in 24 oz. units.

Attribute types can be text, numeric, date, and Boolean. See Chapter 9, “Working with Attributes” for more information.

Selecting Members Associated with a Specific Attribute

You can select all the base dimension members associated with one or more attributes using the <WITHATTR command. For example, you can display all products associated with a member of the Pkg Type attribute dimension. At the point in the script where you want to select the members, enter the following syntax:

```
<WITHATTR (AttrDimName, "Operator", Value)
```

For example:

```
<WITHATTR (Population, "IN", Small)
```

returns all base dimension members that are associated with the attribute Small from the Population attribute dimension.

```
<WITHATTR (Ounces, "<", 32)
```

returns all base dimension members that are associated with the attribute 32 from the Ounces attribute dimension.

Note: The <WITHATTR command can be used within the LINK command to refine member selections. For example:

```
<LINK ((<WITHATTR (Ounces, "<", 32) AND <WITHATTR ("Pkg Type",
"=", Can))
```

For more information on the syntax for the <ATTRIBUTE and <WITHATTR commands, see the *Technical Reference* in the `docs` directory.

Selecting Members by a Specific Date

You can extract attributes data for a specific date, for a period before a specific date, or for a period after a specific date using the <TODATE command. For example, you can extract information on all products that were introduced on December 10, 1996, before December 10, 1996, or after December 10, 1996. The <TODATE command must be used within the <WITHATTR command. For example, the following format:

```
<WITHATTR ("Intro Date", "=", <TODATE ("mm-dd-yyyy",
"12-10-1996")
```

returns data on all products that were introduced *on* December 10, 1996.

The following format:

```
<WITHATTR ("Intro Date", "<", <TODATE ("mm-dd-yyyy",
"12-10-1996")
```

returns data on all products that were introduced *before* December 10, 1996.

The following format:

```
<WITHATTR ("Intro Date", ">", <TODATE ("mm-dd-yyyy",
"12-10-1996")
```

returns data on all products that were introduced *after* December 10, 1996.

Note: The types of date format supported are mm-dd-yyyy or dd-mm-yyyy. The date must be between January 1, 1970 and January 1, 2038 (inclusive).

For more information on the syntax for the <TODATE command, refer to the Report Writer commands in the *Technical Reference* in the `docs` directory.

Selecting Members Using UDAs

A user-defined attribute (UDA) enables you to select and report on data based on a common characteristic. These attributes are particularly useful when performing member selections from an outline with an unbalanced hierarchy (a hierarchy where the members of a dimension do not have identical member levels). You can set UDAs on the server for characteristics such as color, size, gender, flavor, or any other common member characteristics. You must have Database Designer privileges to set UDAs on the server.

UDAs are different from attributes. UDAs are member labels that you create to extract data based on a particular characteristic, but you cannot use UDAs to group data, to perform crosstab reporting, or to retrieve data selectively. Hence, for data analysis, UDAs are not as powerful as attributes.

You can use the UDA command in conjunction with Boolean operators to refine report queries further. See "Selecting Members Using Boolean Operators" on page 998 for more information.

See Chapter 6, "Creating Applications and Databases" for information about creating and maintaining UDAs.

To select members based on a UDA:

At the point in the script where you want to select members based on the UDA, use the following format:

```
<UDA (dimName,"UDAstring")
```

where *dimName* is the dimension of the member that you select, and *UDAstring* is the UDA that is set on the server. For example:

```
<UDA (product,"Sweet")
```

When you run the report script, Essbase incorporates the UDA members into the final report.

Note: You must type the UDA string exactly as it is displayed in the database outline; you cannot create your own UDA string and incorporate it into the report script.

Selecting Members Using Wildcards

You can use wildcards to select either members, generation, or level names in a report script. If you use member names, Essbase searches the member and all descendants of that member. If you specify a generation or level name, Essbase searches only members of that generation or level.

Using wildcards reduces the amount of member information needed for a script and simplifies script maintenance.

The following two types of wildcards are supported in Report Writer:

- Trailing asterisk (*) characters at the end of the string to search for common member properties
- Pattern-matching question mark (?) characters anywhere in the string to represent any single-character member property

To select members using a trailing wildcard:

At the point in the script where you want to select members using a trailing wildcard, use the following format:

```
<MATCH (mbrName,"charName*")
```

where *mbrName* is the name of the member that you select, and *charName* is the beginning character in the following member. Using the Sample Basic database,

```
<MATCH (Year,"J*")
```

returns Jan, Jun, and Jul.

To select members using a pattern-matching wildcard:

At the point in the script where you want to select members using a pattern-matching wildcard, use the following format:

```
<MATCH (mbrName,"???charNames")
```

where *mbrName* is the name of the member to select, and *charNames* are the characters in the following member. Using the Sample Basic database,

```
<MATCH (Product,"???-10")
```

returns 100-10, 200-10, 300-10, and 400-10.

Note: In the Sample Basic database example, three question marks are used to represent the variable three characters in the string. If only two question marks were used in the example, no matches would be found. You can place question mark wildcards anywhere in the match string.

Selecting Members Using Static Member Names

Static member names are nonchanging member names, such as Sales and COGS, that you enter directly into the report script. Although they are easy to establish, static member name definitions can be difficult to maintain if your database outline changes. You must change the member name in every script that is used with that database.

In general, report scripts are easier to maintain if you use generation and level names, or member selection commands, rather than static member names whenever possible. For example, when you remove a product, such as Widgets, from the dimension, the member selection command <children Product works, but a specific list referencing Widgets as a static member name generates the following error message:

```
"Unknown Member [Widgets]."
```

Static members tend to be particularly difficult to maintain when dealing with distributed OLAP databases, where both source and target databases must be consistent.

Note: These user-defined, static member generation and level names are not displayed in the Outline Editor.

The following report script uses a static member called ProductGroups to select all the Level 1, Product member names from the Sample Basic database.

```
<PAGE(Year)
<COLUMN(Product)
<ROW (Measures)
Qtr1
ProductGroups
Sales Profit
    !
```

The report script produces the following report:

```
                        Qtr1 Market Scenario
                  100      200      300      400     Diet
             ======== ======== ======== ======== ========
Sales          25,048   26,627   23,997   20,148   25,731
    Profit      7,048    6,721    5,929    5,005    7,017
```

When you run a report that includes static member definitions, the report displays members in order of their definition in the report script by member name. Sort commands have no effect on static member definitions. See “Sorting Members” on page 1010 for more information about the effects of sorting members.

Suppressing Shared Members

You can suppress the display of duplicate shared members when you extract data for your report. You can only suppress shared member display in conjunction with the following:

- Generation names
- Level names
- DIMBOTTOM command
- OFSAMEGEN command
- ONSAMELEVELAS command

Suppressing shared members is useful when you want to eliminate unnecessary duplication of data within the report.

To suppress shared members:

At the point in the script from which you want to suppress a shared member, use

```
<SUPSHARE
```

This suppresses the display of shared members for the duration of the report script. Use the <SUPSHAREOFF command to reset the display of shared members in the script.

See “Suppressing Page, Column, and Row Formatting” on page 981 for more information about suppressing data from your final report.

Selecting Alias Names for Members

Aliases make reports easier to read and help your reader focus on the data values rather than the meanings of member names. You can display members in a report by their aliases. For example, you can display page, column, and row names, such as Diet Cola or Caffeine Free Cola, rather than the corresponding member names 100-20 and 100-30.

Task	**Report Command**
Display the alias set in the database outline's current alias table, without the member name.	OUTALT
Display the alias set in the database outline's current alias table, followed by the member name.	OUTALTMBR
Display the member name, followed by the alias set in the database outline's current alias table.	OUTMBRALT
Display the alias set in the database outline's current alias table, without the member name, for all members in the report.	OUTALTNAMES
Reset the default display of member names after using the OUTALTNAMES command.	OUTMBRNAMES
Include several alias tables within one report script.	OUTALTSELECT

For a complete listing of alias commands, see the *Technical Reference* in the `docs` directory.

To Display a Member Name and Alias

You can display members in a report as a combination of the member name and its alias. This lets you display more descriptive page, column, and row names, such as Diet Cola 100-20 or 100-30 Caffeine Free Cola.

To display a member name and alias, use the <OUTALTNAMES and <OUTALTMBR commands together in a report script, as shown in the following example:

```
<PAGE (Product, Measures)
<COLUMN (Scenario, Year)
{OUTALTNAMES}
<OUTALTMBR
Actual
<ICHILDREN Qtr1
<ROW (Market)
<IDESCENDANTS "300"
    !
```

The report script produces the following report:

```
                 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
```

Sorting Members

When you sort the members you include in your report, be aware that sorting commands affect members differently, depending on whether they are referenced by member selection commands or by static member definitions. Report Writer commands sort members either by member name or data values.

Member selection commands such as <CHILDREN and <DESCENDANTS, select members in the order specified by the database outline. By default, a report that includes member selection commands displays members in their hierarchical database outline order. You can override this default by specifying a sort order with a sort command.

Because sort commands affect the order of the members selected by the member selection commands, they must precede any member selection commands to which they apply. If you specify a sort command, the sort order is preserved until another sort command overrides it.

Sort commands modify member selection commands, such as <CHILDREN and <DESCENDANTS. Sort commands do not perform any final sorting of rows during formatting. Be careful when you place a sort command in your report script that you do not start the sort too soon, and that you override it to turn it off, if necessary, before the next selection command.

Sort commands have no effect on static member definitions.

Task	**Report Command**
Sort all members alphabetically by the alias name of the member, if aliases are used in the report script.	SORTALTNAMES
Sort all following members in ascending order starting with the lowest generation and moving toward the highest generation.	SORTASC
Sort all following members in descending order starting with the highest generation and moving toward the lowest generation.	SORTDESC
Sort all following members according to the generation of the member in the database outline.	SORTGEN
Sort all following members according to the level of the member in the database outline.	SORTLEVEL

Task	Report Command
Sort all members alphabetically by member name.	SORTMBRNAMES
Disable all previous sorting commands so that members added to the report follow the normal hierarchical order based on the database outline.	SORTNONE

For a list of sorting commands syntax and descriptions, see the *Technical Reference* in the `docs` directory.

Restricting and Ordering Data Values

Several Report Writer commands let you perform conditional retrieval and data sorting in your reports.

Task	Report Command
Specify the conditions the columns of a data row must satisfy before the row is returned.	RESTRICT
Specify the ordering of the rows of a report, based on the data values of data columns.	ORDERBY
Specify the number of rows to return. These rows must contain the top values of a specific data column.	TOP
Specify the number of rows to return. These rows must contain the lowest values of a specific data column.	BOTTOM

For the syntax and definitions of these commands, see the *Technical Reference* in the `docs` directory. For detailed examples using these commands, see Chapter 36, “Examples of Report Scripts.”

Configurable variables are used during conditional retrievals. For information about setting the Report Writer configurable variables, see Chapter 53, “Optimizing Reports and Other Types of Retrieval.”

Defining the Order of Operation

<RESTRICT, <ORDERBY, <TOP, and <BOTTOM can be displayed anywhere in the report script and in any order. When using these commands, place all global script formatting commands before a Page member or a Column member, or before a <PAGE command or <COLUMN command that expands into Page or Column members (for example, IDESCENDANTS, or ICHILDREN).

Essbase extracts data and applies restrictions and ordering in the following order:

1. Applies RESTRICT and any existing restrictive option such as SUPPMISSING, SUPZEROS, and SUPEMPTYROWS.
2. Applies TOP or BOTTOM, or both.
3. Applies ORDERBY.

Essbase then returns rows and displays output.

See Chapter 34, “Quick Start to Report Scripts” for more information.

Using TOP, BOTTOM, and ORDERBY with Sorting Commands

<TOP, <BOTTOM, and <ORDERBY commands sort a report’s output by its data values. Essbase applies <TOP and <BOTTOM first, followed by <ORDERBY. If the report contains a sort command, such as <SORTMBRNAMES, which sorts members and not data, Essbase applies the sort command first, followed by <TOP and <BOTTOM, and then <ORDERBY. <ORDERBY is the final sort that takes place.

Restricting Data Ranges

The arguments of the <RESTRICT command let you specify qualifications for selecting rows. Essbase includes only qualified rows in the resulting report output.

<RESTRICT works only on the range of rows that you specify in a row member selection.

Essbase processes the restrictions from left to right, and does not allow grouping with parentheses in the list of arguments.

For example, the following example is *not* a valid syntax:

```
RESTRICT (... (@DATACOL(1) > 300 AND @DATACOL(2) < 600)...)
```

Only one <RESTRICT is allowed per report, as terminated by the **!** command. If a report script contains more than one report, each <RESTRICT overwrites the one in the previous report. For example:

```
RESTRICT (@DATACOL(1) > @DATACOL(2) AND 800 < @DATACOL(3)
OR @DATACOL(4) <> #MISSING)
```

This <RESTRICT command is equivalent in operation to the following syntax:

```
RESTRICT (((@DATACOL(1) > @DATACOL(2)) AND (800<@DATACOL(3))) OR
(@DATACOL(4) <> #MISSING))
```

Ordering Data

The <ORDERBY command orders the output rows according to the data values in the specified columns. You can specify either ascending <ASC (the default) or descending <DESC.You can specify different sorting directions in different columns of the same report.

To determine the set of rows to be ordered, specify the row grouping dimension in the command. The default row grouping is the innermost row dimension.

Only one <ORDERBY is allowed per report, as terminated by the **!** command. If a report script contains more than one report, each <ORDERBY overwrites the one in the previous report.

Using ORDERBY with Formatting Commands

Follow these guidelines when using the <ORDERBY command in a report script:

- Avoid using row formatting commands when you are using <ORDERBY in a report. Formatting commands scheduled for a given point in the report may show up unexpectedly because <ORDERBY shifted the row that contained the member with formatting.
- In general, avoid using row formatting commands, and place overall script formatting before column members or commands that expand the column members (such as “ICHILDREN column dimension, <column ..., column member”).

Specifying Rows to Return

The <TOP and <BOTTOM commands specify the qualified number of rows with the highest or lowest column values, respectively, within a row group to be returned in a report. If the row group member is not specified, the innermost row group dimension is the default row group.

You can use <TOP and <BOTTOM together in the same report, but only one <TOP and one <BOTTOM is allowed per report. In this case, the two commands should have the same data column as their argument in order to prevent confusion. The result of the <TOP and <BOTTOM command is sorted by the value of the data column specified in the command in descending order.

<TOP and <BOTTOM work only on the range of rows specified in row member selection.

Note: If <TOP or <BOTTOM occurs with <ORDERBY, the ordering column of the <ORDERBY does not have to be the same as the data column of the <TOP or the <BOTTOM.

If any combination of the <ORDERBY, <TOP, or <BOTTOM commands exist together in a report script, the row group member (*<rowgroupmember>*) should be the same. This restriction removes any confusion about the sorting and ordering of rows within a row group.

CAUTION: Essbase discards rows that contain `#MISSING` values in their sorting column from the set of extracted data rows before the applying the TOP or BOTTOM sort.

For example, this command returns two rows with the highest data values in col2 (Actual, Qtr2) per row group:

```
1- TOP (2, @DATACOL(2))
```

When you run this command against the Sample Basic database, the row grouping is Product, which implies that for FLORIDA, the report returns 100-10 and 100-30 product rows, and for MAINE, the report returns 100-10, 100-40 product rows, and so on.

```
                            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
```

This example returns rows with the highest data values in col2 (Actual, Qtr2) per report, because the row grouping is the "`market`."

```
2- TOP("market", 3, @DATACOL(2))
```

This command returns the following rows:

```
New York    100-10     1010     1210     1110      910
            100-40      880      980      880     1080
Maine       100-10      600      800      800      750
```

This example returns two rows with the lowest data values in col2 (Actual, Qtr2) per row group.

```
3- BOTTOM ("market", 2, @DATACOL(2))
```

This command returns the following rows:

```
Maine       100-20      734       334       734      534
            100-30      324       321       235      278
```

Note: <TOP and <BOTTOM put an upper limit on the number of (qualified) rows returned after all restrictions are applied. This upper limit is equal to the number of rows in the <TOP plus the number of rows in the <BOTTOM commands.

How Other Report Configurations Affect Row Specifications

When using the <TOP and <BOTTOM commands, be aware that some other commands affect their operation. In particular, Essbase treats the <SUPPMISSING, <SUPZEROS, and <SUPEMPTYROWS options as restrictions and applies them to the extracted rows along with the <RESTRICT command restrictions. Essbase applies these optional restrictions to the data rows before processing the <TOP or <BOTTOM commands, and before applying an <ORDERBY command.

Using TOP and BOTTOM with Formatting Commands

Whenever a formatting command occurs in a report, it is appended to the member that follows it. For example, in this sequence, {UCOLUMNS}, the underline columns command is appended internally to the member that comes next. In Script 1, it is appended to the row member that can be described as “first child of market, assuming FLORIDA.”

```
SCRIPT 1                 SCRIPT 2
....                     ....
< ROW MARKET             {UCOL}
{UCOL }                  < FLORIDA    (row member)
<ICHILDREN MARKET
< TOP ....               < BOTTOM ....
```

Script 2, appends {UCOLUMNS} to the row member FLORIDA. Essbase executes {UCOLUMNS} whenever it encounters a row that has row member FLORIDA. If the TOP or BOTTOM command returns a row that does not contain FLORIDA, the formatting commands appended to the rows are never executed.

Because of this, it is a good idea to place all general formatting commands before a <COLUMN command, or a command that expands into column members. This guarantees that the formatting is executed. However, you should not use formatting commands that work on rows, because these rows may never be picked up by the <TOP or <BOTTOM command. Also avoid using <SAVEROW and <CALCULATE ROW with the <TOP and <BOTTOM commands.

Converting Data to a Different Currency

If your database has a currency partition, you can calculate currency conversions in report scripts. Use the <CURRENCY command to set the output currency and currency type. Use the <CURHEADING command to display the currency conversion heading.

Note: Currency conversion is not supported across transparent partitions.

For information about creating a currency conversion application, see Chapter 12, "Designing and Building Currency Conversion Applications."

For the syntax and definitions of Report Writer commands, see the *Technical Reference* in the `docs` directory.

Generating Reports Using the C, Visual Basic, and Grid APIs

Use the following table to determine the report API calls that you can make:

Task	C API Function	Visual Basic API Function	C Grid API Function
Start sending a report specification to the active database.	ESSBEGINREPORT	ESBBEGINREPORT	ESSGBEGINREPORT
Mark the end of a report specification being sent to the active database.	ESSENDREPORT	ESBENDREPORT	N/A
Send a report specification to the active database as a single string.	ESSREPORT	ESBREPORT	N/A
Send a report specification to the active database from a file.	ESSREPORTFILE	ESBREPORTFILE	ESSGREPORTFILE

See the *API Reference* in your `docs` directory for syntax and descriptions of these API functions.

Chapter

36

Examples of Report Scripts

This chapter includes report scripts that demonstrate report procedures and formats that are most frequently required in business settings. If you examine the techniques in these scripts and the resulting output, you can adapt them for use in your own reports.

The samples use both the Demo Basic and Sample Basic databases provided with your Hyperion Essbase server. Each sample identifies the database used. The scripts for these examples are available in your `\ESSBASE\APP\DEMO\BASIC` directory or your `\ESSBASE\APP\SAMPLE\BASIC` directory.

The sample reports in this chapter demonstrate the following techniques:

- "Sample 1: Creating a Different Format for Each Page" on page 1020
- "Sample 2: Handling Missing Values" on page 1023
- "Sample 3: Nesting Columns" on page 1025
- "Sample 4: Grouping Rows" on page 1027
- "Sample 5: Reporting on Different Combinations of Data" on page 1032
- "Sample 6: Formatting Different Combinations of Data" on page 1035
- "Sample 7: Using Aliases" on page 1038
- "Sample 8: Creating Custom Headings and % Characters" on page 1041
- "Sample 9: Creating Custom Page Headings" on page 1046
- "Sample 10: Using Formulas" on page 1049
- "Sample 11: Placing Two Page Layouts on the Same Page" on page 1052
- "Sample 12: Formatting for Data Export" on page 1054
- "Sample 13: Creating Asymmetric Columns" on page 1056
- "Sample 14: Calculating Columns" on page 1057

For fundamental information about reports and report scripts, see Chapter 34, "Quick Start to Report Scripts." For detailed information about using Report Writer commands to write reports and reports scripts, see Chapter 35, "Developing Report Scripts." For the syntax and usage of each Report Writer command, see the *Technical Reference* in the `docs` directory.

Sample 1: Creating a Different Format for Each Page

This sample report contains data for Actual Sales. Each report page shows a different Product. The report lists products on the same page until the maximum page length is reached. To place each Product on a separate page, you must use the PAGEONDIMENSION format command, as shown in Sample 2.

Because none of the cities in South sell Stereo or Compact_Disc, the data values indicate #MISSING. You can represent missing values by suppressing the row or substituting a replacement text string, such as N/A. See "Sample 2: Handling Missing Values" on page 1023 for an example of substituting page breaks and labels for missing values.

```
                                Sales Actual Stereo

                             Qtr1     Qtr2     Qtr3     Qtr4
                         ======== ======== ======== ========
East                        7,839    7,933    7,673   10,044
West                       11,633   11,191   11,299   14,018
South                    #Missing #Missing #Missing #Missing
Market                     19,472   19,124   18,972   24,062

                              Sales Actual Compact_Disc

                            Qtr1     Qtr2     Qtr3     Qtr4
                         ======== ======== ======== ========
East                       10,293    9,702    9,965   11,792
West                       14,321   14,016   14,328   17,247
South                    #Missing #Missing #Missing #Missing
Market                     24,614   23,718   24,293   29,039

                                  Sales Actual Audio

                             Qtr1     Qtr2     Qtr3     Qtr4
                         ======== ======== ======== ========
East                       18,132   17,635   17,638   21,836
West                       25,954   25,207   25,627   31,265
South                    #Missing #Missing #Missing #Missing
Market                     44,086   42,842   43,265   53,101
```

Use the following script to create Sample 1:

```
<PAGE (Accounts, Scenario, Product)
Sales
Actual
<IDESCENDANTS Audio

    <COLUMN (Year)
    <CHILDREN Year

<ROW(Market)
<ICHILDREN Market
    !
```

The **!** report output command is required to generate the report.

Because the IDESCENDANTS selection command is used for Audio, the report selects all three members. Only a single member is selected from the other page dimensions, Sales and Actual. As a result, the script creates three report pages. They display as one long report page unless you use the PAGEONDIMENSION format command, as shown in Sample 2.

This report script, `ACTSALES.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 2: Handling Missing Values

This report has the same layout and member selection as Sample 1, and shows you how to use page breaks and labels for missing values.

```
                              Sales Actual Stereo

                         Qtr1     Qtr2     Qtr3     Qtr4
                     ======== ======== ======== ========
East                    7,839    7,933    7,673   10,044
West                   11,633   11,191   11,299   14,018
South                     N/A      N/A      N/A      N/A
  Market               19,472   19,124   18,972   24,062
```

```
                            Sales Actual Compact_Disc

                         Qtr1     Qtr2     Qtr3     Qtr4
                     ======== ======== ======== ========
East                   10,293    9,702    9,965   11,792
West                   14,321   14,016   14,328   17,247
South                     N/A      N/A      N/A      N/A
  Market               24,614   23,718   24,293   29,039
```

```
                               Sales Actual Audio

                         Qtr1     Qtr2     Qtr3     Qtr4
                     ======== ======== ======== ========
East                   18,132   17,635   17,638   21,836
West                   25,954   25,207   25,627   31,265
South                     N/A      N/A      N/A      N/A
  Market               44,086   42,842   43,265   53,101
```

Use the following script to create Sample 2:

```
<PAGE (Accounts, Scenario, Product)
Sales
Actual
<IDESCENDANTS Product
{ PAGEONDIMENSION Product }
{ MISSINGTEXT "N/A" }

      <COLUMN (Year)
      <CHILDREN Year

<ROW(Market)
<ICHILDREN Market
    !
```

The PAGEONDIMENSION format command creates a page break whenever a member from the specified dimension changes. Because the report selects eight Product members, this creates an eight-page report.

The MISSINGTEXT format command substitutes any strings enclosed within double quotes into the `#MISSING` string. To suppress missing values, use the SUPMISSINGROWS command.

You can also combine format commands within one set of braces:

```
{ PAGEONDIMENSION Product MISSINGTEXT "N/A" }
```

This report script, `MISS_LBL.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 3: Nesting Columns

Each page produced by this report sample contains Sales information for a given Market. The report has two groups of columns across the page. The Actual and Budget members are the nested column group below Year members.

Note that the Actual and Budget members are on the same line in the report. You can put multiple commands on one line, but report commands are easier to read if they are spread out.

```
                                      Sales East

                     Jan             Feb             Mar             Qtr1
               Actual  Budget  Actual  Budget  Actual  Budget  Actual  Budget
              ======= ======= ======= ======= ======= ======= ======= =======


Stereo          2,788   2,950   2,482   2,700   2,569   2,700   7,839   8,350
Compact_Disc    3,550   3,450   3,285   3,250   3,458   3,250  10,293   9,950
  Audio         6,338   6,400   5,767   5,950   6,027   5,950  18,132  18,300
Television      5,244   4,800   4,200   4,300   3,960   4,300  13,404  13,400
VCR             4,311   4,200   3,734   3,700   3,676   3,700  11,721  11,600
Camera          2,656   2,850   2,525   2,670   2,541   2,670   7,722   8,190
  Visual       12,211  11,850  10,459  10,670  10,177  10,670  32,847  33,190
    Product    18,549  18,250  16,226  16,620  16,204  16,620  50,979  51,490
```

```
                                      Sales West

                    Jan             Feb             Mar             Qtr1
              Actual  Budget  Actual  Budget  Actual  Budget  Actual  Budget
              ====== ======= ======= ======= ======= ======= ======= =======


Stereo         4,102   4,000   3,723   3,600   3,808   3,600  11,633  11,200
Compact_Disc   4,886   4,700   4,647   4,400   4,788   4,400  14,321  13,500
  Audio        8,988   8,700   8,370   8,000   8,596   8,000  25,954  24,700
Television     5,206   5,100   4,640   4,600   4,783   4,600  14,629  14,300
VCR            4,670   4,650   4,667   4,200   4,517   4,200  13,854  13,050
Camera         3,815   4,050   3,463   3,750   3,478   3,750  10,756  11,550
  Visual      13,691  13,800  12,770  12,550  12,778  12,550  39,239  38,900
    Product   22,679  22,500  21,140  20,550  21,374  20,550  65,193  63,600
```

```
                                   Sales South

                  Jan             Feb              Mar             Qtr1
              Actual  Budget  Actual  Budget  Actual  Budget  Actual  Budget
            ======== ======= ======= ======= ======= ======= ======= =======

Television     3,137   3,400   2,929   3,100   2,815   3,100   8,881   9,600
VCR            3,225   3,400   3,206   3,100   3,120   3,100   9,551   9,600
Camera         2,306   2,400   2,167   2,400   2,168   2,400   6,641   7,200
  Visual       8,668   9,200   8,302   8,600   8,103   8,600  25,073  26,400
    Product    8,668   9,200   8,302   8,600   8,103   8,600  25,073  26,400
```

```
                                  Sales Market

                 Jan             Feb              Mar             Qtr1
             Actual  Budget  Actual  Budget  Actual  Budget  Actual  Budget
            ======= ======= ======= ======= ======= ======= ======= =======

Stereo         6,890   6,950   6,205   6,300   6,377   6,300  19,472  19,550
Compact_Disc  8,436   8,150   7,932   7,650   8,246   7,650  24,614  23,450
  Audio      15,326  15,100  14,137  13,950  14,623  13,950  44,086  43,000
Television   13,587  13,300  11,769  12,000  11,558  12,000  36,914  37,300
VCR          12,206  12,250  11,607  11,000  11,313  11,000  35,126  34,250
Camera        8,777   9,300   8,155   8,820   8,187   8,820  25,119  26,940
  Visual     34,570  34,850  31,531  31,820  31,058  31,820  97,159  98,490
    Product  49,896  49,950  45,668  45,770  45,681  45,770 141,245 141,490
```

Use the following script to create Sample 3:

```
<PAGE (Accounts, Market)
Sales
<ICHILDREN Market
{ PAGEONDIMENSION Market }
{ SUPMISSINGROWS }
      <COLUMN (Year, Scenario)
      <ICHILDREN Qtr1
       Actual Budget
<ROW(Product)
<IDESCENDANTS Product
    !
```

The report selects four Markets because the <ICHILDREN command is applied to Market. Only Sales is selected from the other page dimension, so the report has four pages.

For the South, all the rows of Product data are not displayed. Recall that the cities in the South do not sell every Product. The report uses the SUPMISSINGROWS format command to suppress the output of any member rows with all missing values.

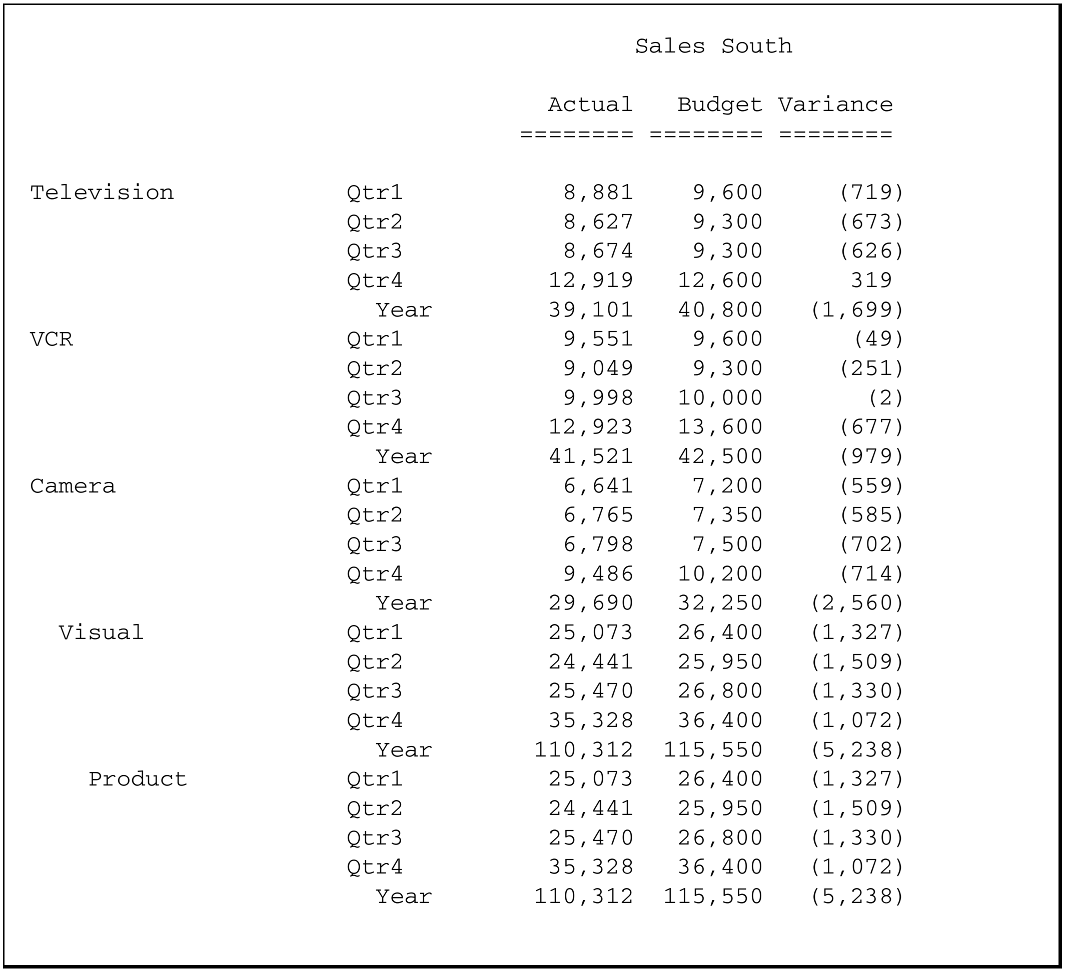
This report script, `COLGROUP.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 4: Grouping Rows

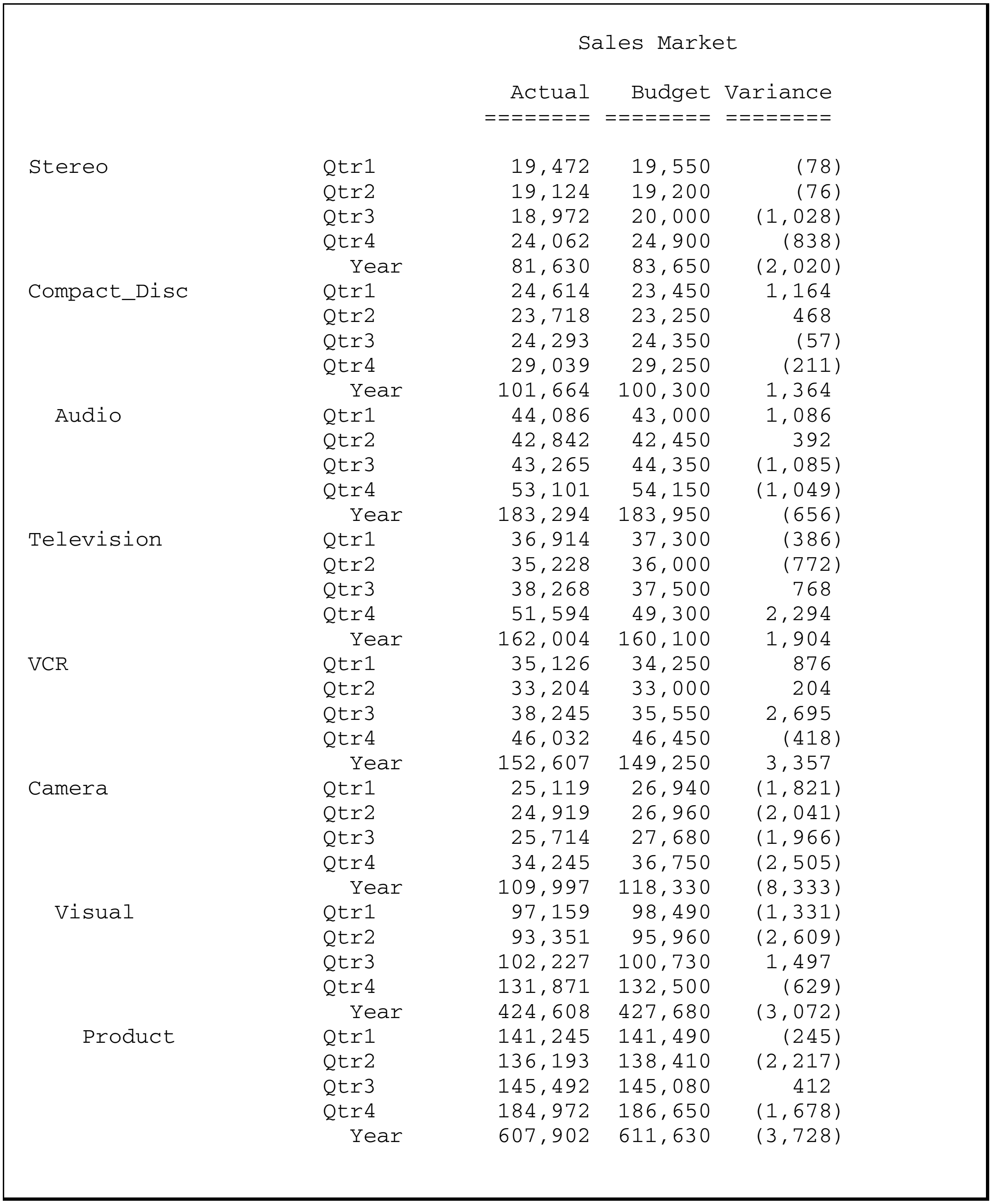
Each page of this report contains Sales information for a given Market. The report page contains members for both Product and Year as groups of rows down the page. This script creates a four-page report because the page dimensions and their member selections are the same as in Sample 3. The row and column layout is switched because the row and column dimensions are different. This section shows a representative part of the output.

```
                                           Sales East

                                    Actual   Budget Variance
                                  ======== ======== ========

Stereo                Qtr1           7,839    8,350     (511)
                      Qtr2           7,933    8,150     (217)
                      Qtr3           7,673    8,350     (677)
                      Qtr4          10,044   10,400     (356)
                        Year        33,489   35,250   (1,761)
Compact_Disc          Qtr1          10,293    9,950      343
                      Qtr2           9,702    9,750      (48)
                      Qtr3           9,965   10,050      (85)
                      Qtr4          11,792   12,550     (758)
                        Year        41,752   42,300     (548)
  Audio               Qtr1          18,132   18,300     (168)
                      Qtr2          17,635   17,900     (265)
                      Qtr3          17,638   18,400     (762)
                      Qtr4          21,836   22,950   (1,114)
                        Year        75,241   77,550   (2,309)
Television            Qtr1          13,404   13,400        4
                      Qtr2          12,115   12,900     (785)
                      Qtr3          15,014   14,200      814
                      Qtr4          17,861   17,300      561
                        Year        58,394   57,800      594
VCR                   Qtr1          11,721   11,600      121
                      Qtr2          10,999   11,100     (101)
                      Qtr3          13,217   11,800    1,417
                      Qtr4          14,386   14,900     (514)
                        Year        50,323   49,400      923
Camera                Qtr1           7,722    8,190     (468)
                      Qtr2           7,581    8,210     (629)
                      Qtr3           8,181    8,630     (449)
                      Qtr4          10,853   11,550     (697)
                        Year        34,337   36,580   (2,243)
  Visual              Qtr1          32,847   33,190     (343)
                      Qtr2          30,695   32,210   (1,515)
                      Qtr3          36,412   34,630    1,782
                      Qtr4          43,100   43,750     (650)
                        Year       143,054  143,780     (726)
    Product           Qtr1          50,979   51,490     (511)
                      Qtr2          48,330   50,110   (1,780)
                      Qtr3          54,050   53,030    1,020
                      Qtr4          64,936   66,700   (1,764)
                        Year       218,295  221,330   (3,035)
```

```
                                    Sales West

                               Actual   Budget Variance
                             ======== ======== ========

Stereo                 Qtr1     11,633   11,200      433
                       Qtr2     11,191   11,050      141
                       Qtr3     11,299   11,650     (351)
                       Qtr4     14,018   14,500     (482)
                         Year   48,141   48,400     (259)
Compact_Disc           Qtr1     14,321   13,500      821
                       Qtr2     14,016   13,500      516
                       Qtr3     14,328   14,300       28
                       Qtr4     17,247   16,700      547
                         Year   59,912   58,000    1,912
  Audio                Qtr1     25,954   24,700    1,254
                       Qtr2     25,207   24,550      657
                       Qtr3     25,627   25,950     (323)
                       Qtr4     31,265   31,200       65
                         Year  108,053  106,400    1,653
Television             Qtr1     14,629   14,300      329
                       Qtr2     14,486   13,800      686
                       Qtr3     14,580   14,000      580
                       Qtr4     20,814   19,400    1,414
                         Year   64,509   61,500    3,009
VCR                    Qtr1     13,854   13,050      804
                       Qtr2     13,156   12,600      556
                       Qtr3     15,030   13,750    1,280
                       Qtr4     18,723   17,950      773
                         Year   60,763   57,350    3,413
Camera                 Qtr1     10,756   11,550     (794)
                       Qtr2     10,573   11,400     (827)
                       Qtr3     10,735   11,550     (815)
                       Qtr4     13,906   15,000   (1,094)
                         Year   45,970   49,500   (3,530)
  Visual               Qtr1     39,239   38,900      339
                       Qtr2     38,215   37,800      415
                       Qtr3     40,345   39,300    1,045
                       Qtr4     53,443   52,350    1,093
                         Year  171,242  168,350    2,892
    Product            Qtr1     65,193   63,600    1,593
                       Qtr2     63,422   62,350    1,072
                       Qtr3     65,972   65,250      722
                       Qtr4     84,708   83,550    1,158
                         Year  279,295  274,750    4,545
```



```
                                          Sales South

                                    Actual   Budget Variance
                                  ======== ======== ========

Television            Qtr1           8,881    9,600     (719)
                      Qtr2           8,627    9,300     (673)
                      Qtr3           8,674    9,300     (626)
                      Qtr4          12,919   12,600      319
                        Year        39,101   40,800   (1,699)
VCR                   Qtr1           9,551    9,600      (49)
                      Qtr2           9,049    9,300     (251)
                      Qtr3           9,998   10,000       (2)
                      Qtr4          12,923   13,600     (677)
                        Year        41,521   42,500     (979)
Camera                Qtr1           6,641    7,200     (559)
                      Qtr2           6,765    7,350     (585)
                      Qtr3           6,798    7,500     (702)
                      Qtr4           9,486   10,200     (714)
                        Year        29,690   32,250   (2,560)
  Visual              Qtr1          25,073   26,400   (1,327)
                      Qtr2          24,441   25,950   (1,509)
                      Qtr3          25,470   26,800   (1,330)
                      Qtr4          35,328   36,400   (1,072)
                        Year       110,312  115,550   (5,238)
    Product           Qtr1          25,073   26,400   (1,327)
                      Qtr2          24,441   25,950   (1,509)
                      Qtr3          25,470   26,800   (1,330)
                      Qtr4          35,328   36,400   (1,072)
                        Year       110,312  115,550   (5,238)
```



```
                                         Sales Market

                                    Actual   Budget Variance
                                  ======== ======== ========

Stereo                Qtr1          19,472   19,550      (78)
                      Qtr2          19,124   19,200      (76)
                      Qtr3          18,972   20,000   (1,028)
                      Qtr4          24,062   24,900     (838)
                        Year        81,630   83,650   (2,020)
Compact_Disc          Qtr1          24,614   23,450    1,164
                      Qtr2          23,718   23,250      468
                      Qtr3          24,293   24,350      (57)
                      Qtr4          29,039   29,250     (211)
                        Year       101,664  100,300    1,364
  Audio               Qtr1          44,086   43,000    1,086
                      Qtr2          42,842   42,450      392
                      Qtr3          43,265   44,350   (1,085)
                      Qtr4          53,101   54,150   (1,049)
                        Year       183,294  183,950     (656)
Television            Qtr1          36,914   37,300     (386)
                      Qtr2          35,228   36,000     (772)
                      Qtr3          38,268   37,500      768
                      Qtr4          51,594   49,300    2,294
                        Year       162,004  160,100    1,904
VCR                   Qtr1          35,126   34,250      876
                      Qtr2          33,204   33,000      204
                      Qtr3          38,245   35,550    2,695
                      Qtr4          46,032   46,450     (418)
                        Year       152,607  149,250    3,357
Camera                Qtr1          25,119   26,940   (1,821)
                      Qtr2          24,919   26,960   (2,041)
                      Qtr3          25,714   27,680   (1,966)
                      Qtr4          34,245   36,750   (2,505)
                        Year       109,997  118,330   (8,333)
  Visual              Qtr1          97,159   98,490   (1,331)
                      Qtr2          93,351   95,960   (2,609)
                      Qtr3         102,227  100,730    1,497
                      Qtr4         131,871  132,500     (629)
                        Year       424,608  427,680   (3,072)
    Product           Qtr1         141,245  141,490     (245)
                      Qtr2         136,193  138,410   (2,217)
                      Qtr3         145,492  145,080      412
                      Qtr4         184,972  186,650   (1,678)
                        Year       607,902  611,630   (3,728)
```

Use the following script to create Sample 4:

```
<PAGE (Accounts, Market)
Sales
<ICHILDREN Market
{ PAGEONDIMENSION Market }
{ SUPMISSINGROWS }

      <COLUMN (Scenario)
      <CHILDREN Scenario

<ROW(Product3, Year)
<ICHILDREN Year
<IDESCENDANTS Product
    !
```

This report script, `ROWGROUP.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 5: Reporting on Different Combinations of Data

Each page represents a different combination of Product, Market, and Budget data. The total number of pages is determined by the number of Market and Product members. This section shows a representative part of the output.

Some data values have four decimal places. The number of decimal places, by default, is output to the true number of decimal values of the data cell. "Sample 6: Formatting Different Combinations of Data" on page 1035 uses the DECIMAL format command to define a specific number of places.

The member selection commands select three Product members and fourteen Market members. This produces a 42-page report. The number of report pages is determined by multiplying the number of members selected from each page dimension.

```
                                 Budget Audio New_York

                         Qtr1     Qtr2     Qtr3     Qtr4     Year
                     ======== ======== ======== ======== ========

Sales                   6,400    6,400    6,700    8,350   27,850
Cost_of_Goods_Sold      3,012    3,012    3,146    3,973   13,143
  Margin                3,388    3,388    3,554    4,377   14,707
Marketing                 525      515      475      555    2,070
Payroll                 1,950    1,950    1,950    1,950    7,800
Miscellaneous               0        0        0        0        0
  Total_Expenses        2,475    2,465    2,425    2,505    9,870
    Profit                913      923    1,129    1,872    4,837
    Profit_%               14       14       17       22       17
    Margin_%               53       53       53       52       53
```

```
                                 Budget Audio Boston

                         Qtr1     Qtr2     Qtr3     Qtr4     Year
                     ======== ======== ======== ======== ========

Sales                   6,050    5,750    5,900    7,350   25,050
Cost_of_Goods_Sold      2,829    2,695    2,762    3,413   11,699
  Margin                3,221    3,055    3,138    3,937   13,351
Marketing                 410      400      400      520    1,730
Payroll                 1,590    1,590    1,590    1,590    6,360
Miscellaneous               0        0        0        0        0
  Total_Expenses        2,000    1,990    1,990    2,110    8,090
    Profit              1,221    1,065    1,148    1,827    5,261
    Profit_%               20       19       19       25       21
    Margin_%               53       53       53       54       53
```

```
                                   Budget Product Market

                              Qtr1     Qtr2     Qtr3     Qtr4     Year
                          ======== ======== ======== ======== ========

Sales                      141,490  138,410  145,080  186,650  611,630
Cost_of_Goods_Sold          55,860   54,579   57,379   73,276  241,093
  Margin                    85,630   83,831   87,702  113,374  370,537
Marketing                   10,555   10,680   10,780   13,915   45,930
Payroll                     43,234   43,248   43,248   43,248  172,978
Miscellaneous                    0        0        0        0        0
  Total_Expenses            53,789   53,928   54,028   57,163  218,908
    Profit                  31,841   29,903   33,674   56,211  151,629
    Profit_%                    23       22       23       30       25
    Margin_%                    61       61       60       61       61
```

Use the following script to create Sample 5:

```
<PAGE (Scenario, Product, Market)
Budget
<ICHILDREN Product
<IDESCENDANTS Market
{ PAGEONDIMENSION Product }    // New page at each new Product
{ PAGEONDIMENSION Market }     // New page at each new Market
       <COLUMN (Year)
       <ICHILDREN Year

<ROW(Accounts)
<DESCENDANTS Accounts
    !
```

This report script, `COMBO1.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 6: Formatting Different Combinations of Data

This report uses the same layout and member selection as Sample 5, and adds more formatting in the report body. Note the use of line formatting.

```
                         Budget Audio New_York

                    Qtr1     Qtr2     Qtr3     Qtr4     Year
                ======== ======== ======== ======== ========


Sales              6,400    6,400    6,700    8,350   27,850
Cost_of_Goods_Sold 3,012    3,012    3,146    3,973   13,143
                -------- -------- -------- -------- --------
  Margin           3,388    3,388    3,554    4,377   14,707

Marketing            525      515      475      555    2,070
Payroll            1,950    1,950    1,950    1,950    7,800
Miscellaneous          0        0        0        0        0
                -------- -------- -------- -------- --------
  Total_Expenses   2,475    2,465    2,425    2,505    9,870

    Profit           913      923    1,129    1,872    4,837
                ======== ======== ======== ======== ========
    Profit_%       14.27    14.42    16.85    22.42    17.37
    Margin_%       52.94    52.94    53.04    52.42    52.81
```

```
                                 Budget Audio Boston

                          Qtr1     Qtr2     Qtr3     Qtr4     Year
                        ======== ======== ======== ======== ========

Sales                      6,050    5,750    5,900    7,350   25,050
Cost_of_Goods_Sold         2,829    2,695    2,762    3,413   11,699
                        -------- -------- -------- -------- --------
  Margin                   3,221    3,055    3,138    3,937   13,351

Marketing                    410      400      400      520    1,730
Payroll                    1,590    1,590    1,590    1,590    6,360
Miscellaneous                  0        0        0        0        0
                        -------- -------- -------- -------- --------
  Total_Expenses           2,000    1,990    1,990    2,110    8,090

    Profit                 1,221    1,065    1,148    1,827    5,261
                        ======== ======== ======== ======== ========
    Profit_%               20.18    18.52    19.46    24.86    21.00
    Margin_%               53.24    53.13    53.19    53.56    53.30
```

Use the following script to create Sample 6:

```
<PAGE (Scenario, Product, Market)
{ PAGEONDIMENSION Product PAGEONDIMENSION Market }
Budget
<ICHILDREN Product
<IDESCENDANTS Market
      <COLUMN (Year)
      <ICHILDREN Year
<ROW(Accounts)
{ SUPBRACKETS DECIMAL 0 }
Sales
Cost_of_Goods_Sold
{ UDATA "-" }      //line formatting command
Margin
{ SKIP }
Marketing
Payroll
Miscellaneous
{ UDATA "-" }      //line formatting command
```

```
Total_Expenses
{ SKIP }
Profit
{ UDATA DECIMAL 2 }    //line formatting command
Profit_%
Margin_%
    !
```

Format commands apply to members that follow the commands. The report begins each new page with the formats in place at the end of the previous report page. For example, if a report page ends with two decimal places, the following page begins with two decimal places. This report demonstrates the use of several important format commands:

- DECIMAL—The script for this report specifies the DECIMAL 0 format command before the Sales member.
- SUPBRACKETS—By default, negative numbers are enclosed in brackets, (). The SUPBRACKETS format command causes negative numbers to be output with a minus sign.
- UDATA—The UDATA command places underline characters under data columns. The character is specified within double quotes. The default is a double underline.

This report script, `COMBO2.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 7: Using Aliases

This report outputs members in the middle of a page and uses aliases or alternate names. The default row member indentation is turned off.

```
                        Stereo Market

       Qtr4                                 Year
  Actual   Budget                     Actual   Budget
======== ========                    ======== ========


  24,062   24,900 Sales                81,630   83,650
  13,937   14,442 COGS                 47,654   48,517
-------- --------                    -------- --------
  10,125   10,458 Margin               33,976   35,133

   1,438    1,600 Marketing             4,933    5,465
   7,110    6,840 Payroll              28,440   27,360
    -200        0 Misc.                  -143        0
-------- --------                    -------- --------
   8,348    8,440 Total_Expenses       33,230   32,825

   1,777    2,018 Profit                  746    2,308
======== ========                    ======== ========
    7.39     8.10 Profit_%               0.91     2.76
   42.08    42.00 Margin_%              41.62    42.00
```

```
                    Compact_Disc Market

       Qtr4                                Period
  Actual   Budget                      Actual   Budget
======== ========                    ======== ========


  29,039   29,250 Sales               101,664  100,300
  10,830   11,115 COGS                 38,120   38,114
-------- --------                    -------- --------
  18,209   18,135 Margin               63,544   62,186
   1,669    1,780 Marketing             6,067    5,975
   5,721    5,415 Payroll              22,200   21,660
    -226        0 Misc.                    97        0
-------- --------                    -------- --------
   7,164    7,195 Total_Expenses       28,364   27,635

  11,045   10,940 Profit               35,180   34,551
======== ========                    ======== ========
   38.04    37.40 Profit_%              34.60    34.45
   62.71    62.00 Margin_%              62.50    62.00
```

Use the following script to create Sample 7:

```
<PAGE (Product, Market)
{ PAGEONDIMENSION Product }
{ PAGEONDIMENSION Market }
<IDESCENDANTS Product
{ DECIMAL 0 }
<SYM

     <COLUMN (Year, Scenario)
     Qtr4 Year
     Actual Budget
<ROW(Accounts)
{ SUPBRACKETS OUTALTNAMES NOINDENTGEN ORDER 1,2,0,3,4 }
Sales Cost_of_Goods_Sold
{ UDATA "-" }
Margin
{ SKIP }
Marketing Payroll Miscellaneous
{ UDATA "-" }
```

```
Total_Expenses
{ SKIP }
Profit
{ UDATA DECIMAL 2 }
Profit_%
Margin_%
    !
```

The SYM command forces the report to output symmetric column groups. The default is to display two columns: one for Qtr4 Actual and one for Year Budget. Because the report calls for Actual and Budget under both Qtr4 and Year, the SYM command is required. Alternatively, repeat the Actual and Budget names under Qtr4 and Year.

The OUTALTNAMES format command causes the report to use aliases or alternate names instead of member names.

The NOINDENTGEN format command causes row members to not be indented. By default, members are indented two spaces for each level.

The ORDER command moves specified output columns to new locations. The row name is considered column 0.

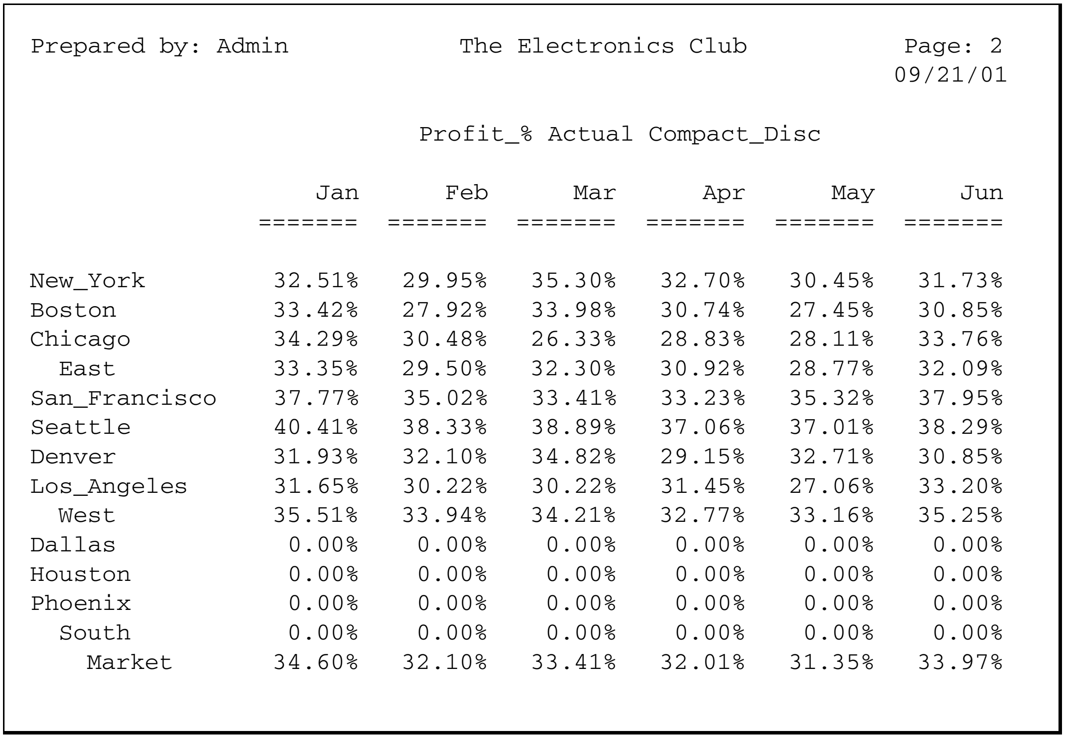
The FIXCOLUMNS format command restricts the number of output columns. Reports often require both ORDER and FIXCOLUMNS. You can use ORDER to remove unwanted columns, and FIXCOLUMNS to stop these columns from displaying after the report columns.

This report script, `MIDDLE.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 8: Creating Custom Headings and % Characters

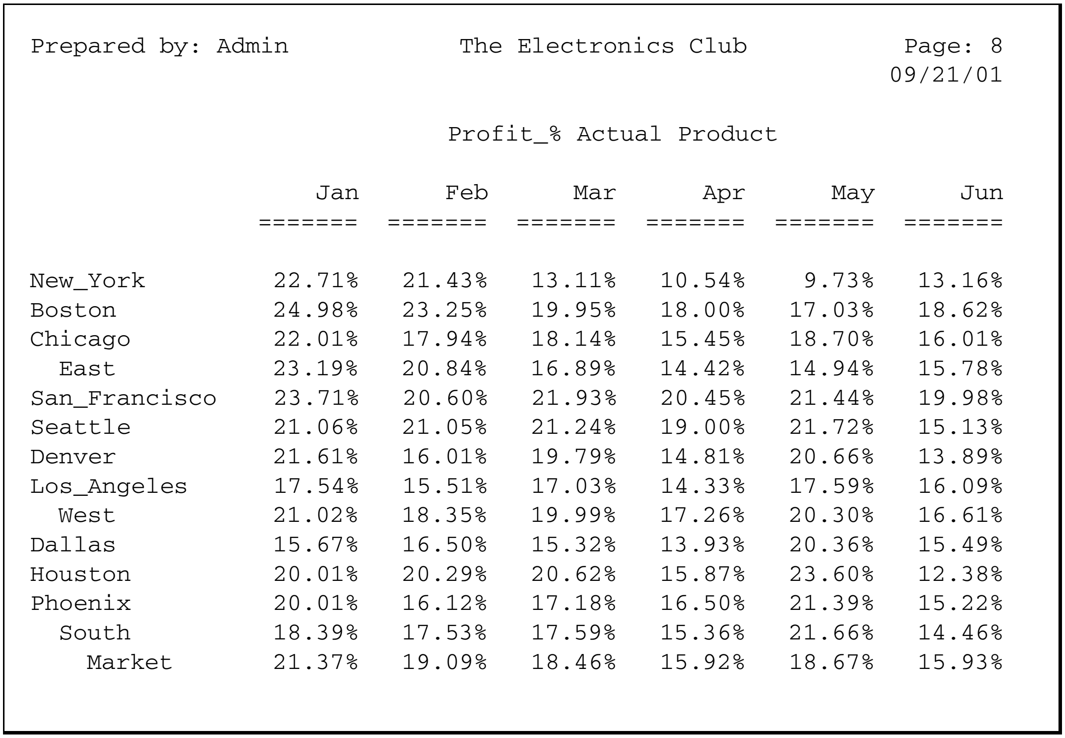
This report displays custom headings and percent sign (%) characters after each data value. This section shows a representative part of the output.

```
Prepared by: Admin            The Electronics Club          Page: 1
                                                            09/21/01

                              Profit_% Actual Stereo

                     Jan       Feb       Mar       Apr       May       Jun
                 =======   =======   =======   =======   =======   =======

New_York           1.43%   -10.00%    -3.51%    -2.22%     1.14%    -6.18%
Boston            -0.34%    -2.51%    -4.44%    -4.89%    -7.02%   -13.15%
Chicago           -0.65%    -0.72%    -2.28%    -3.53%    -6.33%   -10.79%
  East             0.18%    -4.47%    -3.39%    -3.41%    -3.60%    -9.70%
San_Francisco      1.43%    -1.87%     4.42%     2.15%    -1.26%     0.66%
Seattle            0.95%    -5.66%     1.42%    -6.82%   -11.47%   -12.34%
Denver             3.03%    -1.11%    -5.88%    -6.52%    -5.17%   -13.83%
Los_Angeles       -1.50%    -3.94%    -2.86%    -3.29%     3.12%    -2.51%
  West             0.98%    -2.95%    -0.13%    -2.81%    -2.62%    -5.61%
Dallas             0.00%     0.00%     0.00%     0.00%     0.00%     0.00%
Houston            0.00%     0.00%     0.00%     0.00%     0.00%     0.00%
Phoenix            0.00%     0.00%     0.00%     0.00%     0.00%     0.00%
  South            0.00%     0.00%     0.00%     0.00%     0.00%     0.00%
    Market         0.65%    -3.56%    -1.44%    -3.06%    -3.03%    -7.29%
```



```
Prepared by: Admin           The Electronics Club          Page: 2
                                                          09/21/01

                        Profit_% Actual Compact_Disc

                   Jan      Feb      Mar      Apr      May      Jun
               ======= ======= ======= ======= ======= =======

New_York        32.51%   29.95%   35.30%   32.70%   30.45%   31.73%
Boston          33.42%   27.92%   33.98%   30.74%   27.45%   30.85%
Chicago         34.29%   30.48%   26.33%   28.83%   28.11%   33.76%
  East          33.35%   29.50%   32.30%   30.92%   28.77%   32.09%
San_Francisco   37.77%   35.02%   33.41%   33.23%   35.32%   37.95%
Seattle         40.41%   38.33%   38.89%   37.06%   37.01%   38.29%
Denver          31.93%   32.10%   34.82%   29.15%   32.71%   30.85%
Los_Angeles     31.65%   30.22%   30.22%   31.45%   27.06%   33.20%
  West          35.51%   33.94%   34.21%   32.77%   33.16%   35.25%
Dallas           0.00%    0.00%    0.00%    0.00%    0.00%    0.00%
Houston          0.00%    0.00%    0.00%    0.00%    0.00%    0.00%
Phoenix          0.00%    0.00%    0.00%    0.00%    0.00%    0.00%
  South          0.00%    0.00%    0.00%    0.00%    0.00%    0.00%
    Market      34.60%   32.10%   33.41%   32.01%   31.35%   33.97%
```

```
Prepared by: Admin            The Electronics Club           Page: 3
                                                            09/21/01

                               Profit_% Actual Audio

                    Jan       Feb       Mar       Apr       May       Jun
                ======= ======= ======= ======= ======= =======

New_York         19.35%    13.64%    18.64%    16.55%    16.70%    14.65%
Boston           18.34%    14.44%    18.94%    14.94%    12.14%    12.42%
Chicago          18.50%    16.67%    13.18%    14.12%    12.70%    13.74%
  East           18.76%    14.88%    17.09%    15.32%    14.05%    13.68%
San_Francisco    20.32%    17.38%    18.92%    18.03%    18.23%    20.57%
Seattle          23.36%    21.40%    23.37%    20.17%    18.82%    19.04%
Denver           18.36%    17.25%    18.88%    13.43%    15.84%    12.14%
Los_Angeles      17.15%    14.76%    15.44%    15.76%    15.10%    17.07%
  West           19.75%    17.53%    19.00%    16.88%    17.01%    17.52%
Dallas            0.00%     0.00%     0.00%     0.00%     0.00%     0.00%
Houston           0.00%     0.00%     0.00%     0.00%     0.00%     0.00%
Phoenix           0.00%     0.00%     0.00%     0.00%     0.00%     0.00%
  South           0.00%     0.00%     0.00%     0.00%     0.00%     0.00%
    Market       19.34%    16.45%    18.21%    16.24%    15.78%    15.96%
```



```
Prepared by: Admin            The Electronics Club          Page: 8
                                                           09/21/01

                             Profit_% Actual Product

                      Jan      Feb      Mar      Apr      May      Jun
                  =======  =======  =======  =======  =======  =======

New_York           22.71%   21.43%   13.11%   10.54%    9.73%   13.16%
Boston             24.98%   23.25%   19.95%   18.00%   17.03%   18.62%
Chicago            22.01%   17.94%   18.14%   15.45%   18.70%   16.01%
  East             23.19%   20.84%   16.89%   14.42%   14.94%   15.78%
San_Francisco      23.71%   20.60%   21.93%   20.45%   21.44%   19.98%
Seattle            21.06%   21.05%   21.24%   19.00%   21.72%   15.13%
Denver             21.61%   16.01%   19.79%   14.81%   20.66%   13.89%
Los_Angeles        17.54%   15.51%   17.03%   14.33%   17.59%   16.09%
  West             21.02%   18.35%   19.99%   17.26%   20.30%   16.61%
Dallas             15.67%   16.50%   15.32%   13.93%   20.36%   15.49%
Houston            20.01%   20.29%   20.62%   15.87%   23.60%   12.38%
Phoenix            20.01%   16.12%   17.18%   16.50%   21.39%   15.22%
  South            18.39%   17.53%   17.59%   15.36%   21.66%   14.46%
    Market         21.37%   19.09%   18.46%   15.92%   18.67%   15.93%
```

Use the following script to create Sample 8:

```
<PAGE (Accounts, Scenario, Product)
{ PAGEONDIMENSION Product }   // New page when Product changes
Profit_%
Actual
<IDESCENDANTS Product

      <COLUMN (Year)
      Jan Feb Mar Apr May Jun

<ROW(Market)

{ STARTHEADING
TEXT   1 "Prepared by:"
      14 "*USERNAME"
       C "The Electronics Club"
      65 "*PAGESTRING"
TEXT  65 "*DATE"
SKIP
ENDHEADING }

{ Decimal 2 AFTER "%" SUPBRACKETS }   // Place % at end and
    // suppress bracket
<IDESCENDANTS Market
      !
```

Each data value in the report has a percent sign, %. This label is defined with the AFTER "%" format command. You can specify any character within quotation marks.

This report has custom headings at the top of each page. All format commands specified between the STARTHEADING and ENDHEADING format commands are displayed at the top of each report page.

TEXT format commands define text labels. The report generator provides *dynamic* text with **options*. See the *Technical Reference* in the `docs` directory for a full list of the available options. This report uses the following options:

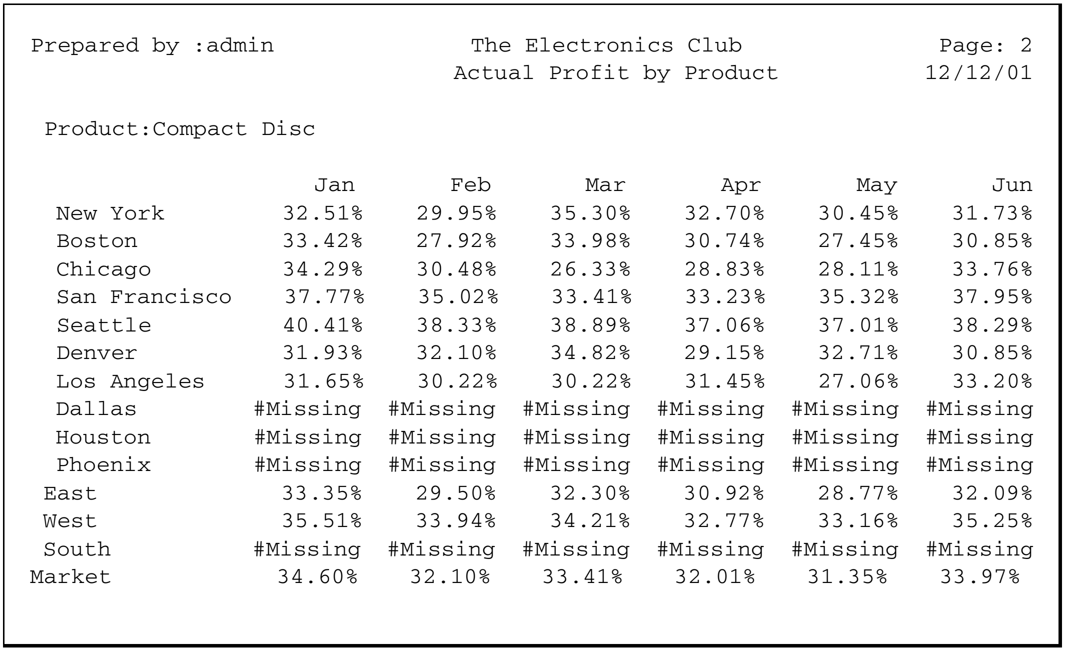
- *USERNAME, which outputs the user name used when connecting to Hyperion Essbase
- *PAGESTRING, which outputs the current page number of the report
- C, which centers the report title

This report script, `HEADING1.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

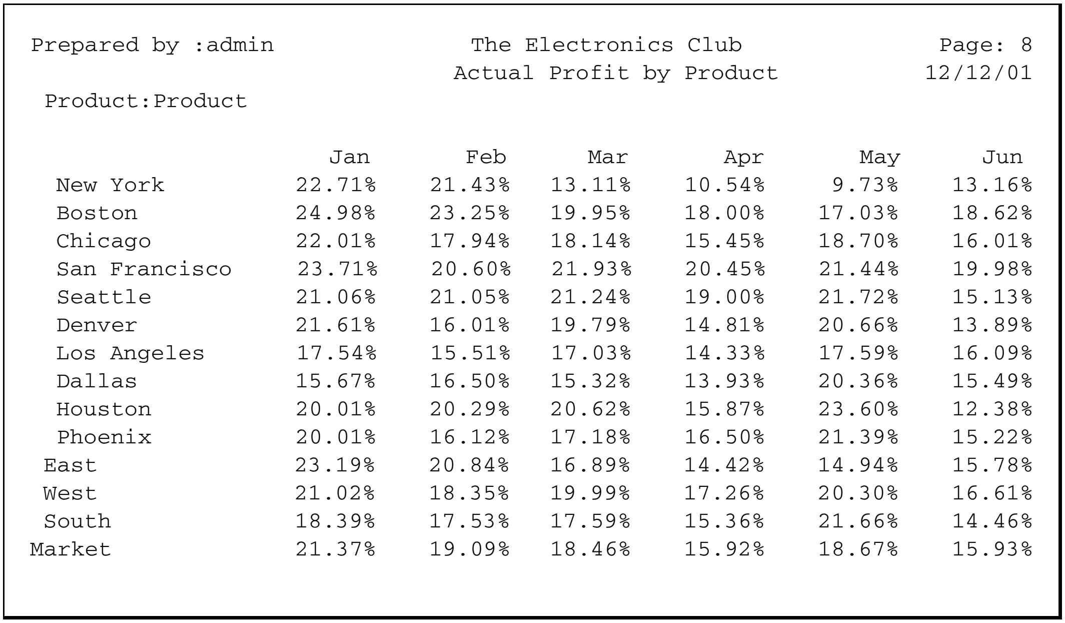
Sample 9: Creating Custom Page Headings

This report builds on Sample 8 by adding custom page headings. By default, page dimension members are output at the top center of a report page. This section shows a representative part of the output.

```
Prepared by :admin             The Electronics Club             Page: 1
                              Actual Profit by Product         12/12/01

 Product: Stereo

                    Jan       Feb       Mar       Apr       May       Jun

New York          1.43%   -10.00%    -3.51%    -2.22%     1.14%    -6.18%
Boston           -0.34%    -2.51%    -4.44%    -4.89%    -7.02%   -13.15%
Chicago          -0.65%    -0.72%    -2.28%    -3.53%    -6.33%   -10.79%
San Francisco     1.43%    -1.87%     4.42%     2.15%    -1.26%     0.66%
Seattle           0.95%    -5.66%     1.42%    -6.82%   -11.47%   -12.34%
Denver            3.03%    -1.11%    -5.88%    -6.52%    -5.17%   -13.83%
Los Angeles      -1.50%    -3.94%    -2.86%    -3.29%     3.12%    -2.51%
Dallas         #Missing  #Missing  #Missing  #Missing  #Missing  #Missing
Houston        #Missing  #Missing  #Missing  #Missing  #Missing  #Missing
Phoenix        #Missing  #Missing  #Missing  #Missing  #Missing  #Missing
 East             0.18%    -4.47%    -3.39%    -3.41%    -3.60%    -9.70%
 West             0.98%    -2.95%    -0.13%    -2.81%    -2.62%    -5.61%
 South         #Missing  #Missing  #Missing  #Missing  #Missing  #Missing
Market            0.65%    -3.56%    -1.44%    -3.06%    -3.03%    -7.29%
```



```
Prepared by :admin            The Electronics Club            Page: 2
                             Actual Profit by Product         12/12/01

 Product:Compact Disc

                       Jan       Feb       Mar       Apr       May       Jun
  New York          32.51%    29.95%    35.30%    32.70%    30.45%    31.73%
  Boston            33.42%    27.92%    33.98%    30.74%    27.45%    30.85%
  Chicago           34.29%    30.48%    26.33%    28.83%    28.11%    33.76%
  San Francisco     37.77%    35.02%    33.41%    33.23%    35.32%    37.95%
  Seattle           40.41%    38.33%    38.89%    37.06%    37.01%    38.29%
  Denver            31.93%    32.10%    34.82%    29.15%    32.71%    30.85%
  Los Angeles       31.65%    30.22%    30.22%    31.45%    27.06%    33.20%
  Dallas          #Missing  #Missing  #Missing  #Missing  #Missing  #Missing
  Houston         #Missing  #Missing  #Missing  #Missing  #Missing  #Missing
  Phoenix         #Missing  #Missing  #Missing  #Missing  #Missing  #Missing
 East               33.35%    29.50%    32.30%    30.92%    28.77%    32.09%
 West               35.51%    33.94%    34.21%    32.77%    33.16%    35.25%
 South            #Missing  #Missing  #Missing  #Missing  #Missing  #Missing
Market             34.60%    32.10%    33.41%    32.01%    31.35%    33.97%
```



```
Prepared by :admin            The Electronics Club            Page: 8
                             Actual Profit by Product         12/12/01
 Product:Product

                        Jan       Feb       Mar       Apr       May       Jun
  New York           22.71%    21.43%    13.11%    10.54%     9.73%    13.16%
  Boston             24.98%    23.25%    19.95%    18.00%    17.03%    18.62%
  Chicago            22.01%    17.94%    18.14%    15.45%    18.70%    16.01%
  San Francisco      23.71%    20.60%    21.93%    20.45%    21.44%    19.98%
  Seattle            21.06%    21.05%    21.24%    19.00%    21.72%    15.13%
  Denver             21.61%    16.01%    19.79%    14.81%    20.66%    13.89%
  Los Angeles        17.54%    15.51%    17.03%    14.33%    17.59%    16.09%
  Dallas             15.67%    16.50%    15.32%    13.93%    20.36%    15.49%
  Houston            20.01%    20.29%    20.62%    15.87%    23.60%    12.38%
  Phoenix            20.01%    16.12%    17.18%    16.50%    21.39%    15.22%
 East                23.19%    20.84%    16.89%    14.42%    14.94%    15.78%
 West                21.02%    18.35%    19.99%    17.26%    20.30%    16.61%
 South               18.39%    17.53%    17.59%    15.36%    21.66%    14.46%
Market               21.37%    19.09%    18.46%    15.92%    18.67%    15.93%
```

Use the following script to create Sample 9:

```
<PAGE (Accounts, Scenario, Product)
<IDESCENDANTS Product
<SORTLEVEL
{ PAGEONDIMENSION Product }
{ STARTHEADING
TEXT    1 "Prepared by:"
       14 "*USERNAME"
        C "The Electronics Club"
       65 "*PAGESTRING"
SUPPAGEHEADING
UNDERLINECHAR " "
TEXT    C "Actual Profit by Product"
       65 "*DATE"
TEXT    1 "Product:"
       10 "*PAGEHDR 3"
SKIP
ENDHEADING }
Profit_%
Actual

      <COLUMN (Year)
      Jan Feb Mar Apr May Jun
<ROW(Market)

{ DECIMAL 2 AFTER "%" SUPBRACKETS UNDERSCORECHAR " " }
{ INDENTGEN 1 }
<IDESCENDANTS Market
     !
```

The SUPPAGEHEADING format command suppresses the default page headings from output.

The *PAGEHDR command customizes the location of page member labels. The Sample 9 script uses page heading number 3, Product because this is the third page dimension.

You may have also noticed that member names do not have underscores. The UNDERSCORECHAR format command blanks out underscores.

Another difference is the underlining of column headings. The UNDERLINECHAR format command causes the underlining to character to change to the character in quotes.

The report rows are also sorted according to their levels in the database outline. Sort commands, such as SORTLEVEL, do not affect individual members selected in reports. Instead, these commands work in conjunction with member selection commands.

Note: You can use only one sort command in a report.

Sample 9 reverses the indentation of levels from previous reports. The INDENTGEN command indents members to the specified number of characters.

This report script, `HEADING2.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 10: Using Formulas

Column calculation formulas manipulate the column value of a particular row or a constant. In this report sample, each % column represents the quarterly values as a percent of Sales for the respective quarter. In addition, the Avg column represents an average value for the two quarters.

```
                                      Actual Product Market

                              Qtr1      %     Qtr2      %      Avg
                          ======== ====== ======== ====== ========

Sales                      141,245 100.00  136,193 100.00  138,719
Cost_of_Goods_Sold          58,104  41.14   56,281  41.32   57,193
  Margin                    83,141  58.86   79,912  58.68   81,527
Marketing                   11,211   7.94   11,302   8.30   11,257
Payroll                     43,817  31.02   43,827  32.18   43,822
Miscellaneous                  302   0.21    1,859   1.36    1,081
  Total_Expenses            55,330  39.17   56,988  41.84   56,159
    Profit                  27,811  19.69   22,924  16.83   25,368
    Profit_%                    20   0.01       17   0.01       18
    Margin_%                    59   0.04       59   0.04       59
```

Use the following script to create Sample 10:

```
// This report performs column calculations based on values in a
// report row.

<PAGE (Scenario, Product, Market)
Actual

      <COLUMN (Year)
      Qtr1 Qtr2

{ DECIMAL 2 3 4 }
{ NAMEWIDTH 22 WIDTH 7 3 4 }
{ ORDER 0 1 3 2 4 5 }

<ROW (Accounts)
{ SAVEROW } Sales
    !

{ CALCULATE COLUMN "%" = 1 % "Sales" 1 }
{ CALCULATE COLUMN "% " = 2 % "Sales" 2 }
{ CALCULATE COLUMN "Avg" = 1 + 2 / 2. }

<DESCENDANTS Accounts
    !
```

Note: You can include comments in the report by preceding the text with //. The Report Extractor ignores everything that follows the double slash. You can use comments to explain report processing.

The SAVEROW command reserves space for a row member that the CALCULATE COLUMN command calculates. In this case, the calculation affects SALES. The ! is required after the member name.

The CALCULATE COLUMN command allows column numbers, row names, or constants in formulas. You can read the first calculation this way: “% equals column 1 as a percent of Sales in column 1.”

Each calculated column label must be unique. Note how the second calculated column label has a blank space after the % sign.

To specify a constant, define a number followed by a period. You can use a constant in either a column or row calculation. The last column calculation takes the sum of columns 1 and 2 and divides by the value 2. This formula is interpreted as (1+2)/2, *not* 1 + (2/2.).

As noted in Sample 7, the ORDER command arranges columns in the specified order. By default, calculated columns are added to the end of existing columns retrieved from the database. In this example, columns 0–2 are automatically retrieved, based on selected members. Columns 3–5 are the calculated columns. The ORDER command applies to both retrieved and calculated columns.

This report script, `COLCALC1.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 11: Placing Two Page Layouts on the Same Page

This sample report has two different page layouts on the same page.

```
                               Year Profit_% Actual

                           East      West     South    Market
                      ========= ========= ========= =========

Stereo                   -0.52%     1.91%     0.00%     0.91%
Compact_Disc             32.60%    36.00%     0.00%    34.60%
  Audio                  17.86%    20.81%     0.00%    19.60%
Television               20.40%    16.57%    13.50%    17.21%
VCR                      30.81%    32.43%    33.70%    32.24%
Camera                   16.66%    21.66%    17.83%    19.07%
  Visual                 23.16%    23.56%    22.27%    23.09%
    Product              21.34%    22.50%    22.27%    22.04%

                                    Sales Actual Product

                          Qtr1     Qtr2     Qtr3     Qtr4     Year
                     ========= ======== ======== ======== ========

New_York               $18,631  $17,681  $19,923  $24,403  $80,638
Boston                 $15,812  $15,050  $16,716  $19,159  $66,737
Chicago                $16,536  $15,599  $17,411  $21,374  $70,920
  East                 $50,979  $48,330  $54,050  $64,936 $218,295
San_Francisco          $19,761  $19,019  $20,722  $24,807  $84,309
Seattle                $13,766  $13,546  $14,204  $19,034  $60,550
Denver                 $13,800  $13,588  $13,838  $18,232  $59,458
Los_Angeles            $17,866  $17,269  $17,208  $22,635  $74,978
  West                 $65,193  $63,422  $65,972  $84,708 $279,295
Dallas                 $ 9,226  $ 9,175  $ 9,481  $12,700  $40,582
Houston                $ 7,690  $ 7,363  $ 7,646  $10,785  $33,484
Phoenix                $ 8,157  $ 7,903  $ 8,343  $11,843  $36,246
  South                $25,073  $24,441  $25,470  $35,328 $110,312
    Market            $141,245 $136,193 $145,492 $184,972 $607,902
```

Use the following script to create Sample 11:

```
<PAGE (Year, Accounts, Scenario)

      <COLUMN (Market)
      <ICHILDREN Market

<ROW(Product)
<IDESCENDANTS Product

Actual
{ DECIMAL 2 WIDTH 10 SUPBRACKETS AFTER "%" }
Profit_%
    !

<PAGE (Accounts, Scenario, Product)
Actual
Sales
Product

      <COLUMN(Year)
      <ICHILDREN Year

<ROW(Market)
{ DECIMAL 0 After " " BEFORE "$" }
<IDESCENDANTS Market
    !
```

In a single report, you can select multiple dimension layouts and members. To define a multiple layout report, define reports as you normally would. Separate the commands with exclamation marks as shown above. Whenever the column, row, or page dimensions change between ! output commands, new headings are automatically generated to match the new layout.

The BEFORE format command places a character in front of data values. The AFTER format command turns off the percent signs from the first report layout.

This report script, `2LAYOUTS.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 12: Formatting for Data Export

This sample creates a report with a member name in each column. This format is required when you export Hyperion Essbase data to another product, such as an SQL database, with a flat file.

```
New York        Stereo           Sales              1000.0    950.0
New York        Stereo           Cost of Goods Sold  580.0    551.0
New York        Stereo           Margin              420.0    399.0
New York        Stereo           Marketing            80.0     80.0
New York        Stereo           Payroll             340.0    340.0
New York        Stereo           Miscellaneous         0.0      0.0
New York        Stereo           Total Expenses      420.0    420.0
New York        Stereo           Profit                0.0    -21.0
New York        Stereo           Profit %              0.0     -2.2
New York        Stereo           Margin %             42.0     42.0
New York        Compact Disc     Sales              1200.0   1150.0
New York        Compact Disc     Cost of Goods Sold  456.0    437.0
New York        Compact Disc     Margin              744.0    713.0
New York        Compact Disc     Marketing            95.0     95.0
New York        Compact Disc     Payroll             310.0    310.0
New York        Compact Disc     Miscellaneous         0.0      0.0
New York        Compact Disc     Total Expenses      405.0    405.0
New York        Compact Disc     Profit              339.0    308.0
New York        Compact Disc     Profit %             28.3     26.8
New York        Compact Disc     Margin %             62.0     62.0
New York        Audio            Sales              2200.0   2100.0
New York        Audio            Cost of Goods Sold 1036.0    988.0
New York        Audio            Margin             1164.0   1112.0
New York        Audio            Marketing           175.0    175.0
New York        Audio            Payroll             650.0    650.0
New York        Audio            Miscellaneous         0.0      0.0
New York        Audio            Total Expenses      825.0    825.0
New York        Audio            Profit              339.0    287.0
New York        Audio            Profit %             15.4     13.7
New York        Audio            Margin %             52.9     53.0
New York        Television       Sales              1800.0   1600.0
```

Use the following script to create Sample 12:

```
<PAGE(Scenario)

<COLUMN(Year)

<ROW (Market, Product, Accounts)
<CHILDREN East
<DESCENDANTS Product

{ DECIMAL 1
WIDTH 9
SUPBRACKETS
SUPCOMMA
MISSINGTEXT " "
UNDERSCORECHAR " "
SUPHEADING
NOINDENTGEN
SUPFEED
ROWREPEAT

Budget
      Jan Feb

<DESCENDANTS Accounts
    !
```

The ROWREPEAT command produces rows of data that have the member names repeat for each row dimension.

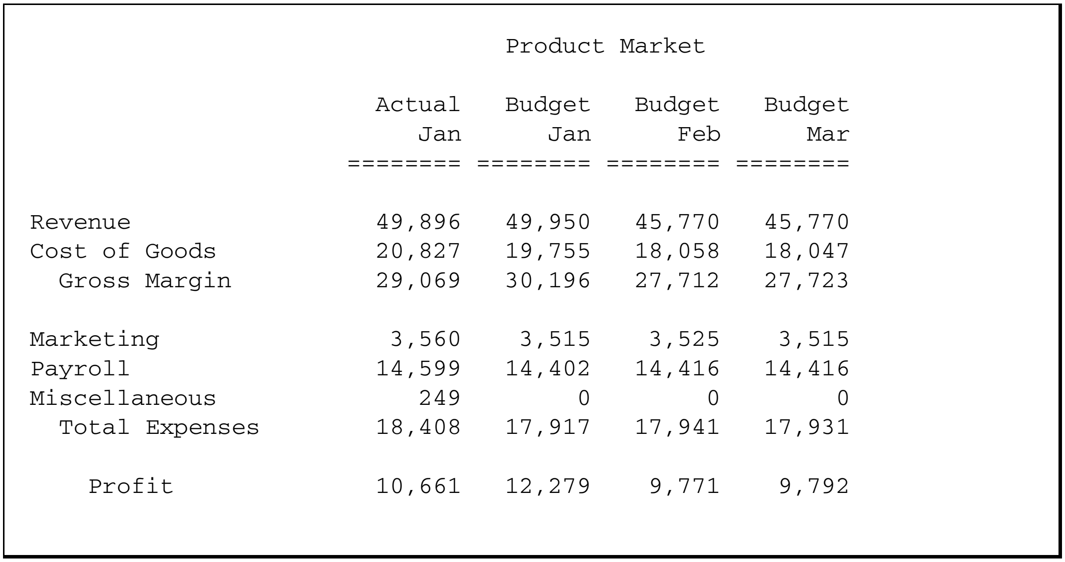
The SUPFEED command suppresses page feeds. A page feed automatically occurs when the report output reaches the default page length of 66 rows, unless you enter the PAGELENGTH command to change this setting. When a large flat file is created, you can use this command to prevent page breaks (blank rows) from being displayed in the report every time output reaches a logical page length.

This report script, `FLAT2SQL.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 13: Creating Asymmetric Columns

Asymmetric columns make up this report. Typically, a report contains symmetric columns. That is, when multiple dimensions are displayed across the page as column groups, each level of nested columns has the same number of members nested below. Because Actual has only one nested column, Jan, and Budget has three nested columns, this report is considered asymmetric.

Some rows in the report use names other than the member names from the database. In addition to allowing aliases, as in Sample 7, you can rename a row name in the reporter.



```
                                   Product Market

                          Actual   Budget   Budget   Budget
                             Jan      Jan      Feb      Mar
                        ======== ======== ======== ========

Revenue                   49,896   49,950   45,770   45,770
Cost of Goods             20,827   19,755   18,058   18,047
  Gross Margin            29,069   30,196   27,712   27,723

Marketing                  3,560    3,515    3,525    3,515
Payroll                   14,599   14,402   14,416   14,416
Miscellaneous                249        0        0        0
  Total Expenses          18,408   17,917   17,941   17,931

    Profit                10,661   12,279    9,771    9,792
```

Use the following script to create Sample 13:

```
<PAGE (Product, Market)

      <COLUMN (Scenario, Year)
      Actual   Budget Budget Budget
         Jan      Jan    Feb    Mar

<ROW (Accounts)

{ RENAME "Revenue" } Sales
{ RENAME "Cost of Goods" } Cost_of_Goods_Sold
{ RENAME "Gross Margin" } Margin

{ SKIP UNDERSCORECHAR " " }
<ICHILDREN Total_Expenses

{ SKIP }
Profit
!
```

To create an asymmetric report, you must specify the member name of each column. Because the report output has two column groupings, Scenario and Year, you must specify a member from each dimension for each column. If you do not specify each column member, the resulting report format is symmetric.

The RENAME command redefines a member name when the report is output. This is useful when you do not want to use an aliases table.

This report script, `ASYMM.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 14: Calculating Columns

This section contains two examples of CALCULATE COLUMN scripts and the reports they produce. CALCULATE COLUMN supports standard mathematical operations.

Sample 14-A: Basic Calculated Columns

```
                                         East

          Actual                                                  Budget          Var
   Jan     Feb     Mar    Qtr1                      Jan     Feb     Mar      Q1      Q1

====== ====== ====== ======                 ====== ====== ====== ====== =======
 1,295  1,132     553  2,980  Tele~ Profit     1,240     950     950  3,140   (160)
    25     27      14     66        Profit_%      26      22      22     70     (4)
    56     62      59    177        Margin_%      60      60      60    180     (3)
 1,417  1,120     898  3,435  VCR   Profit     1,466   1,161   1,161  3,788   (353)
    33     30      24     87        Profit_%      35      31      31     98    (10)
    61     61      62    183        Margin_%      63      63      63    189     (6)
   400    272     256    928  Cam~  Profit       528     360     360  1,247   (319)
    15     11      10     36        Profit_%      19      13      13     45    (10)
    70     70      70    211        Margin_%      71      71      71    213     (2)
 3,112  2,524   1,707  7,343  Visu~ Profit     3,234   2,471   2,471  8,175   (832)
    25     24      17     66        Profit_%      27      23      23     74     (7)
    61     63      63    187        Margin_%      64      64      64    191     (4)
```

Use the following script to create Sample 14-A:

```
<PAGE(Market)
East
      <COLUMN (Scenario, Year)
      Actual  Budget
      Jan Feb Mar
{ CALCULATE COLUMN "Qtr1"  = 2 : 4
  CALCULATE COLUMN "Q1" = 5 : 7
  CALCULATE COLUMN "Var~Q1" = 8 - 9

  ORDER 2,3,4,8,0,1,5,6,7,9
  WIDTH 7 WIDTH 10 0 1
}
<ROW (Product, Accounts)
<ICHILDREN Visual

<CHILDREN Accounts
    !
```

This report script, `COLCALC2.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 14-B: Asymmetric Columns

The following sample has two regular columns defined in *asymmetric* mode. For more information on asymmetric columns, see “Sample 13: Creating Asymmetric Columns” on page 1056.

```
                        East

  Budget                           Actual   Actual
     Jan                              Jan  % Sales
========                         ======== ========

   1,200 Television  Payroll        1,236        25%
     440             Marketing        365         9%
   1,240             Profit         1,295        26%
   4,800             Sales          5,244       100%

   1,030 VCR         Payroll        1,044        25%
     150             Marketing        156         4%
   1,466             Profit         1,417        35%
   4,200             Sales          4,311       100%

   1,195 Camera      Payroll        1,167        42%
     300             Marketing        288        11%
     528             Profit           400        19%
   2,850             Sales          2,656       100%

   3,425 Visual      Payroll        3,447        29%
     890             Marketing        809         8%
   3,234             Profit         3,112        27%
  11,850             Sales         12,211       100%
```

Use the following script to create Sample 14-B:

```
<PAGE(Market)
East

     <COLUMN(Scenario, Year)
     Budget  Actual
     Jan     Jan

{ ORDER 2,0,1,3,4 WIDTH 12 0 1 NOINDENTGEN AFTER "%" 4
  SKIPONDIMENSION Product LMARGIN 10 }

<ROW(Product, Accounts)

{ CALCULATE ROW "Sales" OFF }
{ CALCULATE COLUMN "Actual~% Sales" = 2 % "Sales" 2 }

<ICHILDREN Visual
{ SAVEROW } Sales
     Payroll
     Marketing
     Profit
<DUPLICATE Sales
    !
```

This report script, `COLCALC3.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 15: Calculating Rows

The sample reports in this section demonstrate CALCULATE ROW scripts and the reports they produce.

Sample 15-A: Basic Calculated Row

This sample report demonstrates the basic form of the CALCULATE ROW command.

```
                                Audio Actual Sales
                                 Jan      Feb      Mar
                            ======== ======== ========

Boston                         1,985    1,801    1,954
New_York                       2,310    2,082    2,259
Chicago                        2,043    1,884    1,814

Total Sales                    6,338    5,767    6,027
Avg Sales                      2,113    1,922    2,009
```

Use the following script to create Sample 15-A:

```
     Audio Actual Sales
     Jan Feb Mar

{ CALCULATE ROW "Total Sales" }    //create new calculated row
Boston
New_York
Chicago

{ SKIP
  CALCULATE ROW "Avg Sales" = "Total Sales" /3
  PRINTROW "Total Sales"
  PRINTROW "Avg Sales" }
    !
```

This report script, `ROWCALC1.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 15-B: Calculated Rows and Missing Relationships

This sample report is a simple summary of information in a North/South grouping, which is not part of the database outline. When relationships that you need for reporting are missing in the database outline, often the best solution is to use calculated rows (or columns).

```
           Budget Payroll

                      Jan      Feb      Mar
                     ====     ====     ====
Northern Cities
================
New_York            1,940    1,930    1,930
Boston              1,610    1,610    1,610
Chicago             1,630    1,630    1,630
San_Francisco       1,815    1,815    1,815
Seattle             1,415    1,409    1,409

Southern Cities
================
Denver              1,499    1,499    1,499
Los_Angeles         1,757    1,787    1,787
Dallas              1,002    1,002    1,002
Phoenix               900      900      900
Houston               834      834      834

Total Northern      8,410    8,394    8,394
Total Southern      5,992    6,022    6,022
```

Use the following script to create Sample 15-B:

```
// Declare Calculated Rows to Sum Southern and Northern Cities
{ CALCULATE ROW "Total Southern" OFF

// initially, set operation to OFF
  CALCULATE ROW "Total Northern" OFF  }

<PAGE(Product,Scenario,Accounts)
{ RENAME "" } Product             // all products, so blank out
                                  // the Product Label
Budget
Payroll
     <COLUMN(Year)
     Jan  Feb  Mar

<ROW(Market)                      // Northern Cities

{ SETROWOP "Total Northern" +     // Accumulate for Northern

SKIP 3
IMMHEADING// Put out heading now so text
                                  // will go after it
Text 0 "Northern Cities" UCHARACTERS
}

New_York Boston  Chicago San_Francisco Seattle

//Southern Cities

{ SETROWOP "Total Southern" +   } // Accumulate for Southern
{ SETROWOP "Total Northern" OFF } // Stop Accumulation for Northern

{ SKIP Text 0 "Southern Cities" UCHARACTERS }

Denver  Los_Angeles Dallas  Phoenix Houston

{ SKIP
PRINTROW "Total Northern"         // output calculated rows
PRINTROW "Total Southern"
}
    !
```

This report script, `ROWCALC2.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 15-C: Averaging Rows

This report sample restricts columns during calculation to average rows that contain partly numbers and percentages. The report must calculate the total regional average percentages using previously calculated rows that contain the total sales for the region. Also, the report must compute (for averaging) a count of regions. The number of regions is set as a constant in the database outline. If this number changes, the report definition must be modified. If a count of regions is not computed, a hard-to-notice error could result.

```
 Actual Total Sales for the 3 Video Products in Qtr1:  36,914   35,126   25,119
 Budget Total Sales for the 3 Video Products in Qtr1:  37,300   34,250   26,940
 ==================================================  ======   ======   ======
                  Qtr1

                      Television              VCR             Camera
                     Profit Profit_%       Profit Profit_%  Profit   Profit_%
                     ======  =======       ====== ========  ======  =========
 New_York   Budget    1,020   20.40%        1,382   31.41%     540     16.68%
            Actual      847   17.66%        1,243   29.62%     352     11.79%
 Boston     Budget    1,020   24.88%        1,344   35.37%     277     11.79%
            Actual    1,405   33.48%        1,002   27.49%     207      9.28%
 Chicago    Budget    1,100   25.58%        1,062   31.24%     430     16.54%
            Actual      728   16.51%        1,190   30.68%     369     14.72%
 San_Fran~  Budget      930   21.63%          718   21.12%   1,270      31.75
            Actual      674   15.54%        1,197   31.12%   1,000      27.4%
 Seattle    Budget      390   15.60%          973   32.98%     376     16.00%
            Actual      340   12.20%          977   31.56%     312     13.79%
 Denver     Budget      690   22.26%          929   30.97%     462     18.86%
            Actual      334   11.94%          914   30.48%     361     15.92%
Los_Ange~   Budget      810   18.41%        1,101   29.76%     506     18.40%
            Actual      429    9.11%        1,127   28.81%     377     14.62%
 Dallas     Budget      780   21.08%        1,341   36.24%     333     13.88%
            Actual      163    4.69%        1,055   30.28%     243     10.71%
 Houston    Budget      690   24.64%        1,128   36.39%     432     18.00%
            Actual      256   10.44%        1,064   34.98%     241     10.98%
 Phoenix    Budget      630   20.32%          894   31.93%     498     20.75%
            Actual      251    8.49%          940   31.07%     261     11.99%

                                    Total Regions Averages

 Avg     Budget   806    21.61%    1,087    31.74%      512     19.02%
 Avg     Actual   543    14.70%    1,071    30.49%      372     14.82%
```

Use the following script to create Sample 15-C:

```
{ // Declare some of the Calculated Rows to be used
  CALCULATE ROW "Avg~Budget" OFF
  CALCULATE ROW "Avg~Actual" OFF
  CALCULATE ROW "Tot Sales~Budget" OFF
  CALCULATE ROW "Tot Sales~Actual" OFF
}
// We need the values of Market->Visual->Qtr1->Sales->Actual and
// Market ->Visual->Qtr1->Sales ->Budget to compute some
// percentages at the bottom, so get them now

Market
<CHILDREN Visual Qtr1 Sales
{ SAVEROW "Actual Sales" } Actual  // stores into first 3
                           // data columns
{ SAVEROW "Budget Sales" } Budget  // of these rows, which
                           // are cols 1-3
                           // change to columns 2-4 when we
                           // specify 2 row dimensions in
                           // next section
// Since this is an example, not a formal report, we'll
// type out the values for Actual Sales and Budget Sales here so
// you can check the numbers:

{ SKIP 2
TEXT 0 "Actual Total Sales for the 3 Video Products in Qtr1:" 55 "*CALC"
"Actual Sales"
TEXT 0 "Budget Total Sales for the 3 Video Products in Qtr1:" 55 "*CALC"
"Budget Sales"
UCHARACTERS
SKIP 5 }
    !                      // Now we can do the main report
{ AFTER "%" 3,5,7  DECI 2 3,5,7 ZEROTEXT "--" MISSING "--"
  WIDTH 10 0 1  }

<PAGE(Year)
Qtr1
                <COLUMN(Product,Accounts)
                <CHILDREN Visual
                Profit     // split these 2 accounts onto
                           // 2 lines to prevent default
                Profit_%   // to asymmetric mode
                           // because both column
                           // dimensions have the same # of
                           // members selected. Could have
                           // used <SYM instead.
```

```
<ROW(Market,Scenario)
<ONSAMELEVELAS New_York
            { SETROWOP "Avg~Actual" OFF
              SETROWOP "Avg~Budget" +

              CALCULATE ROW "Count" = "Count" + 1.   }

            Budget

            { SETROWOP "Avg~Budget" OFF
              SETROWOP "Avg~Actual" +           }

           >{ SKIP }

            Actual

{ UCOLUMNS SKIP 2 }
{
  // at this point, Avg~Budget and Avg~Actual ARE NOT YET
  // AVERAGES--they are the SUM of the Profit rows of each type.
  // Before converting them to averages, the report computes
  // Profit as a % of total sales for each type. Since we only
  // have 1 value for "Budget Sales" and "Actual Sales",
  // for each of the three visual products in those
  // rows, the report restricts the reference to those rows to
  // columns 2-4 while computing the percentage columns 3, 5, and 7,
  // based on profits in columns 2, 4 and 6
  // calculate the percentages for Budget
CALCULATE ROW "Avg~Budget" 3 = "Avg~Budget" 2 % "Budget Sales" 2
CALCULATE ROW "Avg~Budget" 5 = "Avg~Budget" 4 % "Budget Sales" 3
CALCULATE ROW "Avg~Budget" 7 = "Avg~Budget" 6 % "Budget Sales" 4

  // now calculate the averages
CALCULATE ROW "Avg~Budget" 2  = "Avg~Budget" / "Count"
CALCULATE ROW "Avg~Budget" 4  = "Avg~Budget" / "Count"
CALCULATE ROW "Avg~Budget" 6  = "Avg~Budget" / "Count"

  // calculate the percentages for Actual
CALCULATE ROW "Avg~Actual" 3 = "Avg~Actual" 2 % "Actual Sales" 2
CALCULATE ROW "Avg~Actual" 5 = "Avg~Actual" 4 % "Actual Sales" 3
CALCULATE ROW "Avg~Actual" 7 = "Avg~Actual" 6 % "Actual Sales" 4

  // now calculate the averages
CALCULATE ROW "Avg~Actual" 2  = "Avg~Actual" / "Count"
CALCULATE ROW "Avg~Actual" 4  = "Avg~Actual" / "Count"
CALCULATE ROW "Avg~Actual" 6  = "Avg~Actual" / "Count"
TEXT C "Total Regions Averages"
PRINTROW "Avg~Budget"
PRINTROW "Avg~Actual" }
    !
```

This report script, `ROWAVG.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 16: Sorting by Top or Bottom Data Values

The following two reports demonstrate the use of TOP and BOTTOM conditional retrieval commands in a report script. For more information, see Chapter 35, “Developing Report Scripts.”

Sample 16-A: Bottom Data Values

This sample report demonstrates the basic use of the BOTTOM command. The report is based on the Sample Basic database.

```
                                              Measures

                                        Actual            Budget
                                     Jan       Dec       Jan       Dec
                                ======== ======== ======== ========
East            200                  158       233       280       340
                300                  184       277       240       210
                Diet                 181       213       200       240
West            100                  378       223       830       530
                300                  755       971       830       950
                400                  454       434       470       370
South           200                  480       496       520       390
                Diet                 355       404       490       430
                300                  188       213       270       240
Central         300                  790       824       930       810
                100                  724       792       900       890
                400                  691       785       660       650
  Market        200                2,141     2,302     2,710     2,810
                300                1,917     2,285     2,270     2,210
                400                1,611     1,720     1,730     1,600
```

Use the following script to create Sample 16-A:

```
<Sym
<Column (Scenario, Year)
Actual Budget
Jan Dec
<Row (Market, Product)
<ICHILDREN Market
<ICHILDREN Product
<Bottom (3, @DataColumn(3))
    !
```

The BOTTOM command specifies that only the three lowest data values are returned for each row grouping, based on the target data values specified in column three (Budget, Jan). Notice that no row dimension is selected here, so the report output defaults to the innermost row.

This report script, `BOTTOM.REP`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory.

Sample 16-B: Top Data Values

This sample report fragment demonstrates the basic use of the TOP command. The report is based on the Sample Basic database.

```
                                                Measures


                                        Actual            Budget
                                       Jan      Dec      Jan      Dec
                                  ======== ======== ======== ========

New York         100-10                262      271      260      250
                 200-40                175      312      200      320
                 400-10                101       89      120       90
                 400-20                 94      133      110      150
                 300-10                111      309      100      210
                 400-30                 54       52       70       60
                 300-20               (113)    (189)     (70)    (150)
                 200-10               (172)    (224)    (170)    (210)
Massachusetts    100-10                367      390      360      360
                 200-40                100       87      110       80
                 400-10                 29       29       40       40
                 400-30                 29       25       40       30
                 300-10                 17        7       30       10
                 200-10                (23)     (20)     (10)     (10)
...
  East           100-10                837      867      860      830
                 200-40                267      383      310      400
                 400-10                215      201      280      230
                 400-30                157      167      210      200
                 300-10                177      368      190      270
                 400-20                 94      133      110      150
                 200-20                 80       79      100      110
                 100-20                 67      122       70      110
                 100-30                 20       37       30       50
                 300-30                 34       12       30       20
```

Use the report script TOP.REP, reproduced here, to create Sample 16-B:

```
<Sym
//Supress shared members from displaying

<Supshare

     <Column (Scenario, Year)
     Actual Budget
     Jan Dec
<Row (Market, Product)
<Desc Market
//Use bottom level of products
<DimBottom Product
<Top (10, @DataColumn(3))
!
```

The TOP command specifies that only the ten highest data values are returned for each row grouping, based on the target data values specified in column three (Budget, Jan).

This report script, TOP.REP, is available in your \ESSBASE\APP\SAMPLE\BASIC directory.

Sample 17: Restricting Rows

The following report demonstrates the use of the RESTRICT conditional retrieval command in a report script. For more information, see Chapter 35, “Developing Report Scripts.”

```
                                          Measures

                                   Actual             Budget
                                Jan       Dec       Jan       Dec
                             ======== ======== ======== ========

East         200                 158       233       280       340
             300                 184       277       240       210
             Diet                181       213       200       240
South        300                 188       213       270       240
             400             #Missing #Missing #Missing #Missing
```

Use the following script to create Sample 17:

```
<Sym
<Column (Scenario, Year)
Actual Budget
Jan Dec
<Row (Market, Product)
<Ichildren Market
<Ichildren Product
<Restrict (@DataCol(3) < $300.00 )
    !
```

The RESTRICT command specifies that only data values that are less than $300.00 are returned for each row grouping, based on the target data values specified in column three (Budget, Jan). Notice that no row dimension is selected here, so the report output defaults to the innermost row.

This report script, `RESTRICT.REP`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory.

Sample 18: Ordering Data Values

The following report demonstrates the use of the ORDERBY conditional retrieval command in a report script. For more information, see Chapter 35, “Developing Report Scripts.”

```
                                           Sales Scenario
                                     Jan      Feb      Mar      Apr
                                ======== ======== ======== ========

New York        100-20          #Missing #Missing #Missing #Missing
                100-30          #Missing #Missing #Missing #Missing
                200-20          #Missing #Missing #Missing #Missing
                200-30          #Missing #Missing #Missing #Missing
                300-30          #Missing #Missing #Missing #Missing
                  Diet          #Missing #Missing #Missing #Missing
                200-10                61       61       63       66
                400-30               134      189      198      198
                300-20               180      180      182      189
                400-20               219      243      213      223
                400-10               234      232      234      245
                300-10               483      495      513      638
                200-40               490      580      523      564
                  200                551      641      586      630
                  400                587      664      645      666
                  300                663      675      695      827
                100-10               678      645      675      712
                  100                678      645      675      712
                    Product        2,479    2,625    2,601    2,835
```

Use the following script to create Sample 18:

```
<Page ("Measures")
<Column ("Scenario", "Year")
<Row ("Market", "Product")
"Sales"
"Scenario"
"Jan" "Feb" "Mar" "Apr"
"New York"
```

```
"Product" "100" "100-10" "100-20" "100-30" "200" "200-10"
"200-20" "200-30" "200-40" "300" "300-10" "300-20" "300-30" "400"
"400-10" "400-20" "400-30" "Diet" "100-20" "200-20" "300-30"

<ORDERBY ("Product", @DATACOL(1) ASC, @DATACOL(2) DESC,
@DATACOL(3) ASC @DataCol (4) DESC)
     !
```

The ORDERBY command is based only on data in the data columns. If the SUPPRESSMISSING command is not used in the report, #MISSING is considered to be the lowest data value. ORDERBY compares data values in the following order:

- Two values in the same column (for example, in COL1, the value associated with 200-10 is compared with the 400-30 data value, as shown in the example below).
- Data values between two data columns (for example, the data value in COL1 is compared with the data value in COL2, as shown in the example next).

If two data values are the same, the sort proceeds to the next column to determine the order.

In the following subset of Sample 18, for Product 200-10, the data values in COL1 and COL2 are both 61; the data in COL1 should be in ascending order, the data in COL2 should be in descending order. The two values are compared, and as they are the same, COL2 and COL3 are compared. Therefore, even though COL2 is supposed to be in descending order, the comparison for the row 400-30 was determined by the values in COL3, which is in ascending order.

```
                    COL 1    COL 2     COL 3    COL 4
                    =====    =====

200-10                 61       61        63       66
400-30                134      189       198      198
300-20                180      180       182      189
```

The report script for Sample 18, ORDERBY.REP, is available in your \ESSBASE\APP\SAMPLE\ BASIC directory.

Sample 19: Narrowing Member Selection Criteria

The following report demonstrates the use of the LINK command to narrow the members returned in a selection in a report script. For more information, see Chapter 35, “Developing Report Scripts.”

```
       Market Measures Scenario

                     Qtr1     Qtr2
                 ======== ========

100-10              5,096    5,892
100-20              1,359    1,534
100-30                593      446
200-10              1,697    1,734
200-20              2,963    3,079
200-30              1,153    1,231
200-40                908      986
300-10              2,544    3,231
300-20                690      815
300-30              2,695    2,723
400-10              2,838    2,998
400-20              2,283    2,522
400-30               (116)     (84)
100-20              1,359    1,534
200-20              2,963    3,079
300-30              2,695    2,723
    Product        24,703   27,107
```

Use the following script to create Sample 19:

```
<Page (Market)
<Column (Year)
Qtr1 Qtr2
<Row (Product)
<Link (<UDA (product, naturally-flavored) OR <LEV (product, 0))
    !
```

The LINK command uses the AND, OR, and NOT Boolean operators to refine your search. In the preceding example, the product with the “naturally-flavored” user-defined attribute (UDA), as well as all Level 0 products, are returned in the search.

Be careful how you group operators in the LINK expression. Hyperion Essbase evaluates operators from left to right. Use parentheses to group the expressions. For example, A OR B AND C is the same as ((A OR B) AND C). In the first expression, Hyperion Essbase evaluates the expression from left to right, evaluating A OR B before evaluating AND C. In the second expression, Hyperion Essbase evaluates the subexpression in parentheses (A OR B) before the whole expression, producing the same result. However, if you use (A OR (B AND C)), Hyperion Essbase evaluates the subexpression in parentheses (B AND C) before the whole expression, producing a different result.

This report script, `LINK.REP`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory.

Sample 20: Using Attributes in Member Selection

This sample report uses members of attribute dimensions to view data on base dimensions that are associated with those attribute dimensions.

```
Profit Actual Caffeinated_True Qtr1 East

          Ounces_32   Ounces_20   Ounces_16   Ounces_12    Ounces
        =========== =========== =========== =========== ===========

Bottle     #Missing         488         240        (586)        142
Can        #Missing    #Missing    #Missing       2,776       2,776
Pkg Type   #Missing         488         240       2,190       2,918
```

Use the following script to create Sample 20:

```
{WIDTH 12}
<Page (Measures, Scenario, Caffeinated, Year, Market)
Profit
Actual
Caffeinated_True
Qtr1
East
<Column (Ounces)
<ICHILDREN Ounces
<Row ("Pkg Type")
<ICHILDREN "Pkg Type"
    !
```

The report output reflects data on Quarter 1 profits for caffeinated products by all their available sizes and package types. The data values indicate #MISSING when there is no data for a specific size in a specific package type. Because attributes are defined only on sparse dimensions, there are several #MISSING values in the sample report. You can represent missing values by suppressing the row or substituting a replacement text string, such as N/A. See "Sample 2: Handling Missing Values" on page 1023 for an example of substituting page breaks and labels for missing values.

This report script, ATTR.REP, is available in your \ESSBASE\APP\SAMPLE\BASIC directory.

Sample 21: Using the WITHATTR Command in Member Selection

This sample report uses the WITHATTR command to view information based on the attributes of the members of a base dimension.

```
                         Profit Actual Qtr1 East

                        Bottle         Can    Pkg Type
                   =========== =========== ===========
100-30                      74    #Missing          74
200-30                #Missing    #Missing    #Missing
200-40                     908    #Missing         908
400-10                     645    #Missing         645
400-20                     290    #Missing         290
400-30                     545    #Missing         545
```

Use the following script to create Sample 21:

```
{WIDTH 12}
<Page (Measures, Scenario, Year, Market)
Profit
Actual
Qtr1
East
<Column ("Pkg Type")
<ICHILDREN "Pkg Type"
<Row (Product)
<WITHATTR(Caffeinated,"<>",True)
<IDESCENDANTS Product
!
```

The report output reflects data on Quarter 1 profits for caffeinated products by their package types. The data values indicate `#MISSING` when there is no data for a specific package type. Because attributes are defined only on sparse dimensions, there are several #MISSING values in the sample report.

This report script, `WITHATTR.REP`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory.

Chapter 37

Copying Data Subsets and Exporting Data to Other Programs

You can move data between OLAP Server databases or to another program by extracting an output file of the data you want to move. For example, you can copy a subset of an Essbase database from OLAP Server to Personal Essbase.

In order to meet the import format specifications of most other programs, use the Report Writer to create a text file.

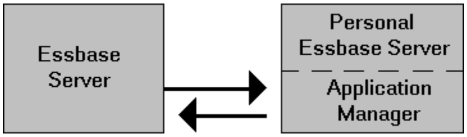
This chapter contains information about the following topics:

Copying a Database Subset to Personal Essbase

This version of Essbase includes the ability to install both the Essbase server and client on a Windows NT or Windows 2000 workstation. This is a one-port license and has its own license number. For information about installing and configuring Personal Essbase on a machine, see the *Essbase Installation Guide*.

Once you have installed Personal Essbase, you can copy the outline file (*dbname*.OTL) and a data subset from the Essbase server and load them into Personal Essbase. The Personal Essbase server does not communicate with the OLAP Server.

Figure 491: Essbase Server and Personal Essbase Interaction



Note: Do not create more than one application and two databases on the Personal Essbase server.

Summary of Steps

To copy a database subset to Personal Essbase, complete these steps. The following sections provide detailed information on completing each step.

1. On your Personal Essbase server, create a new application and database to contain the database subset.
2. Copy the outline file (for example, `source_dbname.OTL`) from your source database to the new database on Personal Essbase.

   You may need to rename the outline file to match the name of your Personal Essbase database (for example, `target_dbname.OTL`), overwriting the existing target database outline file.
3. Create an output file (for example, an ASCII text file) containing the required data subset.
4. Load the output file into the new database that you have created.

   See Chapter 20, “Introducing Data Loading” for more information about loading data.

If required, you can repeat steps 3 and 4 to create an output file from the database on your Personal Essbase server and load the data back into the main Essbase database on a different machine.

The example in the following sections is based on the Sample Basic database. The data subset in the example is the Actual, Measures data for the West market. The example copies the data subset to a Personal Essbase server.

Creating a New Application and Database

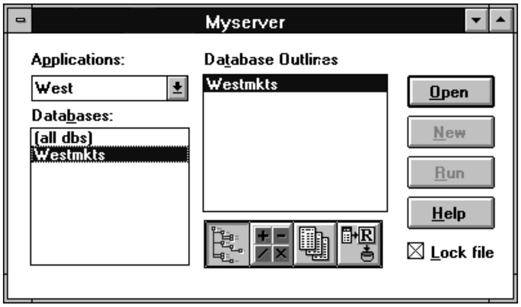
Create a new application and database on your Personal Essbase server. You will copy the required subset of data into this new database. You can give this application and database any name. In this example, you copy a subset of data for the West market, and name the application `WEST.APP`, and the database `WESTMKTS.DB`.

➤ To create the application and database:

1. In Application Manager, choose **File** > **New** > **Application** to open the Create New Application dialog box. Type the new application name in the Application name text box; for example, type **West**.
2. Click **OK**.
3. In Application Manager, choose **File** > **New** > **Database** to open the Create New Database dialog box. Type the new database name in the Database name text box; for example, type **Westmkts**.
4. Click **OK**.

   Essbase creates the new application and database.

Figure 492: New West Application and Westmkts Database



Applications: West

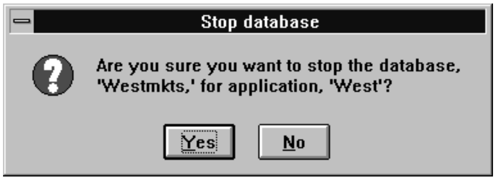
Databases: (all dbs), Westmkts

Database Outlines: Westmkts

☒ Lock file

Ensure that the new, empty database is not running. In Application Manager, select the new database and choose Application > Start/Stop from the main menu. A message box tells you if the database is running. If necessary, click Yes in the Stop database message box to stop the database.

Figure 493: Stop Database Dialog Box



Stop database

Are you sure you want to stop the database, 'Westmkts,' for application, 'West'?

Copying the Outline File from Your Source Database

Copy the outline file (`.OTL`) of your source database to the new database that you have created. In this example, you copy the `BASIC.OTL` outline file from the Sample Basic database and rename it to `WESMKTS.OTL` on Personal Essbase.

How you copy the outline file depends on whether you can connect to the source Essbase database from your Personal Essbase machine.

- If you *can* connect, use File > Save As in Application Manager to copy the outline. See "Copying the Outline File Using Application Manager" on page 1083.
- If you *cannot* connect (for example, if your Personal Essbase machine is a laptop machine that has no network connection), use your operating system to copy the outline file. For more information, see "Copying the Outline File Using the Operating System" on page 1084.

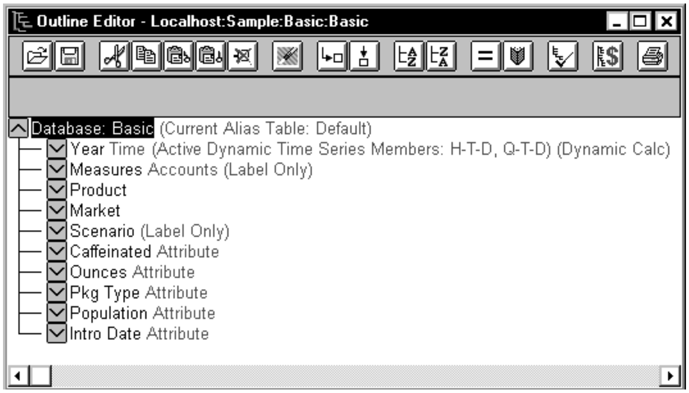
MaxL	You can use the create database statement to perform this task. For more information, see the *Technical Reference* in the `docs` directory.
ESSCMD	You can use the COPYDB command in ESSCMD to perform this task. See the *Technical Reference* in the `docs` directory for information about this command. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Copying the Outline File Using Application Manager

Complete these steps if you *can* connect to the source Essbase database from your Personal Essbase machine.

1. Connect to the source server.
2. In Application Manager, open the source outline; for example, open Sample Basic.

Figure 494: Sample Basic Outline

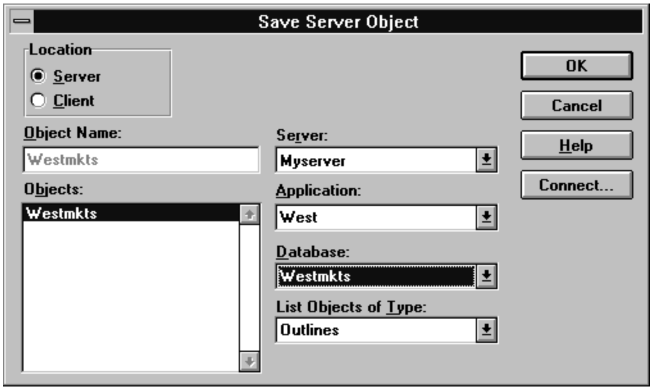


Outline Editor - Localhost:Sample:Basic:Basic

```
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)
  Caffeinated Attribute
  Ounces Attribute
  Pkg Type Attribute
  Population Attribute
  Intro Date Attribute
```

3. Choose **File** > **Save As** from the Outline Editor menu to display the Save Server Object dialog box.

Figure 495: Save Server Object Dialog Box



Save Server Object

Location
● Server
○ Client

Object Name: Westmkts

Objects: Westmkts

Server: Myserver

Application: West

Database: Westmkts

List Objects of Type: Outlines

4. In the Server, Application, and Database list boxes, select the new application and database that you have just created on the Personal Essbase server.
5. Click **OK** in the Save Server Object dialog box to copy the outline file.
6. At the prompt, click **Yes** to replace the existing `WESTMKTS.OTL` file.

   The existing outline file is an empty file, which Essbase created automatically when you created the new application and database.

   - Essbase might prompt you to unlock and overwrite the file. If necessary, click Yes at the prompts.
   - Essbase might warn you that the outline file did not originate from the latest version on the server. If necessary, click Yes to continue.

Copying the Outline File Using the Operating System

Complete these steps if you *cannot* connect to the source Essbase database from your Personal Essbase machine.

1. Use the operating system to copy the source outline file; for example, copy `basic.otl` to `westmkts.otl`.
2. Give the copied outline exactly the same name as the new database.
3. Save the outline file in the `\`*`ARBORPATH`*`\APP\`*`appname`*`\`*`dbname`* directory on your Personal Essbase machine, where *`ARBORPATH`* is the directory in which your installed Essbase, and appname and dbname are the new application and database that you have created.

   For example, copy `basic.otl` to a disk, renaming it to `westmkts.otl`. Then copy `westmkts.otl` from the disk to `c:\essbase\app\west\westmkts\westmkts.otl` on your Personal Essbase machine. It is safe to overwrite the existing, empty `westmkts.otl` file.

   Note: Ensure that the new outline file overwrites the existing, empty outline file, which Essbase created automatically when you created the new application and database.

   Before using the new outline file, you need to stop and restart the new database.

4. Select the new database in Application Manager. For example, select `Westmkts`.
5. Choose **Application** > **Start/Stop** from the main menu to stop the new database, and choose **Application** > **Start/Stop** again to restart the database.

You now have a copy of the database outline on your Personal Essbase server.

Creating an Output File Containing the Required Data Subset

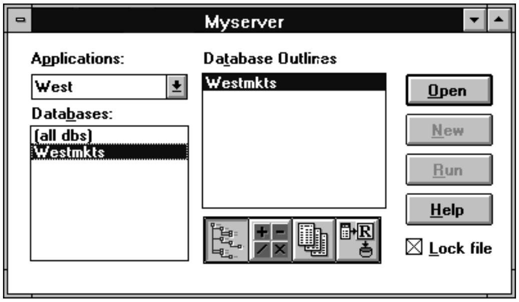
Create an output file that contains the required data subset. The output file can be an ASCII text file, or a spreadsheet file. You can use the Report Writer to create an ASCII text file of the data subset. For example, use <IDESCENDANTS to select a data subset.

Note: You can also use the Spreadsheet Client Retrieval Wizard to create a spreadsheet file of the data subset. For more information on using the Retrieval Wizard, see the *Essbase Spreadsheet Add-in User's Guides*.

➤ To create an ASCII text file that contains the required data subset:

1. In Application Manager, select the source database. For example, select **West Westmkts**.

Figure 496: Westmkts Database Selected in Application Manager

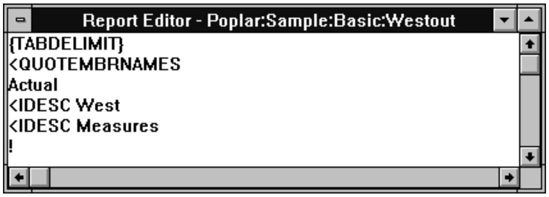


- If you *can* connect to your main Essbase database from your Personal Essbase machine, you can select the source database from your Personal Essbase machine.
- If you *cannot* connect, use a different machine from your Personal Essbase machine to select the source database.

2. Click the **Report** icon, , and then click **New** to open the Report Editor.
3. Write a report script that selects the required data subset. For information on writing report scripts, see Chapter 34, “Quick Start to Report Scripts” and the *Technical Reference* in the `docs` directory.

   For example, the following report script selects the Actual, Measures data for the West market from Sample Basic:

   Figure 497: Report Editor With Sample Basic Report Script

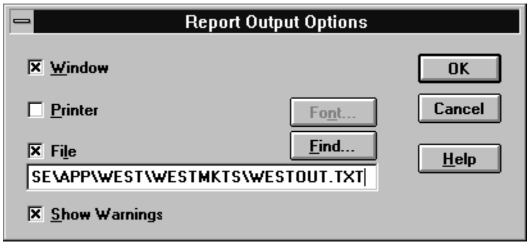
   

   Report Editor - Poplar:Sample:Basic:Westout

   ```
   {TABDELIMIT}
   <QUOTEMBRNAMES
   Actual
   <IDESC West
   <IDESC Measures
   !
   ```

   - Use TABDELIMIT to place tab stops between data, instead of spaces. This ensures that no member names or data values are truncated.
   - Use QUOTEMBRNAMES to place quotation marks (" ") around member names that contain blank spaces. Essbase then recognizes the member names when it loads the data.
4. In the Report Editor, choose **Report** > **Output Options** to open the **Report Output Options** dialog box.
5. Check the **File** and **Window** check boxes.

   Figure 498: Report Output Options Dialog Box

   ☒ Window

   □ Printer

   ☒ File

   SE\APP\WEST\WESTMKTS\WESTOUT.TXT

   ☒ Show Warnings

6. Give the report output file any name with a `.TXT` extension; for example, **WESTOUT.TXT**.

   To load the data, the output file needs to be in the `\`*`ARBORPATH`*`\APP\`*`appname`*`\`*`dbname`* directory on your Personal Essbase server, where *`ARBORPATH`* is the directory in which you installed Essbase, and *appname* and *dbname* are the new application and database directories that you have created.

   If you are using your Personal Essbase machine, you can save the output file directly into the `\`*`ARBORPATH`*`\app\`*`appname`*`\`*`dbname`* directory. For example, type `c:\essbase\app\west\westmkts\westout.txt` in the File text box.

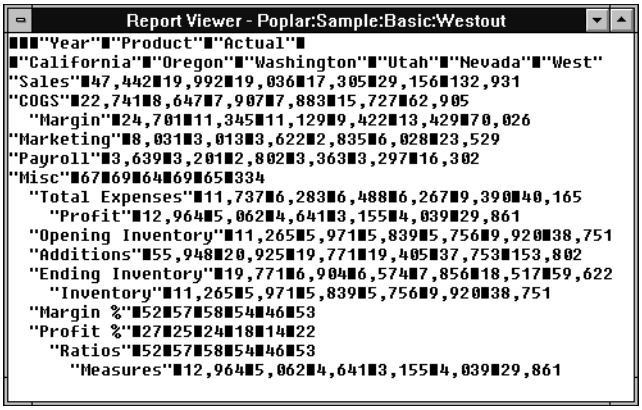
   If you are *not* using your Personal Essbase machine, save the output file anywhere on the current machine. By default, Essbase saves the file on your Essbase client machine, and not on the server. When you run the report, use your operating system to copy the file to the `\`*`ARBORPATH`*`\APP\`*`appname`*`\`*`dbname`* directory on your Personal Essbase server. For example, use a disk to copy the file.

7. Click **OK** in the **Report Output Options** dialog box.

8. Save your report script and choose **Report > Run** from the **Report Editor** menu to run the report.

   Essbase displays the report in a window and sends it to the file you specified in the **Report Output Options** dialog box.

Figure 499: Sample Basic Report Script Output



```
■■■"Year"■"Product"■"Actual"■
■"California"■"Oregon"■"Washington"■"Utah"■"Nevada"■"West"
"Sales"■47,442■19,992■19,036■17,305■29,156■132,931
"COGS"■22,741■8,647■7,907■7,883■15,727■62,905
  "Margin"■24,701■11,345■11,129■9,422■13,429■70,026
"Marketing"■8,031■3,013■3,622■2,835■6,028■23,529
"Payroll"■3,639■3,201■2,802■3,363■3,297■16,302
"Misc"■67■69■64■69■65■334
  "Total Expenses"■11,737■6,283■6,488■6,267■9,390■40,165
    "Profit"■12,964■5,062■4,641■3,155■4,039■29,861
  "Opening Inventory"■11,265■5,971■5,839■5,756■9,920■38,751
  "Additions"■55,948■20,925■19,771■19,405■37,753■153,802
  "Ending Inventory"■19,771■6,904■6,574■7,856■18,517■59,622
    "Inventory"■11,265■5,971■5,839■5,756■9,920■38,751
  "Margin %"■52■57■58■54■46■53
  "Profit %"■27■25■24■18■14■22
    "Ratios"■52■57■58■54■46■53
      "Measures"■12,964■5,062■4,641■3,155■4,039■29,861
```

If you are *not* using your Personal Essbase machine, remember to download and copy the file from the Essbase client directory to the `\ARBORPATH\app\appname\dbname` on your Personal Essbase server. For example, copy the output file to `c:\essbase\app\west\westmkts\westout.txt`.

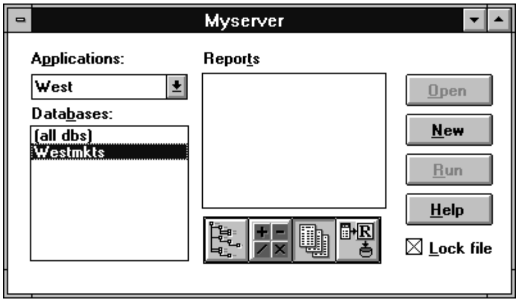
You are now ready to load the text file into your new database.

Loading the Output File Into the New Database

Load the output file into the new database on your Personal Essbase machine.

1. In Application Manager on your Personal Essbase server, select the new database. For example, select `Westmkts`.

Figure 500: Westmkts Database Selected in Application Manager



Applications: West

Databases: (all dbs), Westmkts (selected)

☒ Lock file

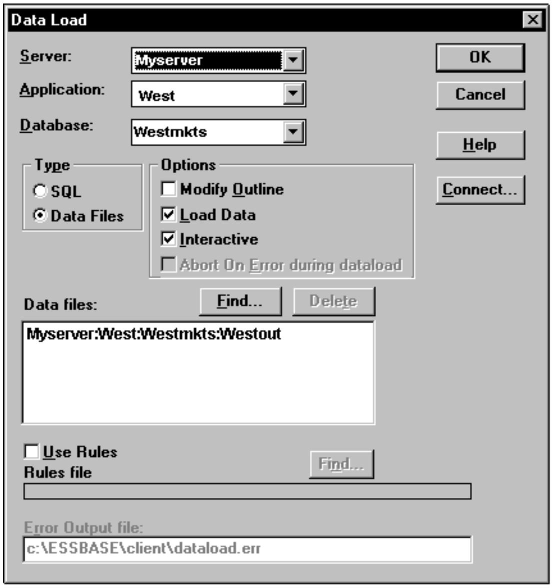
2. Choose **Database** > **Load Data** from the Application Manager main menu.

   The **Data Load** dialog is box displayed.

3. Click **Find** to open the **Open Server Data File Objects** dialog box.
   - In the **Objects** list box, select the ASCII text file you have just created; for example, select `WESTOUT`.
   - Click **OK**.

Note: If WESTOUT is not displayed, check that you gave it a `.TXT` extension, and placed it in the `\`*`ARBORPATH`*`\APP\WEST\WESTMKTS` directory. See "Creating an Output File Containing the Required Data Subset" on page 1085.

Figure 501: Data Load Dialog Box with WESTOUT Selected



Server: Myserver

Application: West

Database: Westmkts

Type

○ SQL

● Data Files

Options

☐ Modify Outline

☑ Load Data

☑ Interactive

☐ Abort On Error during dataload

Data files:

Myserver:West:Westmkts:Westout

☐ Use Rules

Rules file

Error Output file:

c:\ESSBASE\client\dataload.err

4. Click **OK** in the **Data Load** dialog box to load the text file into the new database.

For detailed information on loading data and any errors that may occur, see Chapter 20, "Introducing Data Loading."

You can now view the data on your Personal Essbase machine. You might need to recalculate the database subset. Because you are viewing a subset of the database, a percentage of the data values will be `#MISSING`.

If required, you can copy report scripts and other object files to your Personal Essbase machine to use with the database subset you have created.

Using Report Scripts for Data Exporting

You can use report scripts to export Essbase data to other programs in text format. Report Writer enables you to create text files that meet the import format specifications of most other programs.

Before you can import data into some programs, you must separate, or delimit, the data with specific characters.

If you plan to import Essbase data into a program that requires special delimiters, use the MASK command. For the syntax and usage of Report Writer commands, see the *Technical Reference* in the `docs` directory.

Note: You cannot export data generated by Dynamic Calc members. Because attributes are Dynamic Calc members, you cannot export data generated by attributes.

When you export data to a program that uses a two-dimensional, fixed-field format, you do not need to specify page or column dimensions. To create a two-dimensional report, you can specify every dimension as a row dimension. Use the ROWREPEAT command to add the name of each member specified to each row (rather than the default, nested style). The following script example and report illustrate this situation for a five-dimensional database:

```
<ROW (Year, Measures, Product, Market, Scenario)
{ROWREPEAT}
<ICHILDREN Year
Sales
<ICHILDREN "400"
East
Budget
    !
```

This example produces the following report:

```
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
```

If you want to create a two-dimensional report that contains only bottom-level (level 0) data, use CHILDREN or DIMBOTTOM to select level 0 members.

- To list only level 0 data for specific members, use the CHILDREN command with the level 1 member as a parameter above the data you want to print.
- To list only level 0 data for a given member (including a dimension), use the DIMBOTTOM command as a parameter with the dimension name that contains the data you want to print.

For example, the following script uses the CHILDREN command to select the children of Qtr1, which is a level 1 member, and the DIMBOTTOM command to select all level 0 data in the Product dimension.

```
<ROW (Year, Measures, Product, Market, Scenario)
{ROWREPEAT}
{DECIMAL 2}
<CHILDREN Qtr1
Sales
<DIMBOTTOM Product
East
Budget
     !
```

This example produces the following report:

```
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
```

See Chapter 36, “Examples of Report Scripts,” for another example of formatting for data export.

Importing Data Into Other Databases

Before you import data into some programs, you must delimit the data with specific characters. If you plan to import Essbase data into a program that requires special delimiters, use the MASK command. For the syntax and usage of Report Writer commands, see the *Technical Reference* in the `docs` directory.

Exporting Data with MaxL or ESSCMD

You can use MaxL or ESSCMD to export data at the command line or from a script. For instructions, see “Using Export to Back Up Data” on page 1251, or see the entries for the MaxL **export data** statement or the ESSCMD EXPORT command in the *Technical Reference* in the `docs` directory.

Exporting Data using Application Manager

You can use Application Manager to export data.

➤ Use this procedure to export data with Application Manager:

1. Start Essbase.
2. Start Application Manager and connect to the server containing the application and database from which you want to export data.
3. Select the application and database whose data you want to export.
4. Select the file menu Database > Export command and follow the instructions in the dialog box which appears.

For more information, see the Application Manager online help.

IBM® DB2® OLAP Server™ 8.1

Database Administrator's Guide

Volume III: Maintaining, Automating, and Optimizing Essbase

IBM®

SC18-7003-00

Printed in the U.S.A.

Part

VIII

Storing and Protecting Data

This part describes how to manage data storage for Essbase databases and how to ensure data integrity:

- Chapter 38, “Managing Essbase Kernel Settings,” introduces you to the Hyperion Essbase Kernel and describes how the kernel stores and accesses data, and how you can adjust the database settings to control that access and storage.
- Chapter 39, “Allocating Storage and Compressing Data,” describes how Hyperion Essbase stores data on disk, explains how to use disk volumes, and describes the compression methods Hyperion Essbase uses to compress data.
- Chapter 40, “Ensuring Data Integrity,” explains how Essbase handles transactions and locking and discusses other ways that Hyperion Essbase protects data. This chapter describes transactions, isolation levels, and data recovery, and tells you how to configure Hyperion Essbase to ensure the integrity of your data.

Chapter 38

Managing Essbase Kernel Settings

This chapter introduces you to the Essbase Kernel, explains how Essbase accesses and stores data, and provides an overview of Essbase Kernel settings you can use to optimize performance at the Essbase server level.

This chapter contains the following sections:

Note: For information about fatal errors in the Essbase Kernel, see "Understanding Fatal Error Handling" on page 1460.

Understanding the Essbase Kernel

The Essbase Kernel provides the foundation for a variety of functions of the Essbase server. These functions include data loading, calculations, spreadsheet lock&send, partitioning, and restructuring. The Essbase Kernel reads, caches, and writes data; manages transactions; and enforces transaction semantics to ensure data consistency and data integrity.

The Essbase Kernel has the following functions:

- Handles disk storage and caching of Essbase files
- Handles data retrieval

- Handles data updates
- Controls input-output functions related to Essbase
- Consolidates free space for re-use
- Manages concurrent operations
- Recovers databases after a server crash
- Issues locks
- Manages transactions

The rest of this topic explains the two available access modes, and presents an overview of several Essbase Kernel components:

- "Understanding Buffered I/O and Direct I/O" on page 1100
- "Setting the I/O Access Mode" on page 1101
- "Essbase Kernel Components" on page 1101

Understanding Buffered I/O and Direct I/O

The Essbase Kernel uses buffered I/O (input/output) by default, but direct I/O is available on most of the operating systems and file systems that Essbase supports. For a list of the supported platforms, see the *Essbase Installation Guide*.

Buffered I/O uses the file system's buffer cache.

Direct I/O bypasses the file system's buffer cache, and is able to perform asynchronous, overlapped I/Os. The following benefits are provided:

- Faster response time. A user waits less time for Essbase to return data.
- Scalability and predictability. Essbase lets you customize the optimal cache sizes for its databases.

If you set a database to use direct I/O, Essbase attempts to use direct I/O the next time the database is started. If direct I/O is not available on your platform at the time the database is started, Essbase uses buffered I/O, which is the default. However, Essbase will store the I/O access mode selection in the security file, and will attempt to use that I/O access mode each time the database is started.

Note: Cache memory locking can only be used if direct I/O is used. You also must use direct I/O if you want to use an operating system's no-wait (asynchronous) I/O.

Setting the I/O Access Mode

Buffered I/O is the default for all databases. You can view which I/O access mode a database is currently using or is currently set to, by selecting Database > Information in Application Manager, **display database** in MaxL, or GETDBINFO in ESSCMD.

You must perform these tasks to use direct I/O for Essbase:

- To use direct I/O instead of the default buffered I/O for any database, change the I/O access mode in the database settings by selecting Database > Settings (Storage tab) in Application Manager, **alter database** in MaxL, or SETDBSTATEITEM in ESSCMD. You may also need to increase the size of some caches. See "Sizing Caches" on page 1319 for instructions.
- To use direct I/O instead of the default buffered I/O for all databases migrated from an earlier release and for newly created databases, you can add the configuration setting DIRECTIO to the server configuration file `essbase.cfg`, and set the value to TRUE. See the *Technical Reference* in the `docs` directory for instructions. Be sure to consult the *Essbase Installation Guide*, in PDF format, in the `/essbase/docs/pdf` directory for information and instructions to help you make the transition. Also consult the cache sizing information in "Sizing Caches" on page 1319.

Essbase Kernel Components

The Essbase Kernel contains components that control all aspects of retrieving and storing data:

- The Index Manager finds and tracks the location of requested data. See "Index Manager" on page 1102 for details.
- The Allocation Manager is part of the Index Manager, allocates space and manages some file operations. See "Allocation Manager" on page 1103 for details.
- The Data Block Manager retrieves the data pointed to by the index and stores the data. See "Data Block Manager" on page 1104 for details.
- The LRO Manager handles retrieval and storage of linked reporting objects (LROs). See "LRO Manager" on page 1104 for details.

- The Lock Manager handles the locking of data blocks to regulate concurrent data access. See “Lock Manager” on page 1105 for details.
- The Transaction Manager tracks transactions and handles internal commit and abort operations. “Transaction Manager” on page 1105 for details.

Index Manager

The Index Manager manages the database index and provides a fast way of looking up Essbase data blocks. The Index Manager determines which portions of the database index to cache in the index cache, and manages the index cache.

The Index Manager controls five components. The following table describes these components:

Component	**Description**
Index	The method that Essbase uses to locate and retrieve data. The term index also refers to the index file.
Index File	File that Essbase uses to store data retrieval information. It resides on disk and contains index pages. Essbase names index files incrementally on each disk volume, using the naming convention `ESS`*`xxxxx`*`.IND`, where *xxxxx* is a number. The first index file on each disk volume is named `ESS00001.IND.`
Index page	A subdivision of an index file that contains index entries that point to data blocks.
Index entry	A pointer to a data block. An index entry exists for every intersection of sparse dimensions.
Index cache	A buffer in memory that holds index pages.

The term *index* refers to all index files for a single database. The index can span multiple volumes, and more than one index file can reside on a single volume. Use the disk volumes setting to specify disk spanning parameters. For information on setting the index cache size, see “Sizing Caches” on page 1319. For information about allocating storage space with the disk volumes setting, see “Specifying Disk Volumes” on page 1116.

Allocation Manager

Allocation Manager, part of the Index Manager, performs these tasks:

- Creation and extension of index and data files on disk
- File open and close operations
- Designation of which volume to use for a new file
- Sequence of volume use

When one of these tasks needs to be performed, the Allocation Manager uses this process to allocate space:

1. It attempts to use free space in an existing file.
2. If enough free space is not available, it attempts to expand an existing file.
3. If enough free space is not available in existing files, it creates a new file on the current volume.
4. If it cannot expand an existing file or create a new file on the specified volume, it attempts to use the next specified volume.
5. If all specified volumes are full, an error message is displayed, and the transaction is aborted.

The Allocation Manager allocates space for index and data files based on the database settings for storage. You can check current values and set new values in any of these ways:

- Database > Settings > Storage in the Application Manager
- `SETDBSTATEITEM 23 in ESSCMD`
- **alter database** in MaxL

See “Specifying Disk Volumes” on page 1116 for information about the disk volumes setting.

For more information, see “Storage Allocation” on page 1115.

Data Block Manager

The Data Block Manager brings data blocks into memory, writes them out to data files, handles data compression, and writes data files to disk. The Data Block Manager controls four components. The following table describes each component:

Component	Description
Data file	A file that contains data blocks. Essbase generates the data file upon data load and stores it on disk. Essbase names data files incrementally: `ESS`*`xxxxx`*`.PAG`, where *xxxxx* is a number starting with 00001.
Data block	The primary storage unit within Essbase. A data block is a multidimensional array that represents cells of the dense dimensions for a given intersection of sparse dimensions.
Data cache	A buffer in memory that holds uncompressed data blocks.
Data file cache	A buffer in memory that holds compressed data files (`.PAG`).

The size of the data file cache determines how much of the data within the data files can fit into memory at one time. The data cache size and the data block size determine how many data blocks can fit into memory at one time. Data files for a single database can span multiple volumes; more than one database can reside on the same volume. For information on setting the data file cache size and data cache size, see "Sizing Caches" on page 1319. For information about allocating storage space with the disk volumes setting, see "Specifying Disk Volumes" on page 1116.

LRO Manager

Linked reporting objects (LROs) enable you to associate objects, such as flat files, with data cells. Using the Spreadsheet Add-in, users can create and store LRO files, with an `.lro` extension.

LRO files are stored in the database directory (`\essbase\`*`appname`*`\`*`dbname`*, for example, `\essbase\sample\basic`).

Essbase stores information about linked reporting objects in an LRO catalog. Each catalog resides in its own Essbase index page and coexists in an index file with other, non-LRO Essbase index pages.

For more information about linked reporting objects, see Chapter 11, "Linking Objects to Essbase Data," and the *Essbase Spreadsheet Add-in User's Guide*.

Lock Manager

The Lock Manager issues locks on data blocks, which in turn controls concurrent access to data.

The *committed access* and *uncommitted access* isolation levels use different locking schemes. For more information on isolation levels and locking, see Chapter 40, "Ensuring Data Integrity."

Transaction Manager

The Transaction Manager controls transactions and commit operations and manages database recovery.

Essbase commits data automatically. Commits are triggered by transactions that modify data: data loading, calculating, restructuring, and spreadsheet lock&send operations.

How Essbase commits data depends upon whether the transaction isolation level is set to committed or uncommitted access (the default). For information about specifying isolation level, see "Isolation Levels" on page 1132.

The Transaction Manager maintains a transaction control table, `database_name.TCT`, to track transactions.

For more information about commit operations and recovery, see Chapter 40, "Ensuring Data Integrity."

Understanding the Essbase Kernel Startup

This list is the sequence of events during an Essbase Kernel start-up:

1. After the OLAP Server starts, a user connects to it from a client.
2. The user starts a database.
3. Essbase loads the database.

4. The Essbase Agent passes database settings to the server.
5. The Essbase Kernel begins its initialization process.
6. The Essbase Kernel starts its components: the Index Manager, Lock Manager, LRO Manager, Data Block Manager, and Transaction Manager.

If it encounters an error during start up, the Essbase Kernel shuts itself down.

Precedence of Settings

Hyperion Essbase provides default values for database storage settings in the configuration file `essbase.cfg`. You can leave the default settings, or change their values in two places:

- You can define storage settings for all databases on the Hyperion Essbase server by changing values in the configuration file `essbase.cfg`.
- You can define storage settings for a single database using any of these techniques:
  - In the Database Settings dialog box using Application Manager. See "Using Application Manager for Kernel Settings" on page 1110 for details.
  - In the Database Properties window using Administration Services. See the *Essbase Administration Services Online Help* for more information.
  - Using **alter database** in MaxL. See "Using alter database in MaxL for Kernel Settings" on page 1113, or the *MaxL User's Guide* for more information.
  - Using the SETDBSTATEITEM and SETDBSTATE commands in ESSCMD to define storage settings for a single database. See "Using SETDBSTATEITEM in ESSCMD for Kernel Settings" on page 1113, or the *Technical Reference* in the `DOCS` directory for more information.

Changes made for an individual database permanently override `essbase.cfg` settings and Essbase defaults for the relevant database until they are changed or withdrawn (through ESSCMD, MaxL, or the Database Settings dialog box).

If you change settings at the database level, the changes become effective at different times, as shown in Table 39:

Table 39: Database-level storage settings effectiveness

Setting	When setting becomes effective
• Index cache • Data file cache • Data cache • Cache memory locking • Disk volume	After you stop and restart a database.
Isolation level parameters Concurrency parameters	The first time after setting these values that there are no active transactions.
All other settings	Immediately.

Note: The size of index pages is fixed at 8 K. This is to reduce input-output overhead, as well as to simplify database migration.

Understanding How Essbase Reads Settings

Essbase reads the essbase.cfg file when you start the server, and then applies settings to the appropriate databases that you have created using any of these methods:

- Using the Database Settings dialog box using Application Manager
- Using the Database Properties window in Administration Services
- Using the SETDBSTATE and SETDBSTATEITEM commands from ESSCMD
- Using **alter database** from MaxL, for the appropriate databases

Database settings that are specified in the Database Settings dialog box of Application Manager, the Database Properties window of Administration Services, or in ESSCMD/MaxL always override `essbase.cfg` file settings, even if you change a setting in the `essbase.cfg` file after you have applied a setting for a particular database. Only removing a setting (via the Database dialog box, the Database Properties window, ESSCMD or MaxL) triggers Hyperion Essbase to use the `essbase.cfg` file, and then only after restarting the server.

Viewing Most Recently Entered Settings

To view the most recently entered settings, use any of these methods:

- Use display database in MaxL
- Use the GETDBSTATE command in ESSCMD
- Open the Database Properties window in Administration Services
- Select Database > Settings in Application Manager
- To view the settings that are currently in effect, use the GETDBINFO command in ESSCMD, open the Database Properties window in Administration Services, or select Database > Information in Essbase Application Manager

For information on stopping and starting servers, applications, and databases, see Chapter 41, "Running Essbase Servers, Applications, and Databases."

Customizing Essbase Kernel Settings

You can customize different Essbase Kernel settings for each database on the server. The information in this section helps you understand what each setting controls and how to adjust each setting. For a table of performance-related settings, see Chapter 47, "Improving Essbase Performance."

Note: Configure settings that are applied to an entire Essbase server with the `essbase.cfg` configuration file (`essbase.cfg` on UNIX platforms). For information about how to create this file and about what settings are available, see the *Technical Reference* in the `docs` directory.

You can customize these major kernel settings:

Table 40: Major Kernel Settings

Setting	Where to Find More Information
Index cache size	"Sizing the Index Cache" on page 1320
Data file cache size	"Sizing the Data File Cache" on page 1322
Data cache size	"Sizing the Data Cache" on page 1324
Cache memory locking	"Before You Size: Using Cache Memory Locking" on page 1317
Disk volumes	"Storage Allocation" on page 1115
Data compression	"Data Compression" on page 1124
Isolation level	"Isolation Levels" on page 1132

This section provides an overview of how to specify these settings:

- "Specifying and Changing Kernel Settings" on page 1109
- "Using Application Manager for Kernel Settings" on page 1110
- "Using alter database in MaxL for Kernel Settings" on page 1113
- "Using SETDBSTATEITEM in ESSCMD for Kernel Settings" on page 1113

For a full list of database settings, see "Using Application Manager for Kernel Settings" on page 1110.

Specifying and Changing Kernel Settings

Before you change any kernel settings, be sure to review information about precedence of the settings as changed in different parts of Essbase, and how those settings are read by Essbase:

- "Understanding the Essbase Kernel Startup" on page 1105
- "Understanding How Essbase Reads Settings" on page 1107

You can specify most Essbase Kernel settings in four ways:

- Using the Database Settings dialog box in Essbase Application Manager. See "Using Application Manager for Kernel Settings" on page 1110 for details.
- Using the Database Properties window in Administration Services (see *Essbase Administration Services Online Help* for more information).
- Using ESSCMD's SETDBSTATE and SETDBSTATEITEM. See "Using SETDBSTATEITEM in ESSCMD for Kernel Settings" on page 1113 for details.
- Using MaxL's **alter database** command.

These different methods provide different ways to change the same database settings. In rare cases, you may want to use the `essbase.cfg` server file to specify settings.

CAUTION: In previous versions of Essbase, you could specify many Essbase database settings in the `essbase.cfg` server file. In Version 5.*x* and higher, Essbase overrides most of the `.cfg` settings.
See "Understanding the Essbase Kernel Startup" on page 1105.

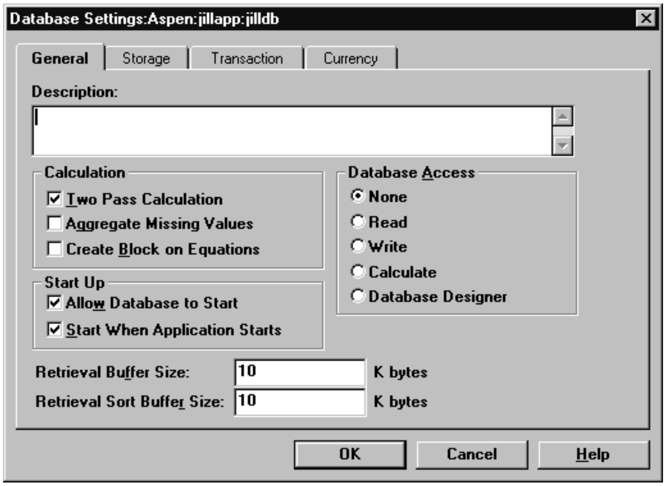
Using Application Manager for Kernel Settings

You can specify database, application, and server settings by using the settings dialog boxes in Essbase Application Manager. This section covers database settings. See Chapter 15, "Managing Security for Users and Applications" for information on application and server settings.

For information about database settings to improve performance, see Chapter 47, "Improving Essbase Performance."

➤ To customize database settings using Application Manager, select Database > Settings. This dialog box is displayed:

Figure 502: The Database Settings Dialog Box



Description:

Calculation

☑ Two Pass Calculation

□ Aggregate Missing Values

□ Create Block on Equations

Start Up

☑ Allow Database to Start

☑ Start When Application Starts

Database Access

● None

○ Read

○ Write

○ Calculate

○ Database Designer

Retrieval Buffer Size: 10 K bytes

Retrieval Sort Buffer Size: 10 K bytes

The dialog box has four tabs, which correspond to the following pages.

The General page includes the following settings:

- Calculations—see Part VI, “Calculating Data” for more information.
- Database access—see Chapter 15, “Managing Security for Users and Applications” for more information.
- Database startup—see the Essbase Application Manager online help for more information.
- Retrieval buffers—see Chapter 53, “Optimizing Reports and Other Types of Retrieval” for more information.

The Storage page includes the following settings:

- Data compression—see “Data Compression” on page 1124 for more information.
- Cache memory locking—see “Before You Size: Using Cache Memory Locking” on page 1317 for more information.
- Index cache size—see “Sizing the Index Cache” on page 1320 for more information.
- Data file cache size—see “Sizing the Data File Cache” on page 1322 for more information.
- Data cache size—see “Sizing the Data Cache” on page 1324 for more information.
- Location of disk volumes—see “Specifying Disk Volumes” on page 1116 for more information.

The Transaction page includes the following settings:

- Isolation level—see “Isolation Levels” on page 1132 for more information.
- Synchronization point—see “Uncommitted Access” on page 1137 for more information.
- Concurrency parameters—see “Committed Access” on page 1134 for more information.

The Currency page includes settings for currency databases. See Chapter 12, “Designing and Building Currency Conversion Applications,” for more information.

Note: Changes that you make in any pages of the Database Settings dialog box affect only the specified database. For information about when settings take effect, see Table 39 on page 1107.

Database settings that you set or change with either of these methods affect only the specified database.

Using *alter database* in MaxL for Kernel Settings

Issue a separate **alter database** statement for each database setting you want to change. For example, the following MaxL script logs in to Essbase, changes three database settings, and logs out:

```
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;
```

Note: Terminate each MaxL statement with a semicolon when issuing them using the MaxL Command Shell; however, if MaxL statements are embedded in Perl scripts, do *not* use the semicolon statement terminator.

You can use MaxL to write batch scripts that automate database setting changes. For information, see the *MaxL Language Reference*, located in the *Technical Reference* in the `docs` directory.

Using SETDBSTATEITEM in ESSCMD for Kernel Settings

For simple items, specify the command, item number representing the parameter, application, database, and value for the parameter:

```
SETDBSTATEITEM 2 "JILLAPP" "JILLDB" "Y";
```

For parameters that require multiple values, such as Isolation Level (item 18), specify multiple values, in this case, all the values after “BASIC”:

```
SETDBSTATEITEM 18 "SAMPLE" "BASIC" "1" "Y" "-1";
```

If you don’t know the parameter number, omit it, and Essbase lists all parameters and their corresponding numbers. Essbase also prompts you for a database and an application name.

Use a separate SETDBSTATEITEM command for each parameter; you cannot string parameter numbers together on the same line.

See the *Technical Reference* in the `docs` directory for information about SETDBSTATE and SETDBSTATEITEM commands. See Chapter 45, “Automating the Production Environment,” for information about ESSCMD.

Note: SETDBSTATEITEM or SETDBSTATE affect only the specified database.

You can include SETDBSTATEITEM (or SETDBSTATE) in batch scripts. For information about writing batch scripts, see Chapter 45, “Automating the Production Environment.” See the *Technical Reference* in the `docs` directory for information on ESSCMD syntax.

Chapter

39

Allocating Storage and Compressing Data

This chapter explains how Essbase stores and compresses data on disk.

This chapter includes the following sections:

Storage Allocation

Essbase uses a data file to store data blocks. By default, a data file is located in its associated database folder. Data files follow the naming convention `ESS`*`n`*`.PAG`, where *n* is greater than or equal to one and less than or equal to 65,535.

Essbase uses an index file to store the index for a database. By default, an index file is located in its associated database folder. Index files follow the naming convention `ESS`*`n`*`.IND`, where *n* is greater than or equal to one and less than or equal to 65,535.

Essbase automatically allocates storage for data and index files. You can use disk volumes to control how storage is allocated for these files.

➤ To specify disk volumes so that you control how storage is allocated, use this procedure:

1. Verify how much space Essbase uses to store index and data files. See "Checking Index and Data File Sizes" on page 1116 for information about how to check sizes.
2. Choose a technique to control storage:
    - Specify which volumes (drives) Essbase uses to store these files. See "Specifying Disk Volumes" on page 1116 for more information.
    - Install Essbase on one volume and store files on another volume. See *Essbase Installation Guide* for instructions.

Checking Index and Data File Sizes

You can view index file (`.IND` file) and data file (`.PAG` file) names, counts, sizes, and totals, and you can determine whether each file is presently open in Essbase. From Application Manager, select Database > Information and click the Files tab to view this information.

You can also use the Database Properties window and click Storage tab to view this information. For more information, see *Essbase Administration Services Online Help*.

You can use the LISTFILES command in ESSCMD to view index file and data file names, counts, sizes, and totals and to determine whether each file is presently open in Essbase. See the *Technical Reference* in the `docs` directory for information about the command.

Note: The file size information that is provided by the Windows NT operating system for index and data files that reside on NTFS volumes may not be accurate. The file size information provided by Application Manager and by LISTFILES is accurate.

Specifying Disk Volumes

Use disk volumes to specify where you want to store Essbase index files (`ESS`*`n`*`.IND`) and data files (`ESS`*`n`*`.PAG`). If you do not use the disk volumes setting, Essbase stores data only on the volume where the `ARBORPATH` directory resides. If the ARBORPATH variable is not set, Essbase stores data only on the volume where the server was started. For information on setting ARBORPATH, see the *Essbase Installation Guide*.

Note: For information about checking the size of the index and data files, see "Checking Index and Data File Sizes" on page 1116.

You can specify disk volumes using Application Manager (Database > Settings > Storage tab), Administration Services (Storage tab of the Database Properties window), or ESSCMD. When you use disk volumes, Essbase provides the following options for each vo.lume:

- Name of the volume
- Maximum space to use on the volume (called Partition Size in Application Manager and Volume Size in ESSCMD).
- File type. You can specify index files, data files, or both. The default is index and data files on the same volume.
- Maximum file size. The default and recommended value is 2,097,152 kilobytes (two gigabytes). When Essbase reaches the maximum file size, it creates a new file and names it incrementally. For example, when `ess00001.ind` is filled to maximum size, Essbase creates `ess00002.ind`.

CAUTION: If you specify a volume name but not a volume size, Essbase uses all available space on the volume.

Essbase creates new data files and index files in either of these situations:

- If the total sizes of all files reach the maximum size you have specified in the disk volumes setting. By default, the total is the sum of all index and data file sizes. If you specify Index as the file type, the total refers to the sum of all index files on a particular volume. Likewise, if you specify Data as the file type, the total refers to the sum of all data files on a particular volume.

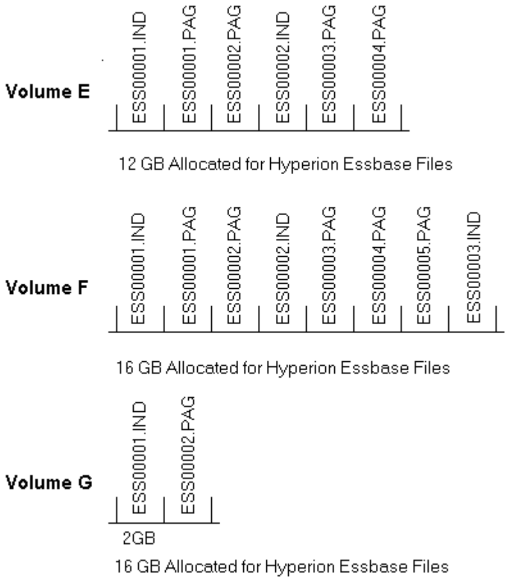
  For example, suppose you want to use up to 12 GB for Essbase files on volume E, 16 GB on volume F, and 16 GB on volume G. Essbase creates a new file on volume F when the sizes of the index and data files reach 12 GB on volume E and more data needs to be written out to disk.

- If the size of an individual index or data file on any volume reaches a size of 2 GB. In the above example, suppose volumes E and F have reached their capacities and Essbase is using volume G. Figure 503 illustrates this example.

  On volume G, Essbase creates file `ess00001.ind` and fills it to the default limit of 2 GB. On volume G, Essbase creates file `ess00001.pag` and fills it to 1 GB.

  You have specified a limit of 16 GB on volume G, and you have used 3 GB. You have 13 GB left to use on volume g, but `ess00001.ind` has reached the maximum file size of 2 GB. The next time Essbase needs storage space when writing index files to disk, Essbase creates a new file on volume G and names it `ess00002.ind`. Essbase then fills `ess00002.ind` to its 2 GB limit and creates `ess00003.ind`. Essbase follows the same procedures for data files.

Figure 503: Example of How Essbase Stores Files Across Volumes



Essbase names files consecutively, starting with `ESS00001.`*`xxx`*`,` where *xxx* is `IND` for an index file and `PAG` for a data file, and continuing up to `ESS65535.`*`xxx`*. This naming convention applies to each volume, so in the above example, volumes E, F, and G each have files named `ESS00001.PAG` and `ESS00001.IND`.

Keep in mind the following guidelines when specifying disk volumes:

- Specify the disk volumes in the order in which you want the volumes to be used. You do not have to specify the volume on which Essbase is installed as one of the volumes; you can install on one volume and store data on other volumes.
- If a volume reaches capacity, Essbase moves on to the next volume.
- If all specified volumes reach capacity, Essbase stops any ongoing database operations, issues an error message, and performs fatal error handling. For more information, see "Understanding Fatal Error Handling" on page 1460. If these events occur, shut down the database, allocate more disk space, and restart the database.
- You can tell Essbase to stop storing files on a particular volume. Essbase can still access the volume as needed, but it no longer stores additional index and data information on the volume. To stop storing information on a volume, select the volume definition that you want to remove, as shown in Figure 504, and click Delete.
- You set disk volumes on a per database basis. It is possible for more than one database to use space on the same volume, so allocate space carefully. For example, if you specify 7 GB on Volume A for Database 1 and 7 GB on Volume A for Database 2, you have allocated 14 GB for Essbase files on Volume A.
- For new files, changes to the disk volumes setting take effect when you next start the database. Existing files and volumes are not affected.

Specifying Disk Volumes with Application Manager

➤ To add a new volume or to change values for a previously allocated volume:

1. Select **Database > Settings > Storage**.
2. In the **Disk Volume** text box, enter the volume name.
3. Specify a value in the **Partition Size** box.

   Partition Size represents the maximum space that Essbase uses on the volume. The default value is Unlimited; the minimum is 8,192 kilobytes.
4. Specify a file type in the **File Type** box.

   You can specify index files, data files, or both. The default is Index + Data.

5. Specify a value in the **File Size** box.

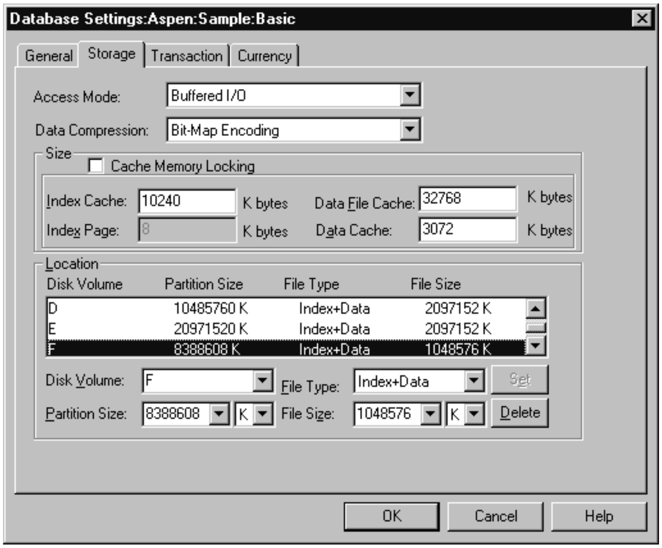
   File Size represents the maximum size that a file specified in File Type can attain before Essbase creates a new file. The default value is 2 GB (2097152 kilobytes); the minimum value is 8,192 kilobytes.

6. Click **Set**.

➤ To stop Essbase from storing additional files on an allocated volume, enter the volume name in the **Disk Volume** text box and click **Delete**.

Essbase writes no new files to that volume, but continues to access and use files previously created on that volume. For example, suppose you want to stop storing information on Volume F. Select the volume definition for volume F and click **Delete**, as shown in Figure 504.

Figure 504: De-allocating a Volume



Database Settings:Aspen:Sample:Basic

Access Mode: Buffered I/O

Data Compression: Bit-Map Encoding

Size

□ Cache Memory Locking

Index Cache: 10240 K bytes

Data File Cache: 32768 K bytes

Index Page: 8 K bytes

Data Cache: 3072 K bytes

Location

Disk Volume	Partition Size	File Type	File Size
D	10485760 K	Index+Data	2097152 K
E	20971520 K	Index+Data	2097152 K
F	8388608 K	Index+Data	1048576 K

Disk Volume: F

File Type: Index+Data

Partition Size: 8388608 K

File Size: 1048576 K

Now, Essbase stores no additional data on Volume F.

Specifying Disk Volumes with ESSCMD

➤ To allocate a new volume, enter `SETDBSTATEITEM 23` in ESSCMD and either follow the prompts or supply the required values on the command line.

ESSCMD prompts you for the number of new disk volumes you want to add, unless you supply the number on the command line.

Then, for each new volume, ESSCMD prompts you for the following values, unless you supply them on the command line.

- Volume name (for each volume).
- Volume size (maximum space to use on the volume)—The default value is Unlimited; the minimum setting is 8 megabytes.

  When you use ESSCMD, you can specify volume size in bytes (B), kilobytes (K), megabytes (M), gigabytes (G), or terabytes (T). ESSCMD displays minimum, maximum, and current values and 0 for unlimited.
- File type—You can specify index files, data files, or both. The default is `3-Index + Data` (index and data files on the same volume).
- File size (maximum size that each file specified in File Type can attain before Essbase creates a new file)—The default value is 2 gigabytes; the minimum setting is 8 megabytes.

  When you use ESSCMD, you can specify file size in bytes (B), kilobytes (K), megabytes (M), or gigabytes (G). ESSCMD displays minimum, maximum, and current values.

The following example allocates up to 10 gigabytes on Volume E, sets a maximum file size of 2 gigabytes, and specifies that data files should be stored only on E:

```
SETDBSTATEITEM 23 "JILLAPP" "JILLDB" "1" "E" "10G" "2" "2G"
```

➤ To change the settings on an allocated volume, enter `SETDBSTATEITEM 24` in ESSCMD and either follow the prompts or supply the required values on the command line.

ESSCMD prompts you for the following values, unless you supply them on the command line:

- Volume number. (Use the GETDBSTATE command in ESSCMD to see a list of the currently defined disk volumes and to see the number assigned to each volume.)
- Volume name
- Volume size
- File type
- File size

The following example allocates up to 20 gigabytes on Volume C and sets a maximum file size of 2 gigabytes:

```
SETDBSTATEITEM 24 "JILLAPP" "JILLDB" "1" "C" "20G" "3" "2G"
```

➤ To stop Essbase from storing additional files on a volume, enter `SETDBSTATEITEM 25` in ESSCMD and either follow the prompts or supply the required values on the command line. Essbase continues accessing files on the deallocated volume, but does not write new files to it.

ESSCMD prompts you for the following value, unless you supply it on the command line: Delete which volume definition. Use the GETDBSTATE command in ESSCMD to see a list of the currently defined disk volumes and to see the number assigned to each volume.

The following example deallocates the volume that is specified as fourth:

```
SETDBSTATEITEM 25 "JILLAPP" "JILLDB" "4"
```

For more syntax information, see the *Technical Reference* in the `docs` directory.

On UNIX, *volume_name* is a mounted UNIX file system. You must enter a fully qualified path name up to the name of the directory you are using for Essbase. Essbase automatically appends the `\app` directory to the path; you do not specify the `\app` directory.

Consider the following examples:

```
/vol2/essbase 10M
```

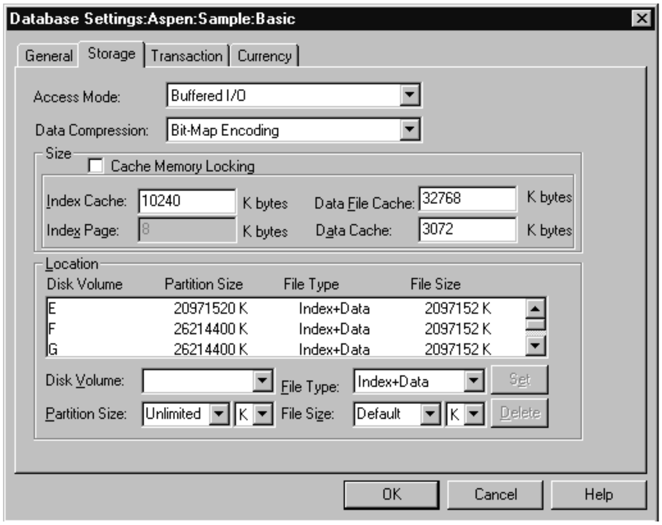
Volume size is the maximum space, in kilobytes, allocated to the volume. The default value is unlimited—Essbase uses all available space on that volume.

Example: Specifying Volumes To Control Storage

Assume you want to use up to 20 GB for Essbase files on Volume E, 25 GB on Volume F, and 25 GB on Volume G. You are using the default file size limit of 2 GB.

When you load data, Essbase stores up to 20 GB on Volume E; if the database is larger than 20 GB, Essbase stores the next 25 GB on Volume F, and so on.

Figure 505: Example of Using Disk Volumes



Access Mode: Buffered I/O

Data Compression: Bit-Map Encoding

☐ Cache Memory Locking

Index Cache: 10240 K bytes

Data File Cache: 32768 K bytes

Index Page: 8 K bytes

Data Cache: 3072 K bytes

Disk Volume	Partition Size	File Type	File Size
E	20971520 K	Index+Data	2097152 K
F	26214400 K	Index+Data	2097152 K
G	26214400 K	Index+Data	2097152 K

Disk Volume:

File Type: Index+Data

Partition Size: Unlimited K

File Size: Default K

Data Compression

Essbase allows you to choose whether or not data blocks that are stored on disk are compressed, as well as which compression scheme to use. When data compression is enabled, Essbase compresses data blocks when it writes them out to disk. Essbase fully expands the compressed data blocks, including empty cells, when the blocks are swapped into the data cache.

Generally, data compression optimizes storage use. You can check compression efficiency by checking the compression ratio statistic. See "Checking the Compression Ratio" on page 1129 for more information.

Essbase provides two options for data compression:

- Bitmap compression, the default. Essbase stores only non-missing values and uses a bitmapping scheme.
- Run-length encoding (`RLE`). Essbase compresses repetitive, consecutive values, including zeros and `#MISSING` values.

Because Essbase compresses data blocks as they are written to disk, it is possible for bitmap, RLE, and non-compressed data blocks to coexist in the same data file. Keep in mind the following:

- When a compressed data block is brought into the data cache, Essbase expands the block to its full size, regardless of the scheme that was used to compress it.
- When Essbase stores a block on disk, Essbase treats the block the same whether the block was compressed or uncompressed when it was brought into the data cache. In either case, Essbase compresses the block according to the specified compression type (including not compressing the block if No Compression is specified).
- If compression is not enabled, Essbase writes out the fully expanded block to disk.

You may want to disable data compression if blocks have very high density (90% or greater) and have few consecutive, repeating data values. Under these conditions, enabling compression consumes resources unnecessarily.

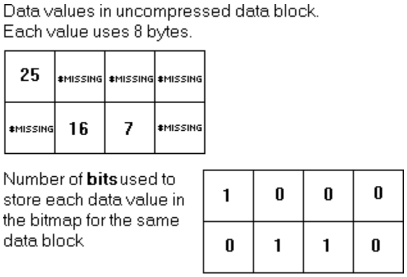
Bitmap Data Compression

With bitmap compression, Essbase uses a bitmap to represent data cells, and stores only the bitmap, the block header, and the other control information. A bitmap uses one bit for each cell in the data block, whether the cell value is missing or non-missing. When a data block is not compressed, Essbase uses 8 bytes to store every non-missing cell.

When using bitmap compression, Essbase stores only non-missing values and does not compress repetitive values or zeros (contrast with RLE compression, described in "RLE Data Compression" on page 1126). When Essbase pages a data block into the data cache, it fully expands the data block, using the bitmap to recreate the missing values.

Because the bitmap uses one bit for each cell in the data block, the bitmap scheme provides a fixed overhead for data compression. Figure 506 represents a portion of a data block, as an example. In this example, Essbase uses 64 bytes to store the data in the fully expanded block, but uses one byte (eight bits) to store the bitmap of the compressed data on disk. (Essbase also uses a 72-byte block header for each block, whether the block is compressed or not.)

Figure 506: Bitmap Data Compression



Data values in uncompressed data block.
Each value uses 8 bytes.

25	#MISSING	#MISSING	#MISSING
#MISSING	16	7	#MISSING

Number of **bits** used to store each data value in the bitmap for the same data block

1	0	0	0
0	1	1	0

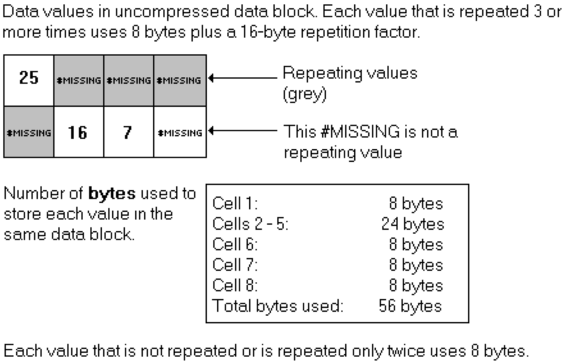
In most cases, bitmap compression conserves disk space more efficiently. However, much depends on the configuration of the data.

RLE Data Compression

When using the run-length encoding (RLE) compression scheme, Essbase compresses any consecutive, repetitive values—any value that repeats three or more times consecutively, including zero. Essbase keeps track of each repeating value and the number of times it is repeated consecutively.

In the example in Figure 507, Essbase uses 64 bytes to store the data in the fully expanded block, but uses 56 bytes to store the compressed data on disk. (Essbase also uses a 72-byte block header for each block, whether the block is compressed or not.)

Figure 507: RLE Data Compression



Data values in uncompressed data block. Each value that is repeated 3 or more times uses 8 bytes plus a 16-byte repetition factor.

25	#MISSING	#MISSING	#MISSING
#MISSING	16	7	#MISSING

Number of **bytes** used to store each value in the same data block.

Cell 1:	8 bytes
Cells 2 - 5:	24 bytes
Cell 6:	8 bytes
Cell 7:	8 bytes
Cell 8:	8 bytes
Total bytes used:	56 bytes

Each value that is not repeated or is repeated only twice uses 8 bytes.

The RLE compression type may be preferable under the following circumstances:

- The average block density in the database is very low (three percent or lower).
- The database has many consecutive zero values or any other consecutive, repeating value.

Changing Data Compression Settings

In Application Manager and Administration Services, data compression is enabled by default. You can view the current settings by selecting Database > Settings and clicking the Storage tab or by opening the Database Properties window and clicking the Storage tab.

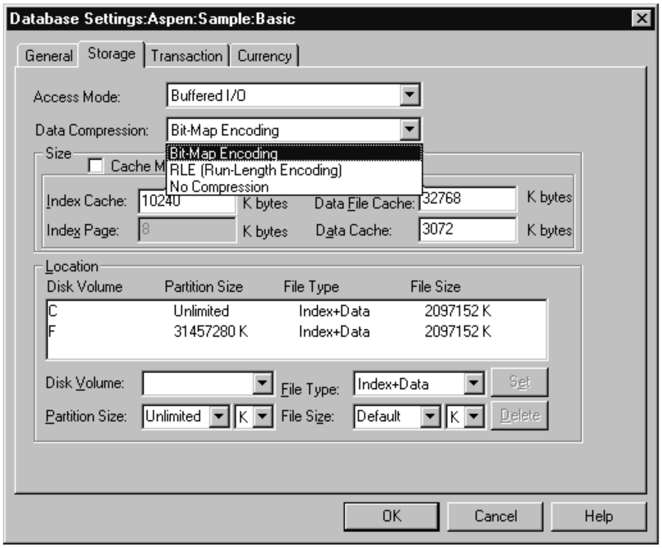
Changes to the data compression setting take effect immediately as Essbase writes data blocks to disk. For blocks already on disk, Essbase does not change compression schemes or enable or disable compression. When you change the data compression settings of blocks already on disk, Essbase uses the new compression scheme the next time Essbase accesses, updates, and stores the blocks.

Changing Data Compression Settings with Application Manager

➤ To specify data compression:

1. Select **Database > Settings**
2. Click the **Storage** tab.
3. Select a value from the **Data Compression** list box:

Figure 508: Setting Data Compression



Database Settings:Aspen:Sample:Basic

Access Mode: Buffered I/O

Data Compression: Bit-Map Encoding

- Bit-Map Encoding
- RLE (Run-Length Encoding)
- No Compression

Size

☐ Cache M

Index Cache: 10240 K bytes

Data File Cache: 32768 K bytes

Index Page: 8 K bytes

Data Cache: 3072 K bytes

Location

Disk Volume	Partition Size	File Type	File Size
C	Unlimited	Index+Data	2097152 K
F	31457280 K	Index+Data	2097152 K

Disk Volume:

File Type: Index+Data

Partition Size: Unlimited K

File Size: Default K

Changing Data Compression Settings with ESSCMD

You can enable or disable data compression by using `SETDBSTATEITEM` or `SETDBSTATE`. To set the data compression type, you must use `SETDBSTATEITEM`.

Using SETDBSTATEITEM

➤ To enable or disable data compression, enter `SETDBSTATEITEM 14` in ESSCMD and either follow the prompts or supply the required values on the command line.

ESSCMD prompts you for the following values, unless you supply them on the command line:

- Data Compression on Disk? Enter `Y` (Yes, the default) or `N` (No).
- Data Compression Type. Enter `1` (run-length encoding) or `2` (bitmap, the default).

➤ To specify the data compression type, enter `SETDBSTATEITEM 15` in ESSCMD and either follow the prompts or supply the required values on the command line. ESSCMD prompts you for a value of “1” (run length encoding) or “2” (bitmap, the default).

The following example enables Bitmap compression:

```
SETDBSTATEITEM 14 "JILLAPP" "JILLDB" "Y" "2"
```

For more syntax information, see the *Technical Reference* in the `docs` directory.

Using SETDBSTATE

➤ To enable data compression:

1. Enter `SETDBSTATE` in ESSCMD.
2. Follow the prompts until the last prompt, Data Compression on Disk?
3. Enter `Y` (Yes, the default) or `N` (No) and press Enter.

To set the data compression type, you must use SETDBSTATEITEM or the Database Settings dialog box.

For more syntax information, see the *Technical Reference* in the `docs` directory.

Checking the Compression Ratio

The compression ratio represents the ratio of the compressed block size (including overhead) to the uncompressed block size, regardless of the compression type in effect. Overhead is the space required by mechanisms that manage compression/expansion.

In Application Manager, select Database > Information and click the Statistics tab to check the compression ratio.

Tip: You can also check the compression ratio without using the Application Manager.

Tool	Instructions	For more information
Administration Services	Click the Statistics tab on the Database Properties window	*Essbase Administration Services Online Help*
ESSCMD	GETDBSTATS	*Technical Reference* in the `docs` directory

Note: The larger the number, the more compression. The compression ratio can vary widely from block to block.

Data Block Size

Data block size is determined by the amount of data in a particular combination of dense dimensions. For example, when you change the dense or sparse configuration of one or more dimensions in the database, the data block size changes. Data block size is 8*n* bytes, where *n* is the number of cells that exist for that combination of dense dimensions.

Note: Eight to 100 kilobytes is the optimum size range.

For information about determining the size of a data block, see "Size of Expanded Data Block" on page 1469.

Tip: You can use any of the following tools to view the block size for a database.

Tool	**Instructions**	**For more information**
Application Manager	Select Database > Information and click the Statistics tab to see the value: block size in bytes.	*Essbase Application Manager Online Help*
Administration Services	Click the Statistics tab on the Database Properties window	*Essbase Administration Services Online Help*
ESSCMD	GETDBSTATS. See the value: actual block size.	*Technical Reference* in the `docs` directory

Chapter 40 Ensuring Data Integrity

This chapter describes how Essbase handles transactions and locking and other ways that Essbase protects data.

This chapter includes the following sections:

If you are migrating from a previous release of Essbase, see the *Essbase Installation Guide*.

Transactions

When a database is in read/write mode, Essbase considers every update request to the server (such as a data load, a calculation, or a statement in a calculation script) as a transaction. Essbase tracks information about transactions in a transaction control file (`*dbname*.TCT`).

The transaction control file contains an entry for each transaction and tracks the current state of each transaction (Active, Committed, or Aborted).

For more information, see "How Essbase Handles Transactions" on page 1140.

Isolation Levels

Isolation levels determine how Essbase commits data to disk. When data is committed, it is taken from server memory and written to the database on disk. Essbase automatically commits data to disk. There are no explicit commands that users perform to commit data blocks. However, setting the isolation level for a database defines how Essbase automatically commits data blocks.

Essbase offers two isolation levels for transactions: *committed access* and *uncommitted access* (the default). You can optimize data integrity by using committed access.

For information about committed access, see "Committed Access" on page 1134. For information about uncommitted access, see "Uncommitted Access" on page 1137.

Data Locks

Essbase issues write (exclusive) locks for blocks that are created, updated, or deleted, and issues read (shared) locks for blocks that need to be accessed but not modified. By issuing the appropriate locks, Essbase ensures that data changed by one operation cannot be corrupted by a concurrent update.

Essbase locks data blocks, not objects. For information about locking and unlocking outlines and other objects, see "Locking and Unlocking Objects" on page 1200.

This table explains the lock types:

Table 41: Basic Lock Types

Lock	Description
Write (exclusive) lock	Prevents any other transaction from accessing the locked data block. Used for all data block updates, including spreadsheet lock&send.
Read (shared) lock	Allows other transactions read-only access to the data block, but prevents other transactions from modifying the data block.

This table shows the locks that Essbase issues for various types of operations.

Table 42: Locking by Higher-Level Functions

Type of Operation	Lock Issued
Spreadsheet retrieve	Read (shared) lock on each data block.
Retrieve and lock	Write (exclusive) lock on all affected blocks. A subsequent send command commits the data.
Calculate derived block	Write lock on the block being calculated. As a block is calculated, all blocks containing the block's children acquire read locks.
Data load	Write lock.
Restructure	Write lock.

How Essbase handles locking depends on whether committed or uncommitted access is enabled.

Committed Access

Committed access provides a high level of data consistency because only one transaction at a time can update data blocks. Under committed access, Essbase allows transactions to hold read/write locks on all data blocks involved with the transaction until the transaction completes and commits. However, you can still allow read-only access to the last committed data values.

Essbase provides options that determine when locks are issued on data blocks:

- Pre-image access (enabled by default). Pre-image access provides users read-only access to data blocks that are locked for the duration of a concurrent transaction. Users see the last committed data values for the locked data blocks.
- Wait (dialog box) or time out (ESSCMD):
  - Indefinite wait (the default): The transaction waits to acquire a lock on the required locked block.
  - Immediate access, or no wait: If a required block is locked by another transaction, Essbase displays a lock time out message, and the transaction aborts.
  - A number of seconds that you specify: The transaction waits that number of seconds to acquire a lock on the required locked blocks. If the specified time runs out before the transaction acquires a lock, Essbase displays a lock time out message, and the transaction aborts.

When you have pre-image access enabled, you are not limited to read-only access to data blocks; if you need write access to locked blocks, your transaction waits for write access or times out, depending on the wait or time-out setting. Your transaction gets immediate write access to data blocks that are not locked by another transaction.

If you do not have pre-image access enabled and if you need read or write access to locked blocks, your transaction waits for write access or times out, depending on the wait or time-out setting.

CAUTION: Under committed access, Essbase retains redundant data until a transaction commits. Allow disk space for double the size of the database to accommodate redundant data.

Locking Under Committed Access

Under committed access, Essbase locks blocks for read and write access:

- For read access, the lock remains until another transaction requests it, whether or not the transaction is complete. Other transactions can read the locked block, but none can alter it.
- For write access, a transaction locks and holds the lock on each block that it modifies until the transaction completes.

Table 43 illustrates locking behavior under committed access when more than one transaction is contending for a lock on the same data. In the example in Table 43, transaction Tx1 is running, and transaction Tx2 is requesting access to the same data.

Note that access to locked blocks depends on what options are enabled. For a list of options, see "Committed Access" on page 1134.

Table 43: Locking Behavior Under Committed Access

		Tx1 holds read lock; **Tx2 requests read lock**	**Tx1 holds read lock;** **Tx2 requests write lock**	**Tx1 holds write lock;** **Tx2 requests read lock**	**Tx1 holds write lock;** **Tx2 requests write lock**
Pre-image access enabled	Wait (time-out) period specified (indefinite wait or a number of seconds wait)	Tx2 gets read lock.	Tx2 waits for Tx1 to release read lock.	Tx2 gets pre-image access.	Tx2 waits for Tx1 to release write lock.
	No wait (time-out) period specified (immediate time-out)	Tx2 gets read lock.	Essbase issues time-out message and aborts the transaction.	Tx2 gets pre-image access.	Essbase issues time-out message and aborts the Tx2 transaction.

Table 43: Locking Behavior Under Committed Access

		Tx1 holds read lock; Tx2 requests read lock	**Tx1 holds read lock; Tx2 requests write lock**	**Tx1 holds write lock; Tx2 requests read lock**	**Tx1 holds write lock; Tx2 requests write lock**
No pre-image access	Wait (time-out) period specified (indefinite wait or a number of seconds wait)	Tx2 gets read lock.	Tx2 waits for Tx1 to release read lock.	Tx2 waits for Tx1 to release write lock.	Tx2 waits for Tx1 to release write lock.
	No wait (time-out) period specified (immediate time-out)	Tx2 gets read lock.	Essbase issues time-out message and aborts the Tx2 transaction.	Essbase issues time-out message and aborts the Tx2 transaction.	Essbase issues time-out message and aborts the Tx2 transaction.

For information about how to set concurrency parameters, see "Specifying Data Integrity Settings" on page 1141.

Concurrency with Committed Access

Occasionally under committed access, a situation results when two transactions are locking or waiting for access to the same blocks, and neither transaction can complete under these conditions. This called a deadlock.

For example, if transaction Tx1 needs to update first data block B1 and then data block B2, it first locks B1 and then attempts to lock B2. Meanwhile, if transaction Tx2 needs to update first data block B2 and then block B1, Tx2 first locks B2 and then attempts to lock B1. Tx1 locked B1 and is waiting for B2, and Tx2 locked B2 and is waiting for B1.

Essbase transactions periodically perform deadlock detection while waiting to acquire a lock. If detected, Essbase issues an error message, and the transaction aborts.

If you try to update a block that is locked by another user, Essbase behaves as follows:

- If wait is set to indefinite, your transaction waits to acquire the needed locks.
- If wait is set to 0 (immediate) and if the required blocks are not immediately available, Essbase displays an error message, and aborts the transaction.
- If wait is set to a user-specified number of seconds and the time has expired, Essbase displays an error message and aborts the transaction.
- If your request times out, try the operation again.

For information about how to set concurrency options, see “Specifying Data Integrity Settings” on page 1141.

Rollback with Committed Access

Under committed access, if the server crashes, Essbase rolls back all database updates by transactions that were in progress when the server stopped. Thus, Essbase ensures that changes made by the aborted transactions are undone.

If a transaction is aborted due to a non-fatal error, all changes made by the transaction are rolled back.

For more information, see “Recovering from a Crashed Database” on page 1147.

Uncommitted Access

With uncommitted access (enabled by default), the Essbase kernel allows transactions to hold read/write locks on a block-by-block basis; Essbase releases a block after it is updated but does not commit blocks until the transaction completes or until a specified limit (a “synchronization point”) has been reached. You can set this limit, as described below.

Concurrent users accessing the same data blocks might experience unexpected results under uncommitted access, because Essbase allows read-only access to data at its last commit point.

With uncommitted access, you can control when Essbase performs an explicit commit operation by specifying synchronization point parameters:

- Commit Blocks (number of blocks modified before a synchronization point occurs). The default is 3,000.

  If you set Commit Blocks to 0, the synchronization point occurs at the end of the transaction.

- Commit Rows (number of rows to data load before a synchronization point occurs). The default is 0, which means that the synchronization point occurs at the end of the data load.

- If either Commit Blocks or Commit Rows has a non-zero value, a synchronization point occurs when the first threshold is reached. For example, if Commit Blocks is 10 but Commit Rows is 0 and you load data, a synchronization point occurs after 10 blocks are updated. If Commit Blocks is 5 and Commit Rows is 5 and you load data, a synchronization point occurs after 5 rows are loaded, or 5 blocks are updated, whichever happens first.

If a user-defined threshold is exceeded during an operation, Essbase issues a synchronization point to commit the data processed to that point. Essbase performs as many synchronization points as are necessary to complete the operation.

Note: Essbase analyzes the value of Commit Blocks and Commit Rows during its analysis of feasibility for parallel calculation use. If Essbase finds the values set too low, it will automatically increase them.

For information about how to specify synchronization point parameters, see "Specifying Data Integrity Settings" on page 1141.

CAUTION: Essbase retains redundant data to enforce transactional semantics. Allow disk space for double the size of the database to accommodate redundant data, particularly if both Commit Blocks and Commit Rows are set to 0.

Locking Under Uncommitted Access

Under uncommitted access, Essbase locks blocks for write access until Essbase finishes updating the block. This is in contrast to committed access, when Essbase holds locks until a transaction completes.

Table 44 illustrates locking behavior under uncommitted access when more than one transaction contends for a lock on the same data. In the example in Table 44, transaction Tx1 is running, and transaction Tx2 is requesting access to the same data.

Table 44: Locking Behavior with Uncommitted Access

Status When Tx2 Makes a Request	If Tx1 holds read lock	If Tx1 holds write lock
Read lock	Tx2 gets read lock.	Tx2 gets read lock.
Write lock	Tx2 gets write lock.	Tx2 waits for Tx1 to release the lock.

Concurrency with Uncommitted Access

With uncommitted access, blocks are released more frequently than with committed access, when all blocks are locked until the end of the transaction.

Rollback with Uncommitted Access

Under uncommitted access, if the server crashes, Essbase rolls back all database updates from the point of the last successful commit. Some of the updates from an aborted transaction may have committed. Whether transactions committed their updates the way users expected depends on the order in which overlapping transactions updated and committed data.

If a transaction is aborted due to a non-fatal error, Essbase commits only the data that the transaction finished processing prior to the abort of the transaction.

For more information, see “Recovering from a Crashed Database” on page 1147.

Parallel Calculation and Uncommitted Access

If Essbase is using parallel calculation, it will check the commit threshold.

Committed Versus Uncommitted Access

Consider these issues when choosing an isolation level:

Table 45: Issues Affecting Selection Of An Isolation Level

Issue	Explanation
Database Performance	Uncommitted access always yields better database performance than committed access. When using uncommitted access, Essbase does not create locks that are held for the duration of a transaction but commits data based upon short-term write locks.
Data Consistency	Committed access provides a higher level of data consistency than uncommitted access. Retrievals from a database are more consistent. Also, only one transaction at a time can update data blocks when the isolation level is set to committed access. This factor is important in databases where multiple transactions attempt to update the database simultaneously.
Data Concurrency	Uncommitted access provides better data concurrency than does committed access. Blocks are released more frequently than they are during committed access. With committed access, deadlocks can occur.
Database Rollbacks	If a server crash or other server interruption occurs while there are active transactions running, the Essbase kernel rolls back the transactions when the server is restarted. With committed access, rollbacks return the database to its state before any transactions began. With uncommitted access, rollbacks may result in some data being committed and some data not being committed. For information about actions to take when a transaction does not complete, see “What to Expect If a Server Interruption Occurs” on page 1148.

How Essbase Handles Transactions

Essbase tracks transactions from start to finish, swapping data blocks in and out of memory as needed and committing data blocks when a transaction completes. The following list describes how Essbase handles a transaction: all list items apply to both committed and uncommitted access (see “Isolation Levels” on page 1132).

1. A user or batch program begins an operation.
2. The OLAP engine notifies the Essbase kernel that a transaction is to begin.
3. The Essbase kernel begins the transaction.
4. The OLAP engine requests data from the Essbase kernel.

5. The Essbase kernel locates the requested data. It passes the data, and some associated control information, to the OLAP engine. If you are using Spreadsheet Add-in, you see this data displayed on the sheet.
6. If you are using Spreadsheet Add-in, when you modify data, you issue the Send command.
7. The Essbase kernel associates the transaction with an entry in its transaction control table.
8. After the operation is complete on the OLAP engine side, the OLAP engine notifies the Essbase kernel about the update, and the Essbase kernel updates internal data structures accordingly.
9. Steps 4 through 8 repeat as often as necessary to complete the operation.
10. The transaction ends. If Essbase encounters an error during transaction processing, it aborts the transaction. If no errors are encountered, Essbase commits the transaction. See "Isolation Levels" on page 1132 for details on the differences in commit behavior under committed and uncommitted access.
11. Essbase issues a message to notify the client that the transaction is complete; for example, "TOTAL CALC ELAPSED TIME..."

Under uncommitted access, it is possible to access uncommitted data when multiple transactions are active and are accessing the same data. Transaction results are unpredictable under uncommitted access.

Under uncommitted access, if you have defined a commit threshold, Essbase may need to break down a single database operation into multiple synchronization points. See "Uncommitted Access" on page 1137 for information on commit thresholds.

Specifying Data Integrity Settings

You can specify isolation level, synchronization point parameters, and concurrency parameters using Application Manager, the Transactions tab of the Database Properties window in Administration Services, or ESSCMD. Changes to isolation level settings take effect the next time there are no active transactions. For information about deciding which settings to choose, see "Isolation Levels" on page 1132.

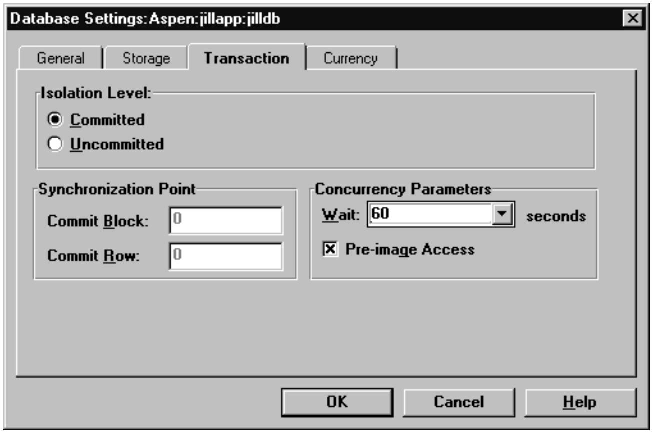
Specifying Settings with Application Manager

➤ To specify isolation level settings in Application Manager use this procedure:

1. Select **Database > Settings** and click the **Transaction** tab.
2. In the **Isolation Leve**l group, select **Committed** or **Uncommitted**.
3. Specify **Synchronization Point** parameters or **Concurrenc**y parameters:
   - If you select Committed access, the Concurrency Parameters option group is available. Specify a Wait time and select Pre-image Access, if desired. For more information on these parameters, see “Committed Access” on page 1134.
   - If you select Uncommitted access, the Synchronization Point option group is available. Specify a number for Commit Block and/or Commit Row. For more information on these parameters, see “Uncommitted Access” on page 1137.
4. Click **OK**.

For example, suppose you want committed access, pre-image access, and a 60-second wait for locked data blocks. Specify settings thus:

Figure 509: Example of Setting Committed Isolation Level



Database Settings:Aspen:jillapp:jilldb

Isolation Level:

● Committed

○ Uncommitted

Synchronization Point

Commit Block: 0

Commit Row: 0

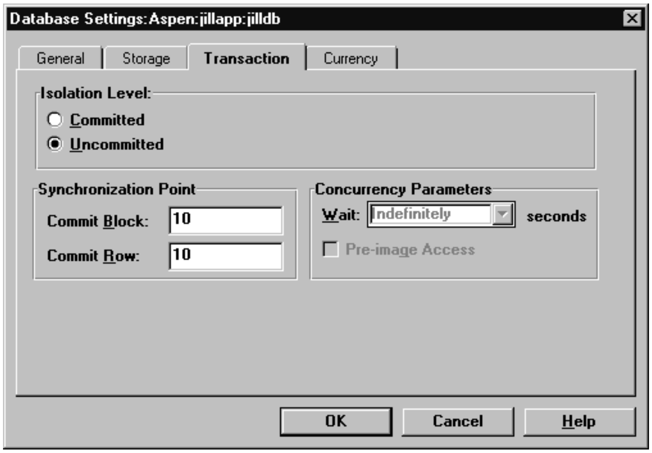
Concurrency Parameters

Wait: 60 seconds

☒ Pre-image Access

As another example, suppose that you want uncommitted access and that you want Essbase to perform a commit operation after 10 blocks are updated, or after 10 rows are loaded. Specify settings as in Figure 510:

Figure 510: Example of Setting Uncommitted Isolation Level



Database Settings:Aspen:jillapp:jilldb

Isolation Level:
○ Committed
● Uncommitted

Synchronization Point
Commit Block: 10
Commit Row: 10

Concurrency Parameters
Wait: Indefinitely seconds
□ Pre-image Access

Specifying settings with ESSCMD

➤ To specify isolation level settings using ESSCMD, enter `SETDBSTATEITEM 18` in ESSCMD and either follow the prompts or supply the required values on the command line.

Choose 1 (committed access) or 2 (uncommitted access, the default). Depending on the type of access that you specify, ESSCMD prompts you for other parameters (or you can supply the values on the command line).

If you choose 1 (committed access), ESSCMD prompts for the following information:

- Pre-image access; Y (Yes) or N (No, the default). Pre-image access provides users read-only access to data blocks that are locked for the duration of a transaction. Users see the last committed data values for the locked data blocks.

- Wait (in the Database Settings dialog box) or time out (in ESSCMD): -1, 0, or *n*.
  - -1 is indefinite wait.
  - 0 is immediate access, or no wait.
  - *n* is a number of seconds that you specify.

If you choose 2 (uncommitted access), ESSCMD prompts for the following values. See "Uncommitted Access" on page 1137 for important details about these options.

- Number of blocks modified before internal commit
- Number of rows to data load before internal commit

You can also specify isolation level parameters (pre-image access and so on) by specifying parameters 19–22 on SETDBSTATEITEM. Enter `SETDBSTATEITEM` with no parameters; ESSCMD displays a list that includes each parameter by number, with a description.

Here is an example of using SETDBSTATEITEM to set an isolation level. This example enables committed access and pre-image access and specifies indefinite wait time.

```
SETDBSTATEITEM 18 "JILLAPP" "JILLDB" "1" "Y" "-1"
```

For more syntax information, see the *Technical Reference* in the `docs` directory.

Specifying Settings with MaxL

To specify isolation level settings using MaxL, use this MaxL statement:

```
alter database dbs_name enable committed_mode
```

For more information, see the *Technical Reference* in the `docs` directory, list of MaxL statements

Accommodating Data Redundancy

To ensure data integrity, the Essbase kernel temporarily retains redundant (duplicate) information. To accommodate redundant information, allow disk space for double the size of your database.

Essbase maintains a file called `dbname.ESM`, in which it stores crucial control information.

CAUTION: The `dbname.TCT` file, `dbname.ESM` file, the index files, and the data files contain information crucial for data recovery. Never alter or delete these files.

Checking Structural and Data Integrity

To validate database integrity and to check for database corruption, use one of the following methods:

- Perform a dense restructure. Because a dense restructure recreates all blocks within a database, this method verifies index nodes and cells for each block.
- Export all levels of data from the database. Exporting an entire database accesses blocks and all data values across the entire database.
- Use the VALIDATE command (ESSCMD) to check structural and data integrity. See “Using VALIDATE to Check Integrity” on page 1146.

If errors occur during any of these checks, you need to restore the database from backups. For more information, see Chapter 44, “Backing Up and Restoring Data.”

Using VALIDATE to Check Integrity

The VALIDATE command performs many structural and data integrity checks:

- Verifies the structural integrity of free space information in the index.
- Compares the data block key in the index page with the data block key in the corresponding data block.
- The Essbase index contains an entry for every data block. For every Read operation, VALIDATE automatically compares the index key in the index page with the index key in the corresponding data block and checks other header information in the block. If it encounters a mismatch, VALIDATE displays an error message and continues processing until it checks the entire database.
- Restructures data blocks whose restructure was deferred with incremental restructuring. For more information, see "Incremental Restructuring and Performance" on page 1348.
- Checks every block in the database to make sure each value is a valid floating point number.
- Verifies the structural integrity of the linked reporting objects (LROs) catalog.

As Essbase encounters mismatches, it records error messages in the VALIDATE error log file. You can specify a file name for error logging; Essbase prompts you for this information if you do not provide it. The VALIDATE utility continues running until it has checked the entire database.

You can use the VALIDATE command in ESSCMD to perform these structural integrity checks. See the *Technical Reference* in the `docs` directory for information about the command. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

During index free space validation, the VALIDATE command verifies the structural integrity of free space information in the index. If integrity errors exist, Essbase records them in the VALIDATE log file. The file that you specified on the VALIDATE command holds the error log.

If VALIDATE detects integrity errors regarding the index free space information, the database must be rebuilt. You can rebuild in any one of three ways:

- Restoring the database from a recent system backup. See Chapter 44, "Backing Up and Restoring Data."

- Exporting data from the database; creating a new, empty database; and loading the exported data into the new database. See Chapter 44, “Backing Up and Restoring Data.”
- Restructuring the database. See Chapter 49, “Optimizing Database Restructuring.”

Even if you do not use VALIDATE, Essbase automatically performs certain validity checking whenever a read operation is performed, to ensure that the index is properly synchronized with the data.

For every read operation, Essbase compares the data block key in the index page with the data block key in the corresponding data block and checks other header information in the block.

If Essbase encounters a mismatch, it displays an “Invalid block header” error message.

Recovering from a Crashed Database

After a server interruption such as a crash, Essbase recovers a database, rolling back all transactions that were active when the interruption occurred. Recovery time depends on the size of the index. The larger the index, the longer it takes to recover the database. Essbase also recovers and consolidates free fragments (unused addressable units in the data blocks).

Essbase recovers data as soon as the server is started after a server interruption. Recovery consists of the following phases:

1. Transaction recovery rolls back all transactions that were active when the interruption occurred.
2. Index file recovery truncates files to their previously committed sizes.
3. Data free space recovery rebuilds the data free space tables. The size of the index determines the duration of this phase.

Only a media failure (faulty disk, disk failure, or head crash) requires you to restore data from backups. For more information, see Chapter 44, "Backing Up and Restoring Data."

CAUTION: Do not move, copy, modify, or delete any of the following files: `ESS`*`xxxxx`*`.IND`, `ESS`*`xxxxx`*`.PAG`, *`dbname`*`.IND`, *`dbname`*`.ESM`, *`dbname`*`.TCT`. Doing so can result in data corruption.

The Essbase kernel uses fatal error handling to display appropriate messages and to shut down the server, depending on the error encountered. For more information, see "Understanding Fatal Error Handling" on page 1460.

For information about how transactions are rolled back after a crash, see "Committed Versus Uncommitted Access" on page 1140.

What to Expect If a Server Interruption Occurs

This table lists types of server interruptions and their results:

Table 46: Essbase Recovery Handling

Type of Interruption	**Result**
• Power loss on server machine • Operating system crash • Server stopped with Ctrl + C keys	Server stops. When you restart the server, Essbase recovers the database.
• Operation cannot complete due to system limitations • Memory shortage • Out of disk space • Application stopped with Ctrl+C keys	Essbase performs fatal error handling. You may need to allocate more memory or disk space and restart the server.
Server crash	Essbase Exception Manager creates error log file of type `.XCP`. Server stops. When you restart the server, Essbase recovers the database.

Table 47 shows what you need to do if a server interruption occurs during a transaction. How Essbase recovers from an interruption depends on the transaction isolation level setting (committed or uncommitted access). See "Isolation Levels" on page 1132 for more information.

Table 47: Recovery Procedures for Server Requests

Type of Request	**Recommended Action**
Lock (for spreadsheet update)	Issue the lock command again.
Send (spreadsheet update)	If Essbase issues an error, repeat the last send operation. If the spreadsheet has been lost or does not exist, and if you are using SSAUDIT spreadsheet logging, reload the `dbname.ATX` file. See "Spreadsheet Update Logging" on page 1150.
Calculation	Check the server and application even logs to see where the calculation left off. Decide whether to start the calculation over. Repeat the last calculation.
Data load	Repeat the last data load (see Chapter 23, "Performing and Debugging a Data Load") or load the error log file (see Chapter 50, "Optimizing Data Loads").
Arithmetic data load (adding to or subtracting from values in the database)	If the database is set to committed access, reload the data. (The transaction has been rolled back.) If the database is set to uncommitted access, the process is not as simple. Some of the data loaded. Therefore, if you reload all of the data, you receive incorrect results for the data values that loaded twice. Therefore, you must perform the following actions: • Clear the database. • Restore the database to its state before the load. • Rerun the load. See Chapter 23, "Performing and Debugging a Data Load" and Chapter 50, "Optimizing Data Loads."
Restructure	The restructure is not complete. First, delete the temporary restructure files: `.pan`, `.inn`, and `.otn`. Repeat the last operation that caused a restructure.

Note: If the UPDATECALC parameter is set to FALSE, Essbase recalculates the entire database if an interruption occurs during a calculation. (The parameter's default value is TRUE.) For more information on UPDATECALC, see the *Technical Reference* in the `docs` directory.

Spreadsheet Update Logging

For extra protection against data loss and for spreadsheet audit information, Essbase provides a spreadsheet update logging facility. Enable this facility by using the SSAUDIT or SSAUDITR parameter in the `ESSBASE.CFG` server file. You can specify SSAUDIT for all databases on the server or for individual databases. For information on the `ESSBASE.CFG` file and for syntax information, see the *Technical Reference* in the `docs` directory.

Essbase handles recovery under normal situations. However, sometimes you may want to load the spreadsheet update log file manually. For example, if you have restored from a recent backup and do not want to lose changes made since the backup was made or you experience a media failure, you can recover transactions from the update log file. To do so, use the Essbase command-line facility, ESSCMD, from the server console.

The following ESSCMD command sequence loads the update log file:

```
LOGIN hostnode username password
SELECT application_name database_name
LOADDATA 3 filepath:application_name.ATX
EXIT
```

To simplify the process of loading the update log file, prepare a batch file as described in Chapter 45, "Automating the Production Environment."

When SSAUDIT or SSAUDITR is specified, Essbase logs spreadsheet update transactions chronologically. Essbase uses two files:

- *`dbname`*`.ATX` stores spreadsheet update transactions as a unit that can be used as the input source for data loads.
- *`dbname`*`.ALG` contains historical information for each transaction, such as username, date, and timestamp, and the number of transaction rows from the the `.ATX` file.

Both files are stored on the server.

The spreadsheet update log can get quite large; even if you are using SSAUDITR, Essbase clears the log only after you back up data. If spreadsheet update activities are frequent in your application, you may want to manually delete the log file periodically.

When a database is started after a shutdown, if spreadsheet logging is enabled, Essbase writes the following message to the database log file:

```
Starting Spreadsheet Log
volumename\application_directory\application_name\
database_name\database_name.atx For Database database_name
```

For example:

```
Starting Spreadsheet Log \ESSBASE\app\app1\sample\sample.atx
for database sample
```

To ensure successful spreadsheet update logging, stop and restart your application after either of the following:

- Any operation that causes a restructure. See Chapter 49, “Optimizing Database Restructuring.”
- Running any of the following ESSCMD commands:

  CREATEAPP
  CREATEDB
  COPYDB
  RENAMEDB

Essbase ensures that if you enable spreadsheet logging, updates cannot take place without being logged. If Essbase cannot write to the update log for any reason, Essbase stops the transaction and issues an error message.

SSAUDIT and SSAUDITR are available only from the `ESSBASE.CFG` file.

Hybrid Analysis

Hybrid Analysis offers a means of integrating a relational database with a multidimensional database so that lower level members and their associated data remain in the relational database while upper level members and their associated data reside in the Essbase database. This presents additional issues regarding data consistency and integrity.

For information on ensuring that your data is correct in all locations, see "Managing Data Consistency" on page 1506 in Appendix D, "Accessing Relational Data with Hybrid Analysis."

Part

IX

Maintaining and Automating Hyperion Essbase

This part describes how to maintain the OLAP Server and existing applications and databases. This part includes information about running Essbase; managing applications and databases; monitoring server, application, and database performance; backing up data; and using MaxL and ESSCMD to automate the production environment. Part VIII contains the following chapters:

- Chapter 41, "Running Essbase Servers, Applications, and Databases," describes the Hyperion Essbase Agent and describes the different ways to start and stop the Hyperion Essbase server, applications, and databases.
- Chapter 42, "Managing Applications and Databases," introduces you to the kinds of files that Hyperion Essbase uses, describes operations you can perform on Hyperion Essbase applications and databases, and discusses some of the issues you face when working with Hyperion Essbase files on different platforms.
- Chapter 43, "Using Essbase Logs," describes each Essbase log, including its contents, and all the actions you can perform on each log.
- Chapter 44, "Backing Up and Restoring Data," describes how to back up databases and how to restore data from backups.
- Chapter 45, "Automating the Production Environment," describes how to use MaxL and ESSCMD to automate the production environment.

Chapter

41

Running Essbase Servers, Applications, and Databases

This chapter describes the various methods used to run the Essbase OLAP Server and its associated applications and databases. It defines the role of the Agent, which is the central organizing subsystem for the server, and describes different methods of starting and stopping the server, applications, and databases.

This chapter includes the following sections:

Essbase Executable Files

Seven main executable files contain the server and client software for Essbase:

Table 48: Main Essbase Executable Files

Executable File	Description
`essbase.exe`	Essbase Agent process. For more information, see "Understanding the Agent" on page 1157.
`esssvr.exe`	Essbase application server process. For more information, see "Starting and Stopping Applications" on page 1166.
`essadmin.exe`	Essbase Application Manager client application. For more information, see the *Essbase Application Manager Online Help*.
`essmsh.exe`	Essbase MaxL Shell. For more information, see the *Technical Reference* in the `docs` directory.
`esscmd.exe`	Essbase ESSCMD command-line client interface. For more information, see "Automating the Production Environment" on page 1257.
`adminsvr.exe`	Essbase Administration Services executable. For more information, see the *Essbase Administration Services Online Help*.
`admincon.exe`	Essbase Administration Services Console application. For more information, see the *Essbase Administration Services Online Help*.

The first five files are stored in the `\essbase\bin` directory (`/essbase/bin` on UNIX). The last two files are stored in the *`ARBORPATH`*`\eas\bin` directory (*`ARBORPATH`*`/eas/bin` on UNIX). On UNIX systems, these files do not have the `.exe` extension.

You can launch any of the executable files by typing its name at the operating-system command prompt. On Windows, you can also launch a program by double-clicking the file in Explorer, or entering its name in the dialog box that comes up when you select Run from the Start menu.

Understanding the Agent

The Essbase Agent is the process that starts and stops all applications and acts as the traffic coordinator for the OLAP Server. Use this section to understand more about the Agent:

- "Flow of Communications Events" on page 1158
- "Multithreading" on page 1159
- "Displaying a List of Agent Commands" on page 1160

The agent log is called the OLAP Server log. For information on viewing the OLAP Server log, see "Viewing the Server and Application Logs" on page 1225.

When you start the Essbase server in the foreground, the Agent becomes active in an operating-system window. You should see information like this appearing in the window:

Figure 511: Sample Agent Output to Operating System Window

```
Unlimited login system
Hyperion Essbase OLAP Server - X.X
Copyright 1991-2000 Hyperion Solutions Corporation.
US Patent Number 5,359,724
All Rights Reserved.
Serial number:  XXXXXXXXXXXX-XXXXXXXXXXX
Registered to:  admin
                hyperion
Please type the system password: ********
Startup sequence completed
Security is enabled
Logins are enabled
Essbase Default Storage type is Multidimensional
[Wed Dec 06 14:00:20 2000]Local/ESSBASE0///Info(1051051)
Hyperion Essbase OLAP Server - started
Waiting for Client Requests...
```

In the Agent, you can enter administrative commands and monitor the behavior of Essbase. The Agent is accessible only from the server console on the server computer. The server console is the primary terminal, or monitor, connected to the server computer. The server computer is the computer on which Essbase runs.

The Agent executable file is `essbase.exe` (`ESSBASE` on UNIX systems), which is stored in the `\`*`ARBORPATH`*`\bin` directory. Launching this executable file starts Essbase. You can start Essbase in the foreground or as a background process. If you start Essbase in the background, the terminal becomes free for other input, and the Agent activities are not visible in the terminal. For more information, see “Starting Essbase as a Background Process” on page 1164.

Flow of Communications Events

When a user logs in to Essbase, these events occur:

1. The Essbase client logs in, using a predefined address:
    - For TCP/IP connections, the address is port number `1423` unless it has been changed using the configuration setting AGENTPORT.
    - For Named Pipes connections, the address is `\\pipe\essbase0`.
2. The user selects an application/database combination.
3. The Agent compares the requested application with applications currently running. If the specified application is already running, the Agent does not need to do anything. If the requested application is not already running, the Agent initiates startup, and the following events occur:
    a. The Agent assigns a dynamic port for the application server or creates a dynamic name for a named pipe.
    b. The application server returns the port number for the application to the Agent and to the client. Port number information is stored at run time in Essbase API.
4. The Agent sets the application server as active with the current user and security information. When the client later sends a data request to the application server, the security information captured by the API is embedded in the request.
5. The Agent sends the client query to the application server (`ESSSVR`).

6. For every request that the client makes, the application server causes the following events to occur:

   a. Connect

   b. Send request

   c. Get response

   d. Send data

   e. Receive data

   f. Disconnect

Multithreading

`essbase.exe` and `esssvr.exe` (`ESSBASE` and `ESSSVR` on UNIX) are multithreaded applications. By default, the number of threads is based on the number of licensed ports that are purchased, as shown in Table 49. The number of ports represents the number of concurrent connections that Essbase supports. Essbase provides one reserve port for the system administrator. The system administrator uses the reserve port to log out one or more users when all other ports are in use.

Table 49: Licensed Ports and Multithreading

Number of Licensed Ports	Default Number of Threads
1–5 ports	5
6–10 ports	10
11 or more ports	20

You can set the number of threads for the Agent or the server in the `essbase.cfg` configuration file. For complete details on setting the number of threads for the Agent and the server, refer to the AGENTTHREADS and AGTSVRCONNECTIONS settings for the Agent, and SERVERTHREADS settings in the *Technical Reference* in the `docs` directory.

Displaying a List of Agent Commands

➤ To display a list of all available Agent commands, press Enter in the operating system window where you started Essbase in the foreground (Figure 511 on page 1157).

Use this table to understand the function of each command displayed, and their MaxL or ESSCMD equivalents:

Table 50: Agent Commands and Equivalent MaxL and ESSCMD Commands

Agent Command	**Function**	**Corresponding MaxL or ESSCMD**
START *appname*	Starts the specified application.	MaxL: **alter system load application** *appname*; ESSCMD: LOADAPP
STOP *appname*	Stops the specified application.	MaxL: **alter system unload application** *appname*; ESSCMD:UNLOADAPP
USERS	Displays a list of all users that are connected to the server. The following information is displayed: • The names of all users connected to the server • The total number of ports installed • The total number of existing connections • The application to which each user is connected • The database to which each user is connected	MaxL: **display user;** (lists all users and shows which users are logged in) ESSCMD: LISTUSERS (lists all users)
PORTS	Displays the number of ports that are installed on the server and the number of ports that are in use.	MaxL: **display system;** (to display available unused ports)

Table 50: Agent Commands and Equivalent MaxL and ESSCMD Commands (Continued)

Agent Command	Function	Corresponding MaxL or ESSCMD
LOGOUTUSER *user*	Disconnects a user from the server and frees a port. This command requires the Essbase system password.	MaxL: **alter system logout session by user** *username*; ESSCMD: LOGOUTUSER
PASSWORD	Changes the system password that is required to start the OLAP Server. This command requires the Essbase system password.	MaxL: **alter user** *system supervisor* **set password** *password*; ESSCMD: SETPASSWORD
DUMP *filename*	Dumps information from the Essbase security system to a specified file in ASCII format. If you do not supply a path with the file name, the file is saved to the `BIN` directory, for example, `\essbase\bin` (`/essbase/bin` on UNIX). This command requires the Essbase system password.	N/A
VERSION	Displays the server software version number.	MaxL: **display system;** ESSCMD: GETVERSION
HELP	Lists all valid Agent commands and their respective functions. Same as pressing Enter.	N/A
QUIT and EXIT	Shuts down all open applications and quits Essbase.	MaxL: **alter system shutdown;** ESSCMD: SHUTDOWN-SERVER

Starting and Stopping

Use these sections for instructions about starting and stopping Essbase, applications, and databases:

- “Starting Essbase” on page 1162
- “Changing the System Password” on page 1163
- “Stopping Essbase” on page 1164
- “Starting Essbase as a Background Process” on page 1164
- “Starting Essbase with Parallel Login Processing” on page 1166
- “Starting an Application” on page 1166
- “Stopping an Application” on page 1168
- “Stopping an Application Improperly” on page 1170
- “Starting a Database” on page 1171
- “Stopping a Database” on page 1173

Starting Essbase

You can start Essbase from the server console. In order to start Essbase, you must have the role of supervisor.

➤ To start Essbase in the foreground, enter this command at the operating system command line on the computer where OLAP Server is installed:

```
ESSBASE password
```

password is the password to access the OLAP Server.

On Windows, you can also start Essbase in any of these ways:

- Select Essbase OLAP Server from the Hyperion Solutions program group on the Start menu.
- Double-click `essbase.exe`, located in the `bin` directory, and enter the password when prompted.
- From the Start menu, select Run, and type `ESSBASE` *`password`* in the Open field.

This action starts the Agent in the operating-system command console. In the Agent, you can enter commands or monitor Essbase activities (see “Understanding the Agent” on page 1157).

You cannot start Essbase from ESSCMD or MaxL.

Changing the System Password

Using an Agent command, you can change the password that is required to start the OLAP Server.

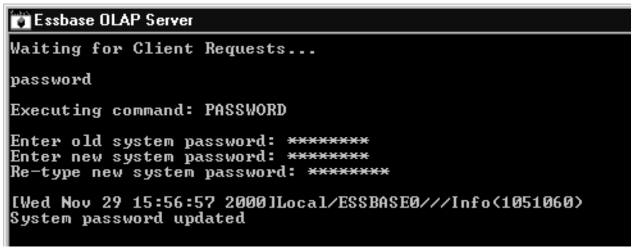
Note: Changing the system password does not change the connection password that is established for the Essbase system supervisor.

➤ To change the Essbase system password:

1. Type **password** at the command prompt of the Agent window.

   Essbase prompts you to enter the old system password, as shown in Figure 512.

   Figure 512: Changing the Essbase System Password

   ```
   Waiting for Client Requests...

   password

   Executing command: PASSWORD

   Enter old system password: ********
   Enter new system password: ********
   Re-type new system password: ********

   [Wed Nov 29 15:56:57 2000]Local/ESSBASE0///Info(1051060)
   System password updated
   ```

2. Type the old (current) system password.
3. Type the new system password.
4. Retype the new system password.

Essbase verifies that the system password has been updated.

For information about changing passwords for users or groups, see Chapter 15, “Managing Security for Users and Applications.”

Stopping Essbase

You can stop Essbase from the server console, or from a MaxL or ESSCMD session running either on the server or on a client connected to the server. In order to stop Essbase, you must have the role of supervisor.

➤ To stop (also “shut down”) Essbase and all running applications, do one of the following:

- Enter quit or exit at the command prompt in the Agent.

  If you stop the Agent by closing the Agent window or by pressing Ctrl + C, the next time you start the database, Essbase rolls back any transactions that were in progress.

  See “Rollback with Committed Access” on page 1137 or “Rollback with Uncommitted Access” on page 1139 for more information about roll backs (undoing all transactions after the last committed transaction).

- Use **alter system shutdown** in MaxL.
- Use the `SHUTDOWNSERVER` command in ESSCMD.

  You must have supervisor privilege to shut down Essbase. See the *Technical Reference* in the `docs` directory for information about MaxL and ESSCMD.

Starting Essbase as a Background Process

Essbase can run as a background process on the server computer.

System administrators might choose to run Essbase as a background process when working in batch mode. Running in batch mode means, for example, that the administrator could start Essbase, launch MaxL or ESSCMD, log in to Essbase, load data, run calculation scripts or reports, stop Essbase, and perform a backup, all from a single UNIX shell script or a Windows NT `.bat` file.

System administrators running Essbase on Windows NT or Windows 2000 might also choose to run Essbase as a background process in order to use Windows settings to give a performance boost to applications running in the foreground.

If you start Essbase in the background, these conditions apply:

- You do not have access to Agent commands.
- You cannot shut down Essbase from the Agent. You must use MaxL or ESSCMD to shut down Essbase.

- You cannot access the application server window to monitor a running application. You must access this information from the application's log (`applicationname.log` in the directory located under `\app` with the same name as the application).
- You cannot monitor server activity using the Agent. You must access this information from the OLAP Server log (`essbase.log` in the `essbase` directory).

To start Essbase in the background on UNIX, or on Windows systems utilizing a UNIX-like shell such as MKS, enter the following command at the command prompt of your operating system:

```
essbase password -b &
```

You can start Essbase without using the ampersand (&) at the end of the command, but if you do not use "&," the command prompt is not returned after the server is started.

Note: The ampersand (&) used in the above context has no effect when used on Windows, unless you are using a UNIX-like shell such as MKS. Essbase starts in the background, but control of the command prompt is not returned. You may need to press the Enter key twice in Windows before the command prompt returns.

On UNIX systems, to find out if Essbase is already running in the background, type this command at the command prompt of your operating system:

```
ps -ef | grep ESS
```

If the server is running in the background, it appears in the process list:

```
essbase password -b &
```

For information about hiding the password from the UNIX process list, see the *Essbase Installation Guide*.

For information about how to run Essbase as a Windows NT Service, or to see if Essbase is already installed as a Windows NT Service, see the *Essbase Installation Guide*.

Starting Essbase with Parallel Login Processing

➤ To speed up parallel login processing with the command-line parameter `quicklogin`, start Essbase using this command:

```
essbase password -quicklogin
```

The parameter `quicklogin` improves performance by caching the security file and writing it to disk at specified time intervals. This parameter is currently unavailable for AIX.

Note: You can use a configuration setting to ensure Essbase always starts with parallel login processing by adding QUICKLOGIN to your server configuration file `essbase.cfg`. For details, see *Technical Reference* in the `docs` directory of your Essbase installation.

Starting and Stopping Applications

When an application is started, Essbase loads the application and all associated databases into memory on the server. All client requests for data, such as data loads, calculations, reports, and spreadsheet lock and sends, are then handled through the application server, the ESSSVR process. The application server is always started by the Agent.

Multiple applications (ESSSVR processes), can run on Essbase concurrently. On Windows, a separate window opens for each ESSSVR process that is running. If an application contains multiple running databases, all databases are managed by the one application server.

When you stop an application, Essbase unloads all information and databases from memory on the server and closes the application server process.

Starting an Application

When you start an application, the following actions can happen:

- Users can connect to the application.
- The application can respond to commands from the Agent.
- Users can change the settings of the application.
- Data and user security are enabled.
- Each database in the application can start.

➤ To start an application from the Agent, enter the following command at the command prompt in the Agent:

```
START appname
```

This action starts the application and, if you are running on Windows, opens the application server window.

➤ To start an application using Application Manager:

1. Connect to the server on which the application resides.
2. Select the application from the Application Desktop window.
3. Select **Application > Start/Stop**.
4. If the application is not already loaded, a confirmation box is displayed.Click **Yes** to load the application.
5. If the application is already loaded, the confirmation box asks if you want to stop the application. Click **No** to prevent Essbase from stopping the application.

You can also start an application by completing any of these actions:

- Creating a new application in Application Manager, which automatically starts the application.
- Starting a database within an application, as discussed under "Starting and Stopping Databases" on page 1171.
- Saving an outline to the server. Opening an outline does *not* start an application.
- Using Essbase Administration Services.
- Using the LOADAPP or SELECT commands in ESSCMD.
- Using **alter system load application** in MaxL.

See the *Technical Reference* in the `docs` directory for information about MaxL and ESSCMD.

See *Essbase Administration Services Online Help* for information about Essbase Administration Services.

You can control how applications are started:

- To configure an application to start automatically when Essbase starts, click the General tab on the Application Properties window in Essbase Administration Services and select Start application when Essbase starts.
- To configure an application to start automatically when Essbase starts, enable the Start When Essbase Starts option in Essbase Application Manager (Application Settings dialog box), or use **alter application enable autostartup** in MaxL.
- To configure an application to start automatically when needed without any explicit commands, enable the Allow Application to Start option in Essbase Application Manager (Application Settings dialog box), or use alter application enable startup in MaxL.

When application **startup** (Allow Application to Start) is enabled, if an application is stopped and a user attempts to retrieve data from any databases within that application, the application starts on the server automatically.

When application **autostartup** (Start When Essbase Starts) is enabled, users may experience better initial performance when they make requests of databases in that application, because the application and databases are already loaded into memory on the server computer.

Stopping an Application

When you stop an application, transactions may be currently running. If you issue the STOP command in the Agent window, the **alter system unload application** statement in MaxL, or an UNLOADAPP command in ESSCMD, the application does not stop if a calculation or data load is in progress. Instead, Essbase displays a message in the Agent console.

If you stop the Agent by closing the Agent window or by pressing `Ctrl + C`, the application stops, and the next time you start the application, Essbase rolls back any transactions that were in progress.

See “Rollback with Committed Access” on page 1137 or “Rollback with Uncommitted Access” on page 1139 for more information about roll backs (undoing all transactions after the last committed transaction).

It is important to stop an application properly by using the Essbase Agent, MaxL, ESSCMD, Essbase Administration Services, or Essbase Application Manager. If the application server is not brought down properly, databases within the application may become corrupt.

Do not use the following methods to stop an application:

- On Windows platforms, performing a Windows operating system End Task.
- On Windows platforms, clicking the Close button in the upper-right corner of the application server window
- On UNIX platforms, killing the ESSSVR process
- On any platform, shutting down the server computer prior to shutting down the Agent

➤ To stop an application from the Agent, enter the following command at the command prompt in the Agent:

```
stop appname
```

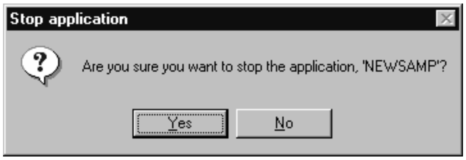
This action stops the application and, if you are running on Windows, closes the application server window.

➤ To stop an application using Application Manager:

1. Connect to the server on which the application resides.
2. Select the application from the Application Desktop window.
3. Select **Application > Start/Stop**.

   A confirmation dialog box is displayed:

   Figure 513: Stop Application Dialog Box



Are you sure you want to stop the application, 'NEWSAMP'?

4. Click **OK** to stop the application. Essbase unloads the application from memory.

Tip: You can stop an application using methods other than Application Manager:

Tool	Instructions	For more information
Administration Services	Enterprise View > select and right-click application > select Stop.	*Essbase Administration Services Online Help*
MaxL	**alter system unload application**	*Technical Reference* in the `docs` directory
ESSCMD	UNLOADAPP	

Stopping an Application Improperly

There are times when stopping the application server process improperly is necessary; for example, if the application server is corrupted and not processing client requests. In this case, stopping the application server by shutting down its corresponding window is the only method of the stopping the application.

The Windows NT Task Manager does not display process IDs for individual Essbase applications: all of the running Essbase applications are displayed as undifferentiated ESSSVR processes. This prevents you from stopping a single application, in the event that the application freezes.

You can find the process ID for individual servers in the `essbase.log` file in your `ESSBASE` directory. When the server starts, a line like the following is displayed in the OLAP Server log:

```
Application [Sample] started with process id [225]
```

If an application freezes, you can stop the application by using the process ID number in conjunction with a kill operating-system utility. Such a utility is provided with Windows NT Resource Kit and with various other toolkits.

On UNIX platforms, you can use the ps output to identify individual applications. If an application freezes, you can stop the application by using this command:

kill -9 <pid>

Starting and Stopping Databases

Starting a database loads the database into memory on the OLAP Server computer. Stopping a database unloads all database information from memory on the server computer.

Starting a Database

When Essbase starts a database and loads it to memory, the entire index cache for that database is allocated in memory automatically. The data cache and data file cache are allocated as blocks are requested from Essbase clients.

When you start an application, Essbase loads the application and its databases into memory on the server computer. When you start a database from an application that is not loaded, the application is loaded along with all its related databases.

➤ To start a database from the Agent, enter the following command at the command prompt in the Agent window:

```
start appname
```

This action starts the application that contains the database you want to start and opens the application server window.

➤ To start a database using Application Manager:

1. Connect to the server on which the database resides.
2. From the Application Desktop window, select the application and then the database.
3. Select **Database > Start/Stop**.

   If the database is not already loaded, a confirmation box is displayed. Click **Yes** to load the database.

   If the database is already loaded, the confirmation box asks if you want to stop the database. Click **No** to prevent Essbase from stopping the database.

You can also start a database by completing any of the following actions in Application Manager:

- Creating a new database, which automatically starts the database.
- Saving an outline to the server. Opening an outline does *not* start a database.
- Clicking the Statistics tab in the Database Information dialog box.

Tip: You can also start a database outside of the Application Manager using any of the following methods.

Tool	Instructions	For more information
Administration Services	Enterprise View > select the name of the database > right-click > click Start.	*Essbase Administration Services Online Help*
MaxL	**alter application load database**	*Technical Reference* in the `docs` directory
ESSCMD	LOADDB or SELECT	

➤ To configure a database to start automatically when its parent application starts,

1. In Application Manager, open the **Database Settings** dialog box.
2. Enable the **Start when Application Starts** option.

Tip: You can also configure a database to start automatically when its parent application starts by using one of the following:

Tool	Instructions	For more information
Administration Services	Database Properties window > General tab > Start database when application starts	*Essbase Administration Services Online Help*
MaxL	**alter database enable autostartup**	*Technical Reference* in the `docs` directory

Stopping a Database

Stopping a database unloads all data from memory and commits any updated data to disk.

If a database is stopped and a user attempts to retrieve data from it, the database starts in the server automatically, without any explicit commands issued.

When you stop a database, transactions may be currently running. If you issue the STOP command in the Agent window, an UNLOADDB command in ESSCMD, or **alter application unload database** in MaxL, the database does not stop if a calculation or data load is in progress. Instead, Essbase displays a message in the Agent window. If you stop the Agent by closing the Agent window or by pressing `Ctrl + C`, the database stops, and the next time you start the database, Essbase rolls back any transactions that were in progress.

See “Rollback with Committed Access” on page 1137 or “Rollback with Uncommitted Access” on page 1139 for more information about roll backs (undoing all transactions after the last committed transaction).

➤ To stop a database from the Agent, enter the following command at the command prompt in the Agent window:

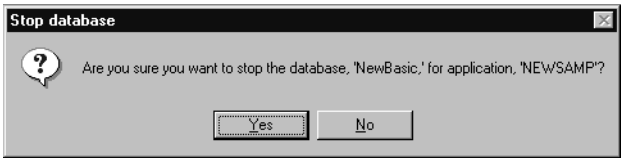
```
stop appname
```

This action stops the application that contains the database you want to stop and closes the application server window.

➤ To stop a database using Application Manager:

1. Connect to the server on which the database resides.
2. From the Application Desktop window, select the application and then the database.
3. Select **Database > Start/Stop**. This confirmation is displayed:

Figure 514: Stop Database Dialog Box



Stop database

Are you sure you want to stop the database, 'NewBasic,' for application, 'NEWSAMP'?

4. Click **Yes** to stop the database. Essbase unloads the database from memory.

You can also stop a database by completing any of the following actions:

Tool	Instructions	For more information
Application Manager	Application Start/Stop command (stops the application that contains the database that you want to stop)	See "Stopping an Application" on page 1168. *Essbase Application Manager Online Help*
Administration Services	Right-click the name of the database > select Stop	*Essbase Administration Services Online Help*
MaxL	**alter application unload database**	*Technical Reference* in the `docs` directory
ESSCMD	UNLOADDB	

Viewing a List of Users and Available Ports

The Agent enables you to view a list of all users that are connected to the server at any given time. Additionally, you can view the total number of ports available, as well as the number of existing connections.

To view a list of all users connected to the server, type the command USERS at the command prompt of the server console. The server console displays the following information:

- The names of all users connected to the server
- The total number of ports available
- The total number of existing connections
- The application to which each user is connected
- The database to which each user is connected

To view the number of ports installed on the server, as well as the number of ports in use, type PORTS at the command prompt of the server console.

Tip: You can also use **display user** in MaxL or the LISTUSERS command in ESSCMD to view a list of users on the system. These do not return information about the available ports and connections or the application or database to which users are connected. For more information, see the *Technical Reference* in the `docs` directory.

You can also use Essbase Administration Services and select the Statistics tab from the OLAP Server Properties window to display the number of ports in use on the server and the number of ports available for use on the server. For more information, see *Essbase Administration Services Online Help*.

Controlling Query Size and Duration

Users may unintentionally request information that is so large or so complex to retrieve that the query will slow performance or fail to complete properly. To prevent this, you may use two configuration settings to control query size or duration:

- QRYGOVEXECTIME limits the length of time the server allows a query to run before terminating the query.
- QRYGOVEXECBLK limits the number of blocks a query can access before terminating the query.

These two configuration settings are also referred to as *query governors*.

You can apply these settings to all the applications and databases on an OLAP Server, to all the databases on a single application, or to a single database.

For details about how to use these configuration settings, see the *Technical Reference* in the `docs` directory, in the “essbase.cfg Settings” topic.

Specifying Non-Default Port Values

If you wish to change the default port values used by the Agent, you must set one or more of these configuration settings:

- AGENTPORT specifies the port that the Agent uses.
- SERVERPORTBEGIN specifies the first port number the Essbase Agent on a single computer will try to use for its first server process.

- SERVERPORTEND specifies the highest value Essbase will try to use for a port when it tries to start a server process. If the value is unavailable, the server process will fail.
- PORTINC specifies the value of the increment in between port numbers used by the Agent.

You may wish to change the default for many reasons. Here are two examples:

- The default value is inappropriate for your site because it specifies a port number already in use. To change default values for one installation on a single computer, see "Changing Port Default Values" on page 1176.
- You may wish to install a second Agent on a single computer to facilitate testing. Use the configuration settings to assign the second Agent to a different port than the first. For more detailed instructions, see "Installing Additional Agents: Windows" on page 1177, or "Installing Additional Agents: UNIX" on page 1178.

CAUTION: Do not use more than one Agent per computer in a production environment. Multiple Agents are only appropriate for test environments.

Changing Port Default Values

If you simply need to change one or more of the default values associated with Agent and server ports, review the *Technical Reference* in the `docs` directory for instructions about the correct configuration setting to change:

- AGENTPORT specifies the port that the Agent uses.
- SERVERPORTBEGIN specifies the first port number the Essbase Agent on a single computer will try to use for its first server process.
- SERVERPORTEND specifies the highest value Essbase will try to use for a port when it tries to start a server process. If the value is unavailable, the server process will fail.
- PORTINC specifies the value of the increment in between port numbers used by the Agent.

Installing Additional Agents: Windows

➤ To install and configure an additional Agent on a single Windows computer:

1. On a computer where you have already installed Essbase according to the instructions in the *Essbase Installation Guide*, install the server portion in a different directory from the first Essbase installation. You need not install any of the clients, such as MaxL or Administration Services.
2. In the `/bin` directory of the new installation, create or modify the server configuration file `essbase.cfg` so that it contains these settings:

   AGENTPORT: The port that this second Agent will use to connect.

   SERVERPORTBEGIN: The first port that the first server process will try to use to connect.

   SERVERPORTEND: The highest value for a port number that a server process can use to connect.

   PORTINC: The increment between ports. For example, if PORTINC is assigned a value of 5, Essbase will look for ports 32700, 32705, and so on up to the value of SERVERPORTEND.

   Use the instructions in the *Technical Reference* in the `docs` directory to select values for these settings.
3. Repeat Step 2 for the client configuration file `essbase.cfg`. The client configuration file is located at `/ARBORPATH/bin`.
4. Create a batch script (`.bat`) that performs these tasks:
   - Sets ARBORPATH to the correct value for this installation
   - Adds this path to the PATH variable:

     ```
     %arborthapth%\bin;@path%
     ```
   - Starts the Essbase Agent
5. Use a client to connect to the new installation.

Note: This is a separate installation of Essbase. It shares none of the security or objects of any other Essbase installation on the same computer.

Installing Additional Agents: UNIX

➤ To install and configure a second agent on a single UNIX computer:

1. On the computer where you plan to use two Agents, create a new operating system login ID to run the new Agent. For instructions, refer to your operating system documentation.
2. On a computer where you have already installed Essbase according to the instructions in the *Essbase Installation Guide*, install the server portion in a different directory from the first Essbase installation. You need not install any of the clients, such as MaxL or Administration Services.
3. Assign the correct privileges so the new login ID can be used run the Agent. For instructions, see the *Essbase Installation Guide*.
4. Create the necessary path and library statements in the profile for the new login ID. You may want to copy an existing login ID profile for a login ID that runs the existing Essbase installation, and change the values for the new installation.
5. Perform Step 2 from "Installing Additional Agents: Windows" on page 1177.
6. Repeat previous step for the client configuration file `essbase.cfg`.
7. Start the new Essbase Agent using the new login ID.

Increasing Agent Connections to Server

You can use the configuration settings AGENTTHREADS and AGTSVRCONNECTIONS to control the maximum number of threads that the Agent creates to perform initial connection to the OLAP Server.

Increasing the maximum number of possible threads between the server and agent allows more than one user to log in and connect to an application and database at a time.

Note: All requests for information after initial connection and before disconnection are handled by a different set of server threads, whose maximum number is controlled by the SERVERTHREADS configuration parameter.

➤ To increase the maximum possible number of agent threads:

1. Create or edit the server configuration file `essbase.cfg` to include these settings:

   - AGENTTHREADS *maximum_number_of_threads*

     Maximum number of threads that can be spawned by the Agent. These threads are agent threads, and are used by both the server threads controlled by AGTSVRCONNECTIONS for initial connection and disconnection and for other Agent tasks such as answering a request to list users, list applications, or create new users, for example.

   - AGTSVRCONNECTIONS *maximum_number_of_threads*

     Maximum number of threads that the server can spawn to communicate with Agent for initial connection and disconnection. These threads are unrelated to the server threads whose maximum number is controlled by SERVERTHREADS.

   Keep the value of *maximum_number_of_threads* for AGTSVRCONNECTIONS equal to or less than that value for AGENTTHREADS to avoid wasting resources. Each connection requires one thread each from server and Agent, so there is no need for higher values for AGTSVRCONNECTIONS. The default value for each setting is 5, the minimum is 1, and the maximum is 500.

2. Save the configuration file `essbase.cfg`.

3. Stop and restart the server to initialize your changes.

For more information, see the *Technical Reference* in the `docs` directory.

Limiting the Number of User Sessions

You can limit the maximum number of user sessions allowed to connect to the OLAP Server at any one time, using the configuration parameter MAXLOGIN. This number includes multiple instances of the same user.

For example, the same user with five open Excel worksheets connected to the same OLAP Server use one port, but five sessions.

You may wish to adjust the value of MAXLOGIN to match computer resources, or to more closely manage concurrent ports and user sessions. A concurrent port is used for each unique combination of client machine, OLAP Server and login name.

To limit the number of simultaneous user sessions using the configuration setting MAXLOGIN, create or edit the server configuration file `essbase.cfg` to contain this setting:

```
MAXLOGIN maximum_number_of_user_sessions
```

The *maximum_number_of_user_sessions* minimum is 1, maximum is 1048575, and the default value is 1000.

User sessions use the threads whose maximum is controlled by the configuration setting SERVERTHREADS, and are not related to the threads whose maximum is controlled by AGENTTHREADS and AGTSVRCONNECTIONS.

For more information, see the *Technical Reference* in the `docs` directory.

Chapter 42

Managing Applications and Databases

This chapter describes the files that are associated with Essbase and describes operations that you can perform to maintain Essbase applications, databases, and database objects.

This chapter contains the following sections:

About Applications and Databases

An application is a management structure that contains one or more Essbase databases and related files. Essbase applications and databases usually reside on the OLAP Server. The server computer can store multiple applications. Applications and databases created on client computers are used only to store database objects, such as outlines and calculation scripts. You cannot load data or calculate data on a client computer.

An Essbase database is a data repository that contains a multidimensional data storage array. A multidimensional database supports multiple views of data so that users can analyze the data and make meaningful business decisions.

Files that are related to Essbase databases are called *objects*. Database objects perform actions against one or more Essbase databases, such as defining calculations or reporting against data. By default, objects are stored in their associated database folder on the server. They can also be saved to a client computer or to other available network directories. For more information about how Essbase stores files, see "How Essbase Files Are Stored" on page 1182.

In Essbase, the common types of database objects include the following:

- A database outline (a storage structure definition)
- Data sources
- Rules for loading data and building dimensions dynamically ("rules files")
- Scripts that define how to calculate data ("calculation scripts")
- Scripts that generate reports on data ("report scripts")
- Security definitions
- Security filters
- Linked reporting objects
- Partition definitions

Some of these objects are optional, such as calculation scripts, filters, and linked reporting objects.

For a complete description of each database object, see "Database Objects" on page 142.

How Essbase Files Are Stored

In order to manage applications and databases, you need to know how Essbase stores server, application, database, and database object files. In particular, there are a few key directories that you should know about.

These directories are created under the root directory of the Essbase installation (the directory path named by the value of ARBORPATH):

- An APP directory stores Essbase application files as they are created. Each application is stored in a subdirectory under the APP directory (for example, \essbase\app\appname).

  Each database in an application is stored in a subdirectory under the application subdirectory (for example, \ESSBASE\app\*appname**dbname*). Database objects, such as outlines, calculation scripts, report scripts, rules files, and data sources, are typically stored on the server in the *appname* or *dbname* directory or on the client computer.

  See Table 52 for a list of application and database files.

- The bin directory contains the Essbase software. See Table 51 for a list of files stored in this directory.
- The client directory contains client-based applications and databases. This folder is created during installation of Application Manager, Spreadsheet Add-in, or the Runtime Client.
- The docs directory is created if you choose to install online HTML or PDF documentation. This directory contains numerous subdirectories and files.
- The locale directory contains the character-set files necessary for all languages that are supported by Essbase, including English.
- If you installed Essbase SQL Interface, ODBC-driver documentation is located in the odbcdocs directory in .PDF format.

Note: On Windows platforms, these directory names may appear with different case.

For more information about all directories created on the server and for information about platform differences, see the *Essbase Installation Guide*.

Server Software File Types

This table lists the types of OLAP Server files that are stored in the `\essbase\bin` directory:

Table 51: Essbase File Types in the \essbase\bin Directory

File Extension	Description
`12a`	Spreadsheet Add-in for Lotus 1-2-3
`bak`	Backup of security file
`bnd`	Microsoft ODBC file for SQL Interface installation using a DB2 database
`cfg`	Essbase configuration file
`cnt`	Online help contents file
`cpl`	Microsoft ODBC driver for Windows platforms
`dll`	Microsoft Windows Dynamic Link Library
`eqd`	Essbase Query Designer files
`exe`	Executable file
`hlp`	Online help file
`lck`	Essbase lock file
`lic`	License information file for ODBC
`pl`	Sample Perl script
`pm`	Perl Module
`mdb`	Message database file
`sec`	Security file
`sl`	HP-UX shared library file
`so`	Solaris shared library file
`xll`	Spreadsheet Add-in for Microsoft Excel

Application and Database File Types

This table lists the file types that Essbase uses to store applications, databases, and their related objects.

Table 52: Essbase File Types for Applications and Databases

File Extension	Description
`123`	Lotus 1-2-3 spreadsheet file
`alg`	Spreadsheet audit historical information
`apb`	Backup of application file
`app`	Application file, defining the name and location of the application and other application settings
`arc`	Archive file
`atx`	Spreadsheet audit transaction
`chg`	Outline synchronization change file
`csc`	Essbase calculation script
`db`	Database file, defining the name, location, and other database settings
`dbb`	Backup of database file
`dbf`	dBASE data file
`ddb`	Partitioning definition file
`ddm`	Temporary partitioning file
`ddn`	Temporary partitioning file
`esm`	Essbase Kernel file that manages pointers to data blocks, and contains control information that is used for database recovery
`esr`	Temporary database root file
`esn`	Temporary Essbase Kernel file
`ind`	Essbase index file
`inn`	Temporary Essbase index file
`log`	Server or application log file

Table 52: Essbase File Types for Applications and Databases (Continued)

File Extension	Description
lro	Linked reporting object file that is linked to a data cell
lst	Cascade table of contents or list of files to back up
mdx	dBASE multiple index file
ocl	Database change log
ocn	Incremental restructuring file
oco	Incremental restructuring file
olb	Backup of outline change log
olg	Outline change log
otl	Essbase outline file
otm	Temporary Essbase outline file
otn	Temporary Essbase outline file
oto	Incremental restructuring file
pag	Essbase database data (page) file
pan	Temporary Essbase database data (page) file
rep	Essbase report script
rul	Essbase data load rules file
scr	Essbase ESSCMD script
sel	Saved member select file
tct	Essbase database transaction control file that manages all commits of data and follows and maintains all transactions
tcu	Temporary database transaction control file
txt	ASCII text file, such as a data file to load or a text document to link as a linked reporting object
xcp	Exception error log file
xls	Microsoft Excel spreadsheet file

API File Types

This table lists the types of Essbase files that are stored in the `\essbase\api` sub-directories:

Table 53: Essbase File Types in the `\essbase\api` Directory

File Extension	Description
`a`	UNIX static library file
`bas`	Microsoft Visual Basic program source file, containing header definitions for the Essbase API
`h`	C or C++ header file, containing header definitions for the Essbase API
`lib`	C or C++ program library
`np`	Named Pipes network library
`tcp`	TCP/IP network library
`w95`	Windows 95 Windows Socket network library

Managing Applications, Databases, and Database Objects

This section explains how to perform the following operations on applications, databases, and database objects:

- "Using the File System to Manage Applications and Databases" on page 1188
- "Monitoring Applications" on page 1189
- "Managing Applications" on page 1189

For a description of Essbase applications, databases, and database objects, see "About Applications and Databases" on page 142.

Using the File System to Manage Applications and Databases

You should not use your platform's file system to copy, move, rename, or delete applications and databases. When an application or database is altered through the file system, the Essbase security file is unable to recognize the changes. This situation creates a mismatch between what actually exists on the hard drive and what exists according to Essbase.

CAUTION: Do not move, copy, modify, or delete any of these files: `ess`*`n`*`.ind`, `ess`*`n`*`.pag`, *`dbname`*`.ind`, *`dbname`*`.esm`, *`dbname`*`.tct`. Doing so may result in data corruption.

The only time the file system should be used to manage applications and databases is during the backup process, where the entire directory for an application or database is copied and stored elsewhere. For information about backups, see Chapter 44, "Backing Up and Restoring Data."

Certain application and database files can be successfully managed through the file system:

- Rules files for dimension builds and data loads (`.rul`)
- Data load files
- Calc scripts (`.csc`)
- Report scripts (`.rep`)

To copy or move an outline file (`.otl`), you must use Essbase Application Manager to open the outline file and save it to the new location. For more information, see "Copying Outlines" on page 170. For information about copying outlines on a UNIX system, see "Transferring Compatible Files" on page 1205.

Monitoring Applications

Each application that is loaded is an open task or process in the operating system. On Windows platforms, the application is displayed in an Essbase server window. On UNIX platforms, the application server is a child process of `ESSBASE`. When the application starts, `ESSBASE` starts the `esssvr` process.

The application server records its activities (such as writing data to an application log file, *`appname`*`.LOG`) in the *appname* directory below the Essbase installation directory, for example, the `\essbase\app\`*`appname`* directory. You can open and view this text file when you need to troubleshoot problems or view application activity. You can also use Administration Services Log Analyzer to view and analyze logs.

On Windows platforms you can also view application activities as they occur in the Essbase server window. On UNIX, you can view application activities with the `tail -f` *`logfile`* command.

On Windows platforms, when an application starts, a new icon is displayed in the taskbar. You can double-click the icon to view the server window.

Managing Applications

This section describes how to use Application Manager, MaxL, or ESSCMD to copy, rename, and delete applications:

- "Copying an Application" on page 1190
- "Renaming an Application" on page 1191
- "Deleting an Application" on page 1192

CAUTION: The only time the file system should be used to manage applications is during the backup process, where the entire directory for an application or database is copied and stored elsewhere. For information about backups, see Chapter 44, "Backing Up and Restoring Data."

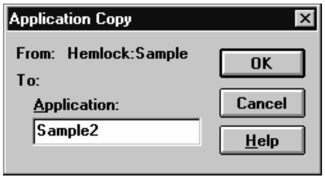
Copying an Application

When you copy an application, Essbase copies all files associated with the application to the destination application. Before copying, make sure that you have enough disk space to contain a full copy of the application, databases, and related files.

➤ To copy an application using Application Manager:

1. Start the OLAP Server.
2. Start the Application Manager.
3. From Application Manager, connect to the OLAP Server on which the application resides, using the Server > Connect menu command.
4. Select an application from the Applications list box.
5. Select Application > Copy.

   Essbase displays the Application Copy dialog box:

   From: Hemlock:Sample
   To:
   Application: Sample2

6. In the Application text box, enter the name for the copy of the application.
7. Click OK. Essbase creates a new application and a new database directory structure on the server under the `\essbase\app` directory.

Tip: You can copy an application using methods other than Application Manager:

Tool	Instructions	For more information
Administration Services	Use Copy Application dialog box	*Essbase Administration Services Online Help*
MaxL	Use **create application as** statement	*Technical Reference* in the `docs` directory
ESSCMD	Use COPYAPP command	

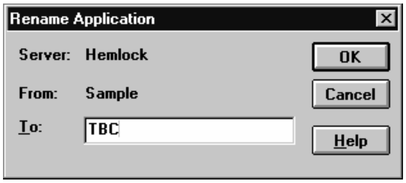
Renaming an Application

When you rename an application, the application and its associated directory (`essbase\app\`*`appname`*) are renamed. All objects within the application (for example, databases or calculation scripts) with the same name as the application are not renamed.

➤ To rename an application using EssbaseApplication Manager:

1. Start the OLAP Server.
2. Start the Application Manager.
3. From Application Manager, connect to the OLAP Server on which the application resides, using the Server > Connect menu command.
4. Select an application from the Applications list box.
5. Select Application > Rename.

   Essbase displays the Rename Application dialog box:



Rename Application

Server: Hemlock

From: Sample

To: TBC

6. In the To text box, enter the new name for the application. The name can be up to 8 alphanumeric characters.
7. Click OK. Essbase renames the application and directory name under the `\essbase\app` directory.

Tip: You can rename an application using methods other than Application Manager:

Tool	Instructions	For more information
Administration Services	Rename Application dialog box	*Essbase Administration Services Online Help*
MaxL	**alter application**	*Technical Reference* in the `docs` directory
ESSCMD	RENAMEAPP	

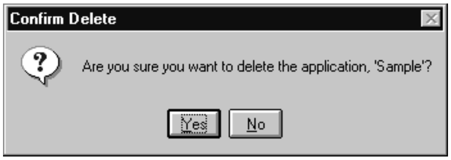
Deleting an Application

When you delete an application, all objects within the application are also deleted. The `\essbase\app\`*`appname`* directory and all files located in the directory are deleted.

➤ To delete an application using Application Manager:

1. Start the OLAP Server.
2. Start the Application Manager.
3. From Application Manager, connect to the OLAP Server on which the application resides, using the Server > Connect menu command.
4. Select an application from the Applications list box.
5. Select Application > Delete.

   Essbase displays the Confirm Delete dialog box:



Confirm Delete

Are you sure you want to delete the application, 'Sample'?

6. Click Yes or No.

Tip: You can delete an application using methods other than Application Manager:

Tool	Instructions	For more information
Administration Services	Enterprise View > select application > right-click > select Delete Application.	*Essbase Administration Services Online Help*
MaxL	**drop application**	*Technical Reference* in the `docs` directory
ESSCMD	DELETEAPP	

Managing Databases

This section describes using Application Manager and ESSCMD to copy, rename, and delete databases:

- “Copying a Database” on page 1193
- “Renaming a Database” on page 1195
- “Deleting a Database” on page 1196

CAUTION: The only time the file system should be used to manage databases is during the backup process, where the entire directory for an application or database is copied and stored elsewhere. For information about backups, see Chapter 44, “Backing Up and Restoring Data.”

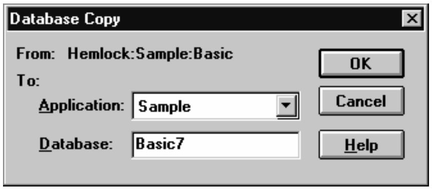
Copying a Database

When you copy a database, all files associated with the database are also copied to the destination application. Before copying, make sure you have enough disk space to contain a full copy of the database and its related files.

➤ To copy a database using Application Manager:

1. Start the OLAP Server.
2. Start the Application Manager.
3. From Application Manager, connect to the OLAP Server on which the application resides, using the Server > Connect menu command.
4. Select the application on which the database resides, from the Applications list box.
5. Select the database that you want to copy from the Databases list box.
6. Select Database > Copy.

   Essbase displays the Database Copy dialog box:



Database Copy

From: Hemlock:Sample:Basic

To:

Application: Sample

Database: Basic7

7. Do either of these tasks:
   - To copy the database to another database within the same application, in the Database text box type a new database name.
   - To copy the database to another database in a different application (the other application must already exist), from the Application list box, select the application name. Then type the name of the new database.
8. Click OK.

Tip: You can copy a database using methods other than Application Manager:

Tool	Instructions	For more information
Administration Services	Copy Database dialog box	*Essbase Administration Services Online Help*
MaxL	**create database as**	*Technical Reference* in the `docs` directory
ESSCMD	COPYDB	

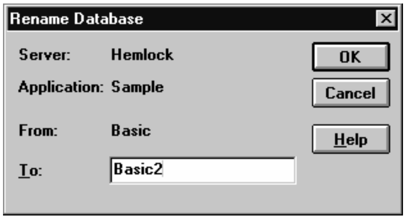
Renaming a Database

When you rename a database, the database and its associated directory (`essbase\app\`*`appname`*`\`*`dbname`*), and the outline file (`.otl`) are renamed. All other objects in the database (for example, calculation scripts) with the same name as the database are not renamed.

➤ To rename a database using Essbase Application Manager:

1. Start the OLAP Server.
2. Start the Application Manager.
3. From Application Manager, connect to the OLAP Server on which the application resides, using the Server > Connect menu command.
4. Select the application on which the database resides, from the Applications list box.
5. Select the database that you want to rename.
6. Select Database > Rename.

   Essbase displays the Rename Database dialog box:



Rename Database

Server: Hemlock

Application: Sample

From: Basic

To: Basic2

7. In the To text box, enter the new name for the database.
8. Click OK. Essbase renames the database and the directory under the `\essbase\app\`*`appname`* directory.

Tip: You can rename an database using methods other than Application Manager:

Tool	Instructions	For more information
Administration Services	Rename Database dialog box	*Essbase Administration Services Online Help*
MaxL	**alter database**	*Technical Reference* in the `docs` directory
ESSCMD	RENAMEDB	

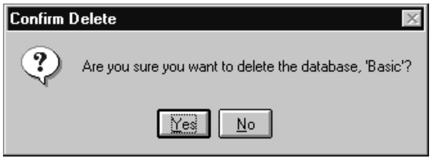
Deleting a Database

When you delete a database, all objects within the database are also deleted. The `\essbase\app\`*`appname`*`\`*`dbname`* directory and all files located in the directory are deleted.

➤ To delete a database using Application Manager:

1. Start the OLAP Server.
2. Start the Application Manager.
3. From Application Manager, connect to the OLAP Server on which the application resides, using the Server > Connect menu command.
4. Select the application on which the database resides, from the Applications list box.
5. Select the database that you want to delete.
6. Select Database > Delete.

   Essbase displays the Confirm Delete dialog box:

   Are you sure you want to delete the database, 'Basic'?

7. Click Yes or No.

Tip: You can delete a database using methods other than Application Manager:

Tool	Instructions	For more information
Administration Services	Enterprise View > select database > right-click > select Delete Database.	*Essbase Administration Services Online Help*
MaxL	Use the **drop database** statement	*Technical Reference* in the `docs` directory
ESSCMD	Use the DELETEDB command	

Managing Objects

This section describes using Application Manager to copy, rename, and delete objects. Objects related to databases or applications include outlines, calculation scripts, report scripts, rules files, and data sources:

- "Copying Objects" on page 1198
- "Renaming Objects" on page 1199
- "Deleting Objects" on page 1199
- "Locking and Unlocking Objects" on page 1200

You can also manage most application and database objects using MaxL or ESSCMD. For more information, see the *Technical Reference* in the `docs` directory.

You can also use Administration Services to manage most application and database objects. For more information, see *Essbase Administration Services Online Help*.

For more information about Essbase database objects, see "Database Objects" on page 142.

CAUTION: The only time the file system should be used to manage applications is during the backup process, where the entire directory for an application or database is copied and stored elsewhere. For information about backups, see Chapter 44, "Backing Up and Restoring Data."

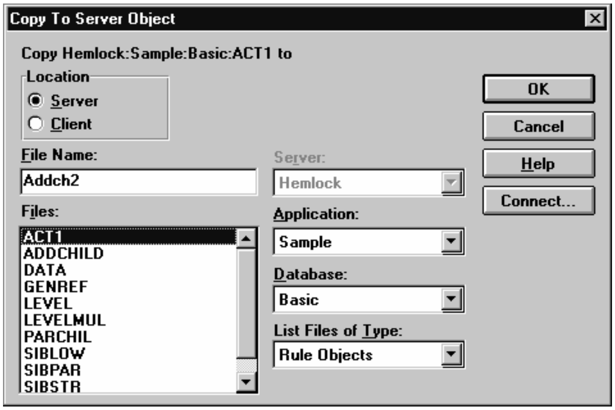
Copying Objects

You can copy any database object, except an outline, to another application, database, server, or client location. For information about copying outlines, see "Copying Outlines" on page 170.

➤ To copy an object using Application Manager:

1. Start the OLAP Server.
2. Start the Application Manager.
3. From Application Manager, connect to the OLAP Server on which the application resides, using the Server > Connect menu command.
4. From the Applications list box, select the application on which the database containing the object resides.
5. In the Application Desktop window, click the button corresponding to the object file type (Calc Scripts, Reports, or Data Load Rules), and then select the object from the list box.
6. Select File > Copy.

   Essbase displays the Copy To Server Object dialog box:



Copy To Server Object

Copy Hemlock:Sample:Basic:ACT1 to

Location
● Server
○ Client

File Name: Addch2

Server: Hemlock

Application: Sample

Database: Basic

List Files of Type: Rule Objects

Files:
ACT1
ADDCHILD
DATA
GENREF
LEVEL
LEVELMUL
PARCHIL
SIBLOW
SIBPAR
SIBSTR

7. In the File Name text box, enter the name of the object.
8. Select the destination server, application, and database.
9. Click OK.

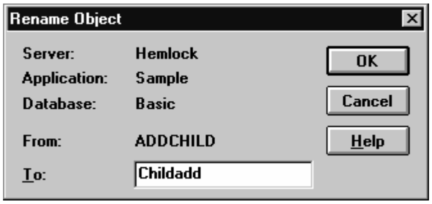
Renaming Objects

You can rename any object, except an outline. An outline always has the same name as the database, so you need to rename the database to rename the outline.

➤ To rename an object using Application Manager:

1. Start the OLAP Server.
2. Start the Application Manager.
3. From Application Manager, connect to the OLAP Server on which the application resides, using the Server > Connect menu command.
4. From the Applications list box, select the application on which the database containing the object resides.
5. In the Application Desktop window, click the button corresponding to the object file type (Calc Scripts, Reports, or Data Load Rules), and then select the object from the list box.
6. Select File > Rename.

   Essbase displays the Rename Object dialog box:



Rename Object

Server: Hemlock

Application: Sample

Database: Basic

From: ADDCHILD

To: Childadd

7. In the To text box, enter the new name for the object.
8. Click OK.

Deleting Objects

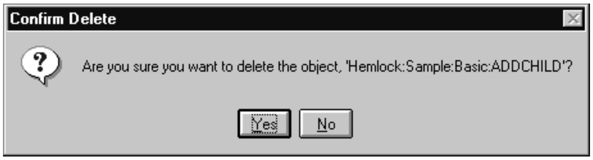
You can delete any object, except an outline. An outlines is a required part of a database, so you need to delete the database to delete the outline.

➤ To delete an object using Application Manager:

1. Start the OLAP Server.
2. Start the Application Manager.

3. From Application Manager, connect to the OLAP Server on which the application resides, using the Server > Connect menu command.
4. From the Applications list box, select the application on which the database containing the object resides.
5. In the Application Desktop window, click the button corresponding to the object file type (Calc Scripts, Reports, or Data Load Rules), and then select the object from the list box.
6. Select File > Delete.

   Essbase displays the Confirm Delete dialog box:



Confirm Delete

Are you sure you want to delete the object, 'Hemlock:Sample:Basic:ADDCHILD'?

7. Click Yes or No.

Locking and Unlocking Objects

Essbase uses a check-out facility for server-based objects to ensure that no more than one user attempts to modify an object at one time. This section describes how to lock and unlock objects.

Note: Locking objects is not the same as locking data blocks. The Essbase Kernel handles locking for data blocks, but not for objects. See Chapter 40, “Ensuring Data Integrity,” for information about locking data blocks.

Locking Objects

By default, whenever a user opens a server-based object, Essbase locks the object. If a second user attempts to open the same object, a message is displayed. The message indicates that the object is locked and tells the user who locked it.

You can avoid the lock error by unchecking “Lock file” in the Application Desktop window before attempting to open an object. This action essentially opens the file in read-only mode, and you cannot save any changes you make.

Unlocking Objects

There are two ways to unlock a previously locked object:

- Save the locked file to the server.
- Select File > Unlock. This action unlocks a file that you previously locked but did not save to the server.

Note: Users can only unlock objects that they have locked themselves. A user with supervisor privileges can unlock any object.

Migrating Applications Using Administration Services

Using Administration Services, you can migrate applications and databases across OLAP Servers, regardless of platform. For example, you can develop and test an application on a Windows server and then migrate it to a production server running UNIX.

When you migrate applications and databases across servers, you can now migrate data, security (including filter associations), and substitution variables. For more information, see the *Essbase Administration Services Online Help*.

Porting Applications Across Platforms

The OLAP Server runs on multiple platforms, including Windows and UNIX. For a list of supported platforms and information on how to install and configure the OLAP Server on each platform, see the *Essbase Installation Guide*.

After you install an application, you may want to port the application to a server that runs a different operating system. This section describes how to port an installed Essbase application to another OLAP Server computer.

Porting Essbase applications across servers involves these steps:

1. Identifying compatible files
2. Checking file names
3. Transferring compatible files
4. Reloading the database

Identifying Compatible Files

If you are porting an Essbase application to a server that uses a different operating system, you need to identify which Essbase files are compatible with the new operating system.

The following file types are compatible between operating systems:

- ASCII text files. The Essbase ASCII text files are calculation scripts (.csc) and report scripts (.rep), and any MaxL or ESSCMD scripts you have developed. Also, data files can be ASCII text files.
- Data load rules files. These files are binary files, but they are compatible between operating systems. Data load rules files have the extension .rul.
- Outline files. These files are binary files, but they are compatible between operating systems. Outline files have the extension .otl.

The following file types are incompatible between operating systems and need to be redefined or reloaded on the new server:

- Database files with the extensions .db and .dbb
- Data files with the extension .pag
- Index files with the extension .ind
- Security files with the extension .sec
- Application files with the extensions .app and.apb
- Essbase Kernel files with the extension .esm

Note: If you are using the Linked Reporting Objects feature, you need to relink any files or cell notes on the new server. For more information, see Chapter 11, “Linking Objects to Essbase Data.”

Checking File Names

When transferring files to a UNIX system, you need to be aware of the case of file names. UNIX is a case-sensitive operating system, and files are recognized only if they have the correct case. For example, in certain ESSCMD operations, you need to specify a file name, and the file name must be entered with the correct case.

The Essbase system files use the following naming conventions on UNIX systems:

- Executable files have no extension and are uppercase (`ESSBASE`, `ESSCMD`).
- Static library files have the file extension `.a` and are in lowercase (`libessnet.a`).
- Shared library files have the file extension `.sl` on HP-UX, `.so` on Solaris, and `.a` on AIX. These file names are in lowercase (for example, `libesscur.sl`).
- Security files have the file extension `.sec` and are in lowercase (`essbase.sec`).
- Message database files have the file extension `.mdb` and are in lowercase (`essbase.mdb`).
- Online help files have the file extension `.hlp` and are in lowercase (`esscmd.hlp`).

Essbase files on UNIX systems are capitalized with *proper* case: the first letter is uppercase, and the remaining letters are lowercase. This table gives examples of names for different file types:

Table 54: File Naming Examples for UNIX

File Type	**Example**
Database files	`Mydb.db`
Data files	`Mydb.pag`
Index files	`Mydb.ind`
Outline files	`Mydb.otl`
Data load rules files	`Atlanta.rul`
Data files to load	`Atlanta.txt`
Calc scripts	`Mycalc.csc`
Report scripts	`Myrepo.rep`
Archive files	`Mydb.arc`
Application log files	`Myapp.log`

Note: The application name is an exception to the above rule. The application name can be in lower case.

This table lists several examples of valid and invalid file names on UNIX systems:

Table 55: Valid and Invalid File Names on UNIX

Valid File Names	Invalid File Names
`Model.csc`	`MODEL.CSC`
`Monthly.rep`	`Monthly.Rep`
`Forecast.otl`	`forecast.otl`
`Actuals.rul`	`AcTuAlS.rUl`
`My_File.txt`	`My_File.Txt`

Note: The Essbase server does not allow long file names for applications, databases, calculation scripts, reports, and other database files. All file names for objects you create must conform to the Windows 8.3 convention.

Transferring Compatible Files

If two servers are connected, you can create the application and database directories on the new server and use either FTP (File Transfer Protocol) or Application Manager to transfer the compatible application files. If the servers are not connected, you need to redefine server information on the new server before reloading the database.

Using FTP to Transfer

Using FTP, you can transfer files directly between operating systems. You should transfer only the files that are compatible between operating systems, and you should transfer the files in binary mode.

If you have files with the wrong case on a UNIX server, Application Manager can see these files but cannot open them. After you use FTP to transfer files, you should rename the files on the server to ensure that they are capitalized with proper case. Alternatively, you can use FTP to rename the file when you transfer the file:

```
ftp>put oldfile Newfile
```

Using Application Manager to Transfer Files

Using Application Manager, you can transfer files from the client computer to the server by using File > Save As. For example, you can connect to a Windows NT server, open an outline, and then save it to a UNIX server. The file types you can transfer from Application Manager are outline files, report scripts, calculation scripts, and data load rules files.

When you save a file from Application Manager to a UNIX server, proper case is applied automatically regardless of what case you type.

Redefining Server Information

If the server you are porting to is not connected to the existing server, you need to redefine some information on the new server.

➤ To redefine server information, follow these steps:

1. To create users and specify their privileges, use Application Manager on the new server. See Chapter 15, “Managing Security for Users and Applications,” for help.
2. To create the applications and databases that you want to port, use Application Manager on the new server. See Chapter 6, “Creating Applications and Databases,” for help.
3. Copy the outline files (`.otl`) for the databases that you want to port from the old server to the same directory location on the new server. Make sure the application is not running while you copy these files. See “Copying Outlines” on page 170 for help.
4. Copy compatible files from the old server to the new server. For help, see “Identifying Compatible Files” on page 1202.
5. Reload the database. For help, see “Reloading the Database” on page 1206.

Reloading the Database

Database files, such as `.db`, `.pag`, `.esm`, and `.ind`, are not compatible between operating systems. If you port an application to a server on a different operating system, you need to repopulate the database by reloading the data from a data file and a data load rules file (if applicable). One way you can reload is to export the data to an ASCII text file, transfer the text file to the new server, and then use the text file to load data.

Chapter 43

Using Essbase Logs

This chapter describes the logs that OLAP Server creates to record information about server, application, and database activities. Table 56 briefly describes each log:

Table 56: Summary of Logs

Type of Log	Location of Log	Information Included
OLAP Server log	ARBORPATH/essbase.log	Server activities and errors
application log	ARBORPATH\app\application_name\application_name.log	Application activities and errors
outline change log	ARBORPATH\app\application_name\database_name\database_name.olg	Changes to the outline
exception log One of these locations	ARBORPATH\log00001.xcp ARBORPATH\app\log00001.xcp ARBORPATH\app\application_name\log00001.xcp ARBORPATH\app\application_name\database_name\log00001.xcp	Errors that result when OLAP Server stops abnormally
dimension build and data load error logs	ARBORPATH\client\dataload.err or dimbuild.err	Errors from a data load or a dimension build

Use this chapter to discover what information is written to a log and how you can use that information to maintain, tune, or troubleshoot OLAP Server.

This chapter includes the following sections:

- "Understanding Server and Application Logs" on page 1208
- "Using Server and Application Logs" on page 1221
- "Understanding and Using the Outline Change Log" on page 1228
- "Understanding and Using Exception Logs" on page 1233
- "Understanding and Using Dimension Build and Data Load Error Logs" on page 1242

Understanding Server and Application Logs

OLAP Server writes activities that occur in OLAP Server and application logs. The OLAP Serverlog is a text file in the *ARBORPATH* directory named `essbase.log`. In a standard installation, for example, the OLAP Server log is `hyperion\essbase\essbase.log`. Each application on an OLAP Server has its own application log. The application log is a text file in the *ARBORPATH*`\app\`*application_name* directory named *application_name*`.log`. For the Sample Basic database, for example, the application log is `hyperion\essbase\app\sample\sample.log`. Information in these logs can help you to pinpoint where and why an error occurred.

The following sections describe server and application logs:

- "Understanding the Contents of the OLAP Server Log" on page 1209
- "Example of an OLAP Server Log" on page 1210
- "Understanding the Contents of the Application Log" on page 1212
- "Example of an Application Log" on page 1213
- "Server and Application Log Message Categories" on page 1219

For information about the actions you can perform on server and application logs, see "Using Server and Application Logs" on page 1221. For information on filtering, searching, and analyzing logs using Administration Services, see *Essbase Administration Services Online Help*. For information about specific error messages, see the *Error Messages Guide* in `\docs\errmsgs` in your Essbase installation.

Understanding the Contents of the OLAP Server Log

OLAP Server writes activities that occur on the server in *ARBORPATH*`\essbase.log`. Use `essbase.log` to find out more about the activities on the server. These are some of the activities that you can assess:

- Who performed a specific operation
- When an operation was performed
- Errors that occurred when an operation was performed or attempted

This table lists the types of actions logged and the information included in the log message. For information about specific error messages, see the *Error Messages Guide* in `\docs\errmsgs` in your Essbase installation.

Table 57: Contents of the OLAP Server Log (essbase.log)

Type of Action	Information Included
Actions performed at the server level, such as logging into a server or setting security	• User name • Date and time • If changing permissions—user and group information and IP address of user • If logging in—the time and date the user last logged in • If logging out—the length of time the user was logged in • If creating or changing an application or database—request to create or open the object; loading, connecting, and starting the application or database to be created; and locking the object to be created or changed • If creating, changing, or viewing an application or database—requests to list applications and databases and requests to retrieve access information • Shut down or start up requests • Requests to retrieve and delete the OLAP Server log

Table 57: Contents of the OLAP Server Log (essbase.log) (Continued)

Type of Action	Information Included
Actions performed at the application level, such as viewing application settings, or viewing and changing custom-defined macros, custom-defined functions, and substitution variables	• User name • Date and time • Requests to retrieve operating system resources, license information, and system-wide configuration • Retrieving and setting global values • If altering substitution variables—request to list and set substitution variables • If altering custom-defined macros—request to list and delete custom-defined macros • Requests to retrieve and delete the application log
Actions performed at the database level, such as creating data load rules, outlines, reports, or calculation scripts	• User name • Date and time • Requests to retrieve client settings, user, and group information • Requests to list applications and databases • Requests to retrieve access information • Requests to lock objects • Returning objects

Example of an OLAP Server Log

Figure 515 shows the entries written to `essbase.log` when an example OLAP Server starts. First, the Sample application and the Basic database are loaded. The log includes information such as the time the application and database are loaded, the process ID assigned to the application by the operating system, and the startup of the security authentication module.

Figure 515: Startup Messages in the OLAP Server Log

```
[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)
Hyperion Essbase OLAP Server - started
```

Figure 516 shows a single error. The admin user tried to rename an application using a name that already exists on that server. The log includes information such as the user name, the time of the error, and the operation that failed and caused the error.

Figure 516: Error Messages in the OLAP Server Log

```
[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
```

Finally, Figure 517 shows a shutdown. The log includes information such as the name of the application shut down and the time of the shutdown.

Figure 517: Shut Down Messages in the OLAP Server Log

```
[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)
Hyperion Essbase OLAP Server - finished
```

Understanding the Contents of the Application Log

OLAP Server writes application activities that occur, such as calculations, on the server in `ARBORPATH\app\application_name\application_name.log`. For the Sample Basic database, for example, the application log is `hyperion\essbase\app\sample\sample.log`. For information about specific error messages, see the *Error Messages Guide* in `\docs\errmsgs` in your Essbase installation.

Use `application_name.log` to find out more about the activities in an application. These are some of the activities you can assess:

- Who performed a specific operation
- When an operation was performed
- Errors that occurred when an operation was performed or attempted
- Information about dimensions and members to aid in optimization

This table lists the types of actions logged and the information included in the log message.

Table 58: Contents of the Application Log (`application_name.log`)

Type of Action	Information Included
Actions performed at the database level, such as loading data, clearing data, or calculating data	• User name • Date and time • Application and database name and time • If starting application—loading Java modules, and reading and writing database and application information • If loading databases— information about dimension sizes, dynamic calculation members, blocks, cache sizes, index page size, and I/O information • If starting databases—application and database setup information, including reading free space information, writing database parameters, retrieving state information, writing application and database definition information, retrieving database volumes, and writing database mapping • If loading—the load command, parallel data load information, cells updated and elapsed load time

Table 58: Contents of the Application Log (`application_name.log`)

Type of Action	Information Included
Actions performed at the outline level, such as restructuring	• User name • Date and time • Information about dimension sizes, dynamic calculation members, blocks, cache sizes, index page size, I/O information, and restructure elapsed time
Actions performed at the spreadsheet level, such as lock and send	• User name • Date and time • Action performed

Example of an Application Log

The following sections show example entries in the application log, including a standard startup and shutdown, and an example of the messages logged when an error occurs.

- "Example 1: Startup Messages in the Application Log" on page 1213
- "Example 2: Errors in the Application Log" on page 1217
- "Example 3: Shutdown Messages in the Application Log" on page 1218

Example 1: Startup Messages in the Application Log

Figure 518 shows all the entries written to *application_name*.log when OLAP Server starts. The log includes information such as the time the application starts, when application and database information is read and written, when the application is ready for login requests, and when the database is loaded.

Figure 518: Initials Startup Messages in the Application Log

```
[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]
```

After OLAP Server starts, OLAP Server writes information about the dimensions and members in the outline, such as the dimension sizes and dynamic calculation information, to the application log:

Figure 519: Database Outline Messages in the Application Log

```
[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 ]
```

Next, OLAP Server writes information about the blocks in the database, including the block size, the number of declared and possible blocks, and the number of blocks needed to perform calculations (you can use this to estimate the retrieval performance for members of sparse dimensions tagged as Dynamic Calc) to the application log:

Figure 520: Block-Related Messages in the Application Log

```
[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.
```

Next, OLAP Server writes information about the caches set for each database to the application log, as illustrated in Figure 521.

Figure 521: Cache Messages in the Application Log

```
[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 ==> [1024] 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
```

The final messages logged at startup refer to general database information:

Figure 522: General Database Messages in the Application Log

```
[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]
```

Example 2: Errors in the Application Log

The following example shows a single error. An unknown member was found in the data load file; this caused the load to fail. First, you see the request for the data load, then the error message, and, finally, information messages describing the data values changed by the data load and the data load elapsed time.

```
[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]
```

Example 3: Shutdown Messages in the Application Log

The following messages are logged when OLAP Server performs a normal shut down. First information about the database is retrieved, then the database is unloaded, free space information is written, and the server shuts down.

```
[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
```

Server and Application Log Message Categories

Table 59 provides error message categories for each error number range that is shown in the first column. When you receive an error message, use this table to identify the Essbase component to which the error is related. For information about specific error messages, see the *Error Messages Guide* in `\docs\errmsgs` in your Essbase installation.

Table 59: Error Message Categories

Error Message Number Range	Component That Generated the Error
1001000-1001999	Report Writer
1002000-1002999	General server
1003000-1003999	Data load
1004000-1004999	General server
1005000-1005999	Backup, export, or validate
1006000-1006999	Data cache
1007000-1007999	Outline restructure
1008000-1008999	System calls, portable layer, ASD, or Agent
1009000-1009999	Restoring ASCII data
1010000-1010999	Internal (block numbering)
1011000-1011999	Internal (utilities)
1012000-1012999	Calculator
1013000-1013999	Requestor
1014000-1014999	Lock manager
1015000-1015999	Alias table
1016000-1016999	Report Writer
1017000-1017999	Currency
1018000-1018999	Not currently used
1019000-1019999	Database objects
1020000-102999	Spreadsheet extractor

Table 59: Error Message Categories (Continued)

Error Message Number Range	Component That Generated the Error
1021000-1021999	SQL Interface
1022000-1022999	Security
1023000-1023999	Partitioning
1024000-1024999	Query Extractor
1030000-1030999	Application Programming Interface (API)
1040000-1040999	General network
1041000-1041999	Network—Named Pipes
1042000-1042999	Network—TCP
1043000-1049999	Not currently used
1050000-1055999	Agent
1056000-1059999	Not currently used
1060000-1060999	Outline API
106100-1069999	Not currently used
1070000-1070999	Index manager
1071000-1079999	Not currently used
1080000-1080099	Transaction manager
1081000-1089999	Not currently used
1090000-1099999	Rules file processing
1010000-1019999	Not currently used
1100000-1100999	Not currently used
1110000-1119999	Hyperion Web Gateway (HWG)
1120000-1129999	Grid API
1130000-1139999	Miscellaneous
1140000-1149999	Linked Reporting Objects (LRO)
1150000-1159999	Outline synchronization

Table 59: Error Message Categories (Continued)

Error Message Number Range	Component That Generated the Error
1160000-1169999	Outline change records
1170000-1179999	Attributes
1180000-1189999	Showcase
1190000-1199999	Enterprise Integration Services
1200000-1200999	Calculator framework

Using Server and Application Logs

The following sections describe the actions you can perform on server and application logs.

- "Setting the Type of Server Messages Logged" on page 1222
- "Setting the Type of Application Messages Logged" on page 1223
- "Viewing the Server and Application Logs" on page 1225
- "Clearing the Server and Application Logs Immediately" on page 1225
- "Clearing the Server and Application Logs Upon Restart" on page 1227
- "Setting Delimiters in the Server and Application Logs" on page 1227

For information about server and application logs, see "Understanding Server and Application Logs" on page 1208.

You can also view, filter, search, and analyze logs using Administration Services. For more information, see *Essbase Administration Services Online Help*. For information about specific error messages, see the *Error Messages Guide* in `\docs\errmsgs` in your Essbase installation.

Setting the Type of Server Messages Logged

By default, the OLAP Server log, *ARBORPATH*\essbase.log, lists the following types of messages:

- Information messages, such as notification of a user action or information about an application or database

  In the following example, the admin user logged out.

  ```
  [Sun Oct 21 16:00:55 2001]Local/ESSBASE0///Info(1051037)
  Logging out user admin, active for 144 minutes
  ```

- Warning messages, such as a notification that an operation was not completed

  Warnings often follow errors. In the following example, the rename operation did not complete due to a previous error in the log.

  ```
  [Fri Nov 02 13:38:14 2001]Local/ESSBASE0///Warning(1051003)
  Error 1051031 processing request [Rename Application] -
  disconnecting
  ```

- Error messages, such as trying to perform an action that OLAP Server cannot perform

  In the following example, the rename operation failed because the application name already existed.

  ```
  [Fri Nov 02 13:38:14 2001]Local/ESSBASE0///Error(1051031)
  Application Testing already exists
  ```

This table lists the setting that you specify in the OLAP Server's essbase.cfg file to determine what types of messages OLAP Server writes to the OLAP Server log. If you change an essbase.cfg setting, restart OLAP Server to apply the change.

Table 60: Setting Messages in the Server Using essbase.cfg

Setting	Definition	For More Information
AGENTLOGMESSAGELEVEL	An essbase.cfg setting that determines whether OLAP Server writes all messages, warning messages, or error messages to the OLAP Server log	*Technical Reference* in the docs directory

Setting the Type of Application Messages Logged

By default, the application log,
`ARBORPATH\app\`*`application_name`*`\`*`application_name`*`.log`,
lists the following types of messages:

- Information messages that detail routine actions that OLAP Server performs

  In the following example, OLAP Server writes the amount of time elapsed during a data load to the application log:

  ```
  [Fri Nov 02 13:04:15 2001]
  Local/Sample/Basic/admin/Info(1003024)
  Data Load Elapsed Time : [3.014] seconds
  ```

- Warning messages that list conditions that are not deemed serious by OLAP Server

  In the following example, OLAP Server writes a statement that no data values were changed during a data load to the application log:

  ```
  [Fri Nov 02 12:43:44 2001]
  Local/Sample/Basic/admin/Warning(1003035)
  No data values modified by load of this data file
  ```

- Error messages that describe errors that occurred while performing the task

  Error messages can range from serious, such as a file not loading correctly, to very serious, such as a disk space error that causes OLAP Server to crash. In the following example, OLAP Server writes a statement that a data value was encountered before all dimensions in the outline were specified to the application log:

  ```
  [Fri Nov 02 12:53:32 2001]
  Local/Sample/Basic/admin/Error(1003007)
  Data Value [678] Encountered Before All Dimensions Selected,
  [2] Records Completed
  ```

This table lists settings that you specify in the OLAP Server's `essbase.cfg` file and a command that you can use in a calculation script to determine what types of messages OLAP Server writes to the application log. If you change an `essbase.cfg` setting, restart OLAP Server to apply the change.

Table 61: Setting Messages in the Application Log

Setting	**Definition**	**For More Information**
LOGMESSAGELEVEL	An `essbase.cfg` setting that determines whether OLAP Server writes all messages, warning messages, or error messages to the application log	*Technical Reference* in the `docs` directory
TIMINGMESSAGES	An `essbase.cfg` setting that determines whether OLAP Server writes the duration of each spreadsheet and report query to the application log	*Technical Reference* in the `docs` directory
SSLUNKNOWN	An `essbase.cfg` setting that determines whether OLAP Server writes error messages when it encounters an unknown member name during a spreadsheet operation to the application log	*Technical Reference* in the `docs` directory
SET MSG	A calculation script setting that determines whether OLAP Server writes the following items to the application log during the duration of the calculation script: • A summary of calculation statistics • Detailed calculation statistics • All messages, warning messages, error messages, or no messages	*Technical Reference* in the `docs` directory

Viewing the Server and Application Logs

When you view a log, you are viewing a snapshot. To view an updated version of a log, close and reopen the log. You can choose to view a log from a specific date to the present or to view the entire log.

You must have Supervisor permissions to view OLAP Server logs, and you must have at least Application Designer permissions to view application logs.

➤ To view a server or application log, follow the instructions in this table:

Table 62: Viewing the Server and Application Logs

Tool	Instructions	For More Information
Application Manager	To view the OLAP Server log, connect to the server and select Server > View Event Log. To view the application log, connect to the server, select the application, and select Application > View Event Log.	Application Manager online help.
Administration Services	To view the OLAP Server log, select the OLAP server in Enterprise View > right-click > select View Log. To view the application log, select the application in Enterprise View > right-click > select View Log.	*Essbase Administration Services Online Help*

Clearing the Server and Application Logs Immediately

The server and application logs use disk space on the server. Occasionally, you may need to clear entries from a log before it grows too large. Clearing the server log removes all entries in the log, but does not remove the server log itself. Clearing the application log removes all the entries in the application log and deletes the application log itself. Be sure to back up each log before you clear it.

You must have Supervisor permissions to clear server and application logs

➤ To clear a server or application log immediately, use the instructions in this table:

Table 63: Clearing the Server and Application Logs Immediately

Tool	Instructions	For More Information
Application Manager	To clear the OLAP Server log, connect to the server and select Server > Delete Event Log. To clear the application log, connect to the server, select the application, and select Application > Delete Event Log.	Application Manager online help.
Administration Services	To view the OLAP Server log, select the OLAP server in Enterprise View > right-click > select Delete Log. To view the application log,, select the application in Enterprise View > right-click > select Delete Log.	*Essbase Administration Services Online Help*
MaxL	To clear the OLAP Server log: **alter system clear logfile;** To clear the application log: **alter application** *application_name* **clear logfile;**	*Technical Reference* in the `docs` directory

ESSCMD	To clear the server log:
	DELETELOG "";
	To delete the application log:
	DELETELOG "appname";	

➤ To clear a server or application log each time the server or application restarts, see "Clearing the Server and Application Logs Upon Restart" on page 1227.

➤ To conserve space by limiting the items logged in the OLAP Server log or application log, see "Setting the Type of Server Messages Logged" on page 1222 or "Setting the Type of Application Messages Logged" on page 1223.

Clearing the Server and Application Logs Upon Restart

By default, OLAP Server appends messages to the end of the server and application logs. As described in Table 64, you can set OLAP Server to clear the OLAP Server log each time the server is restarted or the application log each time the application is restarted. If you change OLAP Server's `essbase.cfg` file, restart OLAP Server to apply the change.

Table 64: Clearing the Server and Application Logs upon Restart

Setting	Description	For More Information
`CLEARLOGFILE`	An `essbase.cfg` setting that, when set to TRUE, clears the OLAP Server log each time OLAP Server restarts and the application log each time the application restarts	*Technical Reference* in the `docs` directory

Note: To clear a server or application log without restarting the server or application, see "Clearing the Server and Application Logs Immediately" on page 1225. To conserve space by limiting the items logged in the OLAP Server log or application log, see "Setting the Type of Server Messages Logged" on page 1222 or "Setting the Type of Application Messages Logged" on page 1223.

Setting Delimiters in the Server and Application Logs

You can change the symbol used to delimit log entries for server and application logs. This also affects messages logged in the Agent window. By default, OLAP Server uses spaces to delimit fields in a log, as in the following example:

```
[Thu May 10 20:14:46 2001]Local/ESSBASE0///Info(1051051)
Hyperion Essbase OLAP Server - started
```

You can also use characters to delimit the entries in the server and application logs. If possible, use a character that does not occur frequently in your data, application, or database. The following example shows a log entry delimited by tildes (~):

```
Thu~May~10~20:16:13~2001~Local~ESSBASE0~~~Info~(1051051)~ \\
Hyperion Essbase OLAP Server - started
```

This table lists setting that you specify in the OLAP Server `essbase.cfg` file to determine the delimiters that OLAP Server uses in the server and application logs. If you change an `essbase.cfg` setting, restart OLAP Server to apply the change.

Table 65: Setting Delimiters in Logs

Setting	Description	For More Information
`DELMITEDMSG`	An `essbase.cfg` setting that, when set to TRUE, adds a tilde (~) between each field in the server and application logs To specify a different delimiter, use the `DELIMITER` setting.	*Technical Reference* in the `docs` directory
`DELIMITER`	An `essbase.cfg` setting that specifies the delimiter used between each field in the server and application logs OLAP Server enables you to use the following characters as log delimiters: tilde (~), the default delimiter; caret (^); colon (:); ampersand (&); and asterisk (*). The `DELIMTER` setting only works when `DELIMITEDMSG` is set to TRUE.	*Technical Reference* in the `docs` directory

Understanding and Using the Outline Change Log

You can set OLAP Server to create an outline change log that saves outline modification information to a text file. You can review the outline change log at any time to see all the changes that have been made to an outline since the log was first created. This information helps you to roll back an outline to a previous version. The outline change log is a text file in the *`ARBORPATH`*`\app\`*`application_name`*`\`*`database_name`* directory named *`database_name`*`.olg`. For the Sample Basic database, for example, the outline change log is `hyperion\essbase\app\sample\basic\basic.olg`.

The following sections describe the outline change log and the actions that you can perform on it.

- "Understanding the Contents of the Outline Change Log" on page 1229
- "Example of an Outline Change Log" on page 1231
- "Creating Outline Change Logs" on page 1232
- "Viewing Outline Change Logs" on page 1232
- "Setting the Size of the Outline Change Log" on page 1232

Understanding the Contents of the Outline Change Log

You can set OLAP Server to write changes to the outline to `ARBORPATH\app\application_name\database_name\database_name.olg`.

This table lists the types of actions written to the outline change log and the information included in the log message.

Table 66: Outline Change Log Contents

Type of Change	Information Included
Add a dimension	• Name of dimension • Type of dimension (dense or sparse) • Dimension tag (if any) • Name of left sibling of dimension (if any) • Level number of dimension • Generation number of dimension
Delete a dimension	• Name of dimension
Update a dimension	• Name of dimension • Dimension tag • Type of dimension (dense or sparse) • Level name changes (if applicable) • Generation name changes (if applicable)
Rename a dimension	• Old name of dimension • New name of dimension

Table 66: Outline Change Log Contents (Continued)

Type of Change	Information Included
Move a dimension to a new position	• Name of dimension • Old location, including left sibling of dimension • New location, including left sibling of dimension
Add a member to a dimension	• Name of new member or members • Unary calculation symbol for member • Level number of member • Generation number of member • Status of member (Store, Share) • Member alias (if applicable) • Account type of member (if applicable) • User-defined attributes of member (if applicable) • Calculation formula for member (if applicable)
Update a member of a dimension	• Name of member updated • Member properties that were updated
Rename a member of a dimension	• Old name of member • New name of member
Move a member of a dimension to a new position	• Name of member moved • New location • Names of parent and left sibling in new location

The outline change log program reads outline information from left to right. If you are looking at an outline, the *left sibling* is the sibling directly above (to the left of) the newly added dimension or member. This rule does not apply if the immediately preceding dimension or member is a *parent*. If a newly added (or moved) member is the first child of its parent or if the member is the first dimension in the outline, the outline change log identifies the old location of the dimension or member as None.

Example of an Outline Change Log

When a user makes and saves changes to an outline, Essbase writes the change information into the outline change log as a group of entries. Each group of entries begins and ends with identifying information so that you can easily identify each revision, from newest to oldest.

Figure 523 shows one record, indicating that since the outline change log was started, the outline was modified once. First, the outline change log lists the beginning of the change, including the application and database changed; the time of the change; and the user making the change. Next, the outline change log lists the change: a member named 100-50 was added to the Product dimension. Finally, the outline log lists the end of the change, including the application and database changed; the time of the change; and the user making the change.

Figure 523: Sample Outline Change Log

```
[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]
```

Creating Outline Change Logs

By default, OLAP Server does not create an outline change log. Table 67 lists the setting that you specify in the OLAP Server's `essbase.cfg` file to create an outline change log. If you change an `essbase.cfg` setting, restart OLAP Server to apply the change.

Table 67: Creating an Outline Change Log Using `essbase.cfg`

Setting	Description	For More Information
`OUTLINECHANGELOG`	An `essbase.cfg` setting that, when set to TRUE, creates an outline change log	*Technical Reference* in the `docs` directory

Note: During a restructure, Essbase holds outline change information in memory until all updates have been made to the outline change log. Turning on the outline change log may, therefore, affect restructure performance. "Conditions Affecting Database Restructuring" on page 1345 discusses other conditions that affect restructure performance.

Viewing Outline Change Logs

Because the outline change log is an ASCII text file, you can view it by opening it in any text editor. The outline change log is located in the *ARBORPATH*`\app\`*application_name*`\`*database_name* directory and is named *database_name*`.olg`. For the Sample Basic database, for example, the outline change log is located in `hyperion\essbase\app\sample\basic\basic.olg`.

Setting the Size of the Outline Change Log

The default size for the outline change log is 64,000 bytes. When the log reaches this size, Essbase copies the contents of *database_name*`.olg` to a new file with the `.olb` extension. For example, when `basic.olg` reaches 64,000 bytes the contents are copied to basic.olb. After this copy, *database_name*`.olg` is cleared. New log entries are written to `basic.olg`.

Each time the outline change log reaches its maximum file size, Essbase writes over the existing *database_name*`.olb` with the current *database_name*`.olg`. Retrieve information from *database_name*`.olb` before *database_name*`.olg` fills up.

The default, minimum, and maximum file sizes for the *database_name*.olb file are the same as the corresponding defaults specified for the *database_name*.olg file. For example, if you change the maximum size of the outline change log to 2 MB, you automatically set the .olb file to the same maximum size.

Table 68 lists the setting that you specify in the OLAP Server's essbase.cfg file to set the size of the *database_name*.olb file. If you change an essbase.cfg setting, restart OLAP Server to apply the change.

Table 68: Setting the Outline Change Log Size Using essbase.cfg

Setting	Instructions	For More Information
OUTLINECHANGE LOGFILESIZE	An essbase.cfg setting that determines the size of the outline change log, *database_name*.olg	*Technical Reference* in the docs directory

Understanding and Using Exception Logs

When an OLAP Server, an application, or a database shut down incorrectly, OLAP Server sometimes creates a text file named log0000*n*.xcp. The following sections describe the server, application, and database exception logs and the actions that you can perform on them.

- "Understanding the Contents of Exception Logs" on page 1233
- "Example of an Exception Log" on page 1236
- "Viewing Exception Logs" on page 1241
- "Overwriting or Saving Existing Logs" on page 1241

Understanding the Contents of Exception Logs

If an OLAP Server, an application, or a database shuts down abnormally and cannot restart, OLAP Server generates an exception log to help troubleshoot the problem. The location of the exception log depends on the component that shut down abnormally and the amount of information that OLAP Server had available at the time of the abnormal shutdown. Table 70 on page 1241 describes the location of the exception log for each type of abnormal shutdown.

This table lists the sections of the exception log and the information included in each section. If OLAP Server could not retrieve all the information before the shutdown finished, some of the later sections may be blank.

Table 69: Contents of the Exception Log (`log00001.xcp`*)*

Section of Log	Information Included
General information	• Date and time • Application and database name • Location of exception log • Process type Use this information to determine which component shut down abnormally, and when it shut down.
Machine registers and stack information	• General registers • Floating point registers • Hex registers • Stack trace • Stack dump Contact Hyperion Technical Support for information on how to analyze this information.
Application-wide configuration	• Server and application name • Elapsed application time; that is, how long the application was running • List of modules Use this information to determine if the application shut down quickly and that all modules are correct. More information about modules is in the system-wide configuration section of the exception log.
Operating system resources	• System date and time • Elapsed operating system time; that is, how long the operating system was running • Resource information, including CPU type, memory information, swap information, and drive information Use this information to see if it is an operating system problem, such as a lack of memory.

Table 69: Contents of the Exception Log (`log00001.xcp`) (Continued)

Section of Log	**Information Included**
System-wide configuration	• Elapsed Essbase time; that is, how long Essbase was running • Essbase release information • Network information • Environment variables • Module information, including module name and release Use this information to make sure that the release is the same for Essbase and each module, and the environment variables are set correctly.
`essbase.cfg` values	• Values of all settings in the `essbase.cfg` file Use this information to make sure that your OLAP Server is configured correctly.
License information	• Serial number and license expiration date • Number of ports purchased and ports in use • Essbase options enabled • Other Hyperion products enabled Use this information to make sure that the correct options of Essbase are installed and that you have purchased enough ports.
Client request activity	• Server name • Application name • Thread information, including the number of threads • Request information • Detailed information about each thread, including the action it is performing, the database, user name, start time, and end time Use this information to see how heavy the load on the server was, based on client requests.

Table 69: Contents of the Exception Log (`log00001.xcp`) (Continued)

Section of Log	**Information Included**
File information	• Page file size • Index file size Use this information to determine whether the page file or the index is too large.
Database information	Use this information to make sure that your database is set up correctly.
Database statistics	Use this information to view dimension information and to see characteristics of data blocks in the database.

Example of an Exception Log

The following example is of an exception log. The first section of the log lists general information about the application and database. In this example, the OLAP Server shut down:

```
----- 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)
```

The next section of the log lists register and stack trace information. Hyperion Technical Support can examine this section of the log to assist in determining why an abnormal shutdown may have occurred.

```
----- 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...
```

The following section of the log lists application information:

```
----- 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...
```

The following section of the log lists operating system information. You can determine how much memory is available, how much swap space is used, and how much memory is available on each drive:

```
----- 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...
```

The following section of the log lists system configuration information, such as paths or `essbase.cfg` settings:

```
----- 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
```

The following section of the log lists license information (such as a serial number), the Essbase options (such as ports purchased), and Hyperion products purchased (such as Enterprise Integration Services):

```
----- License Information -----

Serial Number:         110047420012631A-00A4B010D74
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...
```

The following section of the log lists client activity, such as using Application Manager to view databases or using the Spreadsheet Add-in to view databases:

```
----- 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 -----
```

Viewing Exception Logs

Because the exception change log is an ASCII text file, you can view it by opening it in any text editor. The location of the exception log depends on the component that shut down abnormally and the amount of information that OLAP Server had available at the time of the abnormal shutdown.

This table describes the location of the exception log.

Table 70: Location of the Exception Log

Component That Shut Down	Location of the Exception Log
OLAP Server	The log is in the `ARBORPATH` directory; for example, `d:\essbase\log00001.xcp`.
Application	If the application name is unknown, the log is in the `ARBORPATH\APP` directory; for example, `d:\essbase\app\log00001.xcp.` If the application name is known, the log is in the application directory; for example, if the Sample application shut down abnormally, `d:\essbase\app\sample\log00001.xcp.`
Database	If the database name is unknown, the log is in the application directory; for example, if the Basic database shut down abnormally, `d:\essbase\app\sample\log00001.xcp`. If the database name is known, the log is in the database directory; for example, if the Basic database shut down abnormally, `d:\essbase\app\sample\basic\log00001.xcp`.

Overwriting or Saving Existing Logs

By default, OLAP Server creates a new exception log each time it shuts down abnormally. Subsequent exception logs are numbered sequentially; for example, if `log00001.xcp` exists, the next log is named `log00002.xcp`, the next is `log00003.xcp`, and so on.

You can change a setting in the OLAP Server `essbase.cfg` to overwrite the existing exception log instead of creating a new log. If you change an `essbase.cfg` setting, restart OLAP Server to apply the change.

Use this table for more information about creating new exception logs:

Table 71: Creating New Exception Logs

Setting	Description	For More Information
`EXCEPTIONLOG OVERWRITE`	An `essbase.cfg` setting that, when set to FALSE, creates a new exception log. When set to TRUE, OLAP Server overwrites the existing exception log.	*Technical Reference* in the `docs` directory

Understanding and Using Dimension Build and Data Load Error Logs

OLAP Server writes errors that occur during a dimension build or data load in error logs. The log that OLAP Server chooses for errors depends on the operation that you perform, such as a data load or a dimension build, and how you perform it, such as using Application Manager or ESSCMD. The following sections describe the location of dimension build and data load errors and the actions that you can perform on data load and dimension build error logs:

- "Understanding Dimension Build and Data Load Error Logs" on page 1242
- "Example of a Dimension Build and Data Load Error Log" on page 1243
- "Viewing Dimension Build and Data Load Error Logs" on page 1244
- "Setting the Maximum Number of Errors" on page 1244
- "Loading Dimension Build or Data Load Error Logs" on page 1245

Understanding Dimension Build and Data Load Error Logs

The `dimbuild.err` and `dataload.err` logs contain errors that occurred during a dimension build or a data load. The logs also contain the records that failed to load. After you fix the errors, you can reload the log files. For more information, see "Loading Dimension Build or Data Load Error Logs" on page 1245.

The log that OLAP Server chooses for errors that occur during a dimension build or data load depends on how the dimension build or data load was started.

Use this table to find the default location for errors:

Table 72: Location of Dimension Build and Data Load Errors

Operation	**Tool**	**Location of Error Log**
Dimension build	Database > Load Data command in the Application Desktop window by choosing to modify the outline	*ARBORPATH*\client\dataload.err
	File > Update Outline command in the Outline Editor	*ARBORPATH*\client\dimbuild.err
Data load	Database > Load Data command in the Application Desktop window without using a rules file	*ARBORPATH*\application_name\\ *application_name*.log For more information, see "Understanding the Contents of the Application Log" on page 1212.
	Database > Load Data command in the Application Desktop window using a rules file	*ARBORPATH*\client\dataload.err

To set the location and file name of dimension build and data load error logs, see "Setting a Load or Build Error Log" on page 651.

Example of a Dimension Build and Data Load Error Log

Figure 524 shows an entry in a data load log. The 500 member did not exist in the outline, so no data could be loaded to it. To solve this problem, you can perform a dimension build using the error log to create the missing member or you can add the missing member in the Outline Editor. Then restart the load.

Figure 524: Error Message in the Data Load Log

```
\\ Member 500 Not Found In Database
500   500-10   500-10-10
```

Figure 525shows an entry in a dimension build log. The 600-20 member is not the parent of the 600 member. Make sure that you use the correct rules file with the correct text file. The record looks like it is for a level (bottom-up) build, but the error message indicates that OLAP Server is trying to perform a generation (top-down) build. When you correct the problem, restart the dimension build.

Figure 525: Error Message in the Dimension Build Log

```
\\Record #2 - Incorrect Parent [600-20] For Member [600] (3307)
600-20-10   600-20   600
```

Viewing Dimension Build and Data Load Error Logs

Because the dimension build and data load error logs are ASCII text files, you can view them by opening them in any text editor. The dimension build error log is located in the `ARBORPATH\client` directory and is named `dimbuild.err`. The data load error log is located in the `ARBORPPATH\client` directory and is named `dataload.err`. In a standard installation, for example, the error logs are located in the `hyperion\essbase\client` directory.

If you have no error logs in that directory, see "Setting a Load or Build Error Log" on page 651.

Setting the Maximum Number of Errors

The default size of the `dimbuild.err` and the `dataload.err` files is 1,000 records. When the log reaches this size, OLAP Server does not write any other errors that it encounters to the log. The dimension build or data load, however, continues. Any subsequent errors are lost. You can set OLAP Server to write 1 to 65,000 records in the `dimbuild.err` and the `dataload.err` files. If you change the OLAP Server `essbase.cfg` file, restart OLAP Server to apply the change.

Table 73: Setting the Number of Error Records

Setting	Description	For More Information
`DATAERRORLIMIT`	An `essbase.cfg` setting that determines the number of records logged in the `dimbuild.err` and `dataload.err` files	*Technical Reference* in the `docs` directory

Loading Dimension Build or Data Load Error Logs

If the data load or dimension build fails, you must determine whether the Isolation Level transaction setting is Committed or Uncommitted. For more information, see "Isolation Levels" on page 1132. If the Isolation Level transaction setting is Committed, you must restart the data load from the beginning. If the Isolation Level is Uncommitted, you can load only the records that failed by loading the error log. Only reloading the failed records is much faster than reloading every record, including those records that succeeded during the first load.

➤ To reload the error log:

1. If you load from the server, change the file extension from `.err` to `.txt`. For example, change the `dataload.err` file to `dataload.txt`.

   If you load from the client, you can leave the `.err` extension.

2. Fix the problem that caused the dimension build or data load to fail. This might involve changing the outline, the text in the error log, or the rules file.

   Check the following:

   - Can you validate the rules file? See "Step 6: Validating Dimension Build Rules" on page 554.
   - Is the data source available? See Chapter 50, "Optimizing Data Loads."
   - Is the server available? See Chapter 50, "Optimizing Data Loads."

3. Load the error log using the appropriate rules file. For more information, see Chapter 20, "Introducing Data Loading."

Chapter 44 Backing Up and Restoring Data

This chapter describes how to back up a database and lists the Essbase files that are essential to restoring a database from backups.

This chapter includes the following sections:

If you are migrating from a previous release of Essbase, see the *Essbase Installation Guide*.

Backing Up a Database

A key part of a database maintenance routine includes regular backups of Essbase data. It is important to integrate regular database backups into your production server maintenance.

The frequency of backups is dependent upon the volatility of the database and server environment, as well as upon the demand for quick database restores in the event of server crashes.

There are two methods of backing up a database:

- Preparing the database for file system backup
- Exporting, which makes a copy of data in an ASCII text format

This section tells you which files should be backed up regularly and describes each backup method.

Files to Back Up

You should regularly back up the server, application, and database files listed in Table 74:

Table 74: Files to Back Up

File	Stored Where
`ess`*`n`*`.ind`	`\essbase\app\`*`appname`*`\`*`dbname`*
`ess`*`n`*`.pag`	`\essbase\app\`*`appname`*`\`*`dbname`*
`dbname``.esm`	`\essbase\app\`*`appname`*`\`*`dbname`*
`dbname``.tct`	`\essbase\app\`*`appname`*`\`*`dbname`*
`dbname``.ind`	`\essbase\app\`*`appname`*`\`*`dbname`*
`appname``.app`	`\essbase\app`
`dbname``.db`	`\essbase\app\`*`appname`*`\`*`dbname`*
`x``.lro`	`\essbase\app\`*`appname`*`\`*`dbname`*
`essbase.sec`	`\essbase\bin`
`essbase.bak`	`\essbase\bin`
`essbase.cfg`	`\essbase\bin`
Database object files such as `.otl`, `.csc`, `.rul`, `.rep`, `.eqd`, and `.sel`	`\essbase\app\appname\dbname`
ESSCMD or MaxL scripts	No defined storage location

It is important to back up all `.ind` and `.pag` files related to a database because a single database can have multiple `.ind` and `.pag` files. Remember, the Agent should be shut down before the `essbase.sec` file is backed up.

For a full list and description of all Essbase files, see "How Essbase Files Are Stored" on page 1182.

File System Backup

A common method of creating database backups is by doing a file system backup of the OLAP Server. You can perform the backup using the file system backup software of your choice. You can back up specific directories or files, or you can back up the entire Essbase directory structure.

In most cases, backups occur after Essbase applications and databases, as well as the Agent, are shut down. However, due to user requirements, some Essbase databases must be up and running at the time of backup. For more information, see "Placing a Database in Read-Only Mode" on page 1249.

Be sure to back up data on every disk volume Essbase uses. For information about data storage on multiple volumes, see "Storage Allocation" on page 1115.

Placing a Database in Read-Only Mode

Essbase provides a way to prepare a database for backup when the database must remain running during the backup process. Placing the database in read-only (or "archive") mode protects the database from updates during the backup process. After you perform the backup using the third-party backup utility of your choice, you then return the database to read-write mode.

➤ To place a database in read-only mode, use either of the following methods:

Tool	Instructions	For more information
MaxL	**alter database begin archive**	*Technical Reference* in the `docs` directory
ESSCMD	BEGINARCHIVE	*Technical Reference* in the `docs` directory

Note: If you try to cancel the BEGINARCHIVE ESSCMD command or the 'alter database begin archive' MaxL statement and you receive a "can't cancel" message, the system is most likely in the final stage of writing items to the drive and has reached the point where the operation cannot be cancelled.

➤ To return the database to read-write mode, use either of the following methods:

Tool	Instructions	For more information
MaxL	**alter database end archive**	*Technical Reference* in the `docs` directory
ESSCMD	Use ENDARCHIVE command	*Technical Reference* in the `docs` directory

The begin-archive utility does the following:

- Commits any modified data to disk.
- Switches the database to read-only mode.
- Reopens the database files in shared, read-only mode.
- Creates a file containing a list of files that need to be backed up. By default, the file is called `archive.lst`. It is stored in the database directory.

If a user tries to modify data during the backup process, an error message informs the user that data is in read-only mode for backup.

The end-archive utility does the following:

- Returns the database to read-write mode.
- Re-opens database files in exclusive, read-write mode.

Begin-archive and end-archive utilities do not perform the backup; they simply protect the database during the backup process.

CAUTION: If you back up your data without using a begin-archive utility, make sure that all Essbase applications are closed and that all users are logged out during the backup process. Otherwise, you risk corrupting the database.

Performing a Backup

After putting the database in read-only mode, you are ready to perform the backup.

➤ To backup data, use a third-party backup utility to back up the files listed in `archive.lst`.

Make sure you back up the files listed in "Files to Back Up" on page 1248. Alternatively, you can back up the entire Essbase directory structure.

For information on restoring files from backup, see "Restoring Data from Backups" on page 1255.

Returning a Database to Read-Write Mode

After performing the backup, you need to return the database to read-write mode.

➤ To return the database to read-write mode, issue an 'alter database end archive' statement in MaxL, or the ENDARCHIVE command in ESSCMD.

See the *Technical Reference* in the `docs` directory for syntax information.

Note: You must use the end-archive utility to put the database back into read-write mode, even if you shut down and restart the database. The end-archive utility does not restart the database.

Using Export to Back Up Data

You can back up data by exporting it. Exporting data copies it to an ASCII text file that you specify; it does not compress data. The export file contains data only and does not include control, outline, or security information.

You might consider exporting data for the following reasons:

- To transfer data across platforms
- To back up only a certain portion of the data; for example, level 0 blocks
- To create an exported file in text format, rather than binary format

Note: You can export subsets of data by creating reports. For more information, see Chapter 37, "Copying Data Subsets and Exporting Data to Other Programs."

Export Considerations

Using export to back up data provides the following advantages:

- You can use the resulting ASCII files to load data from the source database into databases on other platforms.

  When loading an export file into a database, it is important that the database outline contains all the members found within the export file. If not, the load will fail. Also, if the outline changes between the time that the export file is created and reloaded (and the new outline contains all the members found within the export file), the load time might be significantly higher than if the outlines were identical.
- During an export, data integrity is verified because every block is checked to confirm whether corresponding page and index files match.
- You can reduce fragmentation in a database by exporting data into an ASCII file, clearing all data from the database, and reloading the ASCII file.
- You can export a database in column format from Application Manager or MaxL. Then, you can use a data load rules file to load the column-formatted file. Using column format is helpful when you need to manipulate the export file.

Using export to back up data provides the following disadvantages:

- Because dynamic calculations are not executed at the time of the export, only stored data and data from previously calculated Dynamic Calc And Store members are included in the export.
- At the time of a database export, Essbase users cannot write to the database. Users receive an error message if they try to write to the database during an export. After an export has started, users can do read operations. Exports of large databases require considerable amounts of time, time during which users can only read the data.

Exporting Data

To export data, use any of these clients:

Tool	Instructions	For More Information
Application Manager	Select database > Database > Export	*Essbase Application Manager Online Help*
ESSCMD	EXPORT or PAREXPORT	*Technical Reference* in the `docs` directory
MaxL	**export data**	*Technical Reference* in the `docs` directory

All methods require the same basic information:

- The amount of data to export:
  - All data
  - Level 0 blocks only (blocks containing only level 0 sparse member combinations)
  - Data from input blocks only (blocks containing data from a previous data load or spreadsheet Lock & Send)
- Whether to export data in a columnar or non-columnar format

  To facilitate loading the exported data into a relational database, export the data in columns. In each row, the columnar format displays a member name from every dimension. Names can be repeated from row to row.

  The columnar format provides a structure to the exported data, so that it can be used for further data processing by applications other than Essbase tools. In non-columnar format, sparse members identifying a data block are included only once for the block. Because the export file in non-columnar format is smaller than in columnar format, reloading a file in non-columnar format is faster.
- The export data file names

Improving Export Performance

To improve export performance, you can now export data in parallel to a specified number of files, using the **export** statement in MaxL or the PAREXPORT command in ESSCMD. For details see the *Technical Reference* in the `docs` directory.

Exporting Files Larger Than 2 GB

Some file management systems do not support ASCII files larger than 2 GB. On any operating system, if Essbase anticipates that an export file exceeds 2 GB, it creates two or more export files, as needed.

When Essbase creates multiple export files, it uses the requested file name for the main file. An underscore and a sequential cardinal number are appended to the names of the additional files, starting with `_1`. For example, if the requested file name is `expJan.txt` and the exported data would exceed 4 GB, Essbase creates three files, naming them: `expJan.txt`, `expJan_1.txt`, and `expJan_2.txt`. Exported data files can be reloaded in any sequence.

Reloading Exported Data

To reload exported data, use any one of the following tools:

Tool	Instructions	For More Information
Application Manager	Perform a data load without a rules file. Select database > select Database > Data Load	Chapter 23, “Performing and Debugging a Data Load”
ESSCMD	IMPORT	*Technical Reference* in the `docs` directory
MaxL	**import data**	*Technical Reference* in the `docs` directory

When you reload data that has been exported, Essbase marks the data as input data. If you reload data exported from level 0 blocks or input blocks, you must recalculate the database after reloading. When you recalculate the database, Essbase recalculates every data block.

If you export *all* data in a database and then reload, Essbase marks all blocks in the database as input blocks. Consequently, if you try to clear data by selecting Database > Clear Data > Non-Input Blocks from Application Manager, no data is cleared because the database contains no non-input blocks.

When you reload data that has been exported, Essbase also marks the data blocks as dirty. If you had calculated the database prior to exporting it, to save time during the next calculation, you should set the status of the blocks as clean. If you had not calculated the database prior to exporting it, you do not need to set the status of the blocks as clean.

➤ To clean the status of the blocks in a database after exporting all data and reloading, you can run the following calculation script:

```
Set ClearUpdateStatus Only;
Calc All;
```

Restoring Data from Backups

To restore a database, replace the files on disk with the corresponding files from backup. See "Files to Back Up" on page 1248 for a list of files that should be backed up on a regular basis.

The application should be stopped, unless you are restoring from an export file. In that case, ensure the application is not accepting client connections.

Essential Database Files

These files are all key components of an Essbase database:

Table 75: Essential Database Files

File	Description
`ess`*`n`*`.pag`	Essbase data file
`ess`*`n`*`.ind`	Essbase index file
`dbname``.esm`	Essbase Kernel file that contains control information used for database recovery
`dbname``.tct`	Transaction control table
`dbname``.ind`	Free fragment file for data and index free fragments
`dbname``.otl`	Outline file, which does not store data but does store all metadata for a database and defines how data is stored

If there is a problem with any one of these files, the entire database becomes corrupted. The database must then be restored from backups or reloaded from exports.

There have been cases in which database files have become corrupted. In such situations, the database is not able to start up on the OLAP Server. Therefore, no data can be reloaded to restore the database. In these cases, the only way to restore the database is to delete all of the following files:

- `ess`*`n`*`.pag`
- `ess`*`n`*`.ind`
- *`dbname`*`.esm`
- *`dbname`*`.tct`
- *`dbname`*`.ind`

After the files are deleted, restart the database and reload from data files or from export files created prior to the corruption.

Chapter 45

Automating the Production Environment

The OLAP Server includes MaxL and ESSCMD, both command-line interfaces that perform operations interactively or through batch or script files.

MaxL improves upon ESSCMD in that it is a multi-dimensional access language for Essbase. Using MaxL, you make requests to Essbase using English-like statements, rather than commands. MaxL scripts can be developed flexibly to accommodate many uses, and the MaxL language is easy to learn. MaxL is installed with a Perl module that enables you to embed its statements in Perl programs.

This chapter focuses on ESSCMD. It describes how to use ESSCMD's interactive and batch processing modes, and provides sample script and batch files. It includes the following topics:

For an overview of MaxL, see the *MaxL User's Guide* (`essmaxl.pdf` in the `\docs\pdf` directory of your Essbase installation).

For MaxL and ESSCMD syntax and usage information, see the *Technical Reference* in the `docs` directory.

For information on the Administration Services and MaxL Script Editor, see *Essbase Administration Services Online Help*.

ESSCMD Basics

With ESSCMD, you can execute server operations at the command line, in either interactive or batch mode.

Interactive mode means entering commands at the ESSCMD command line, and receiving prompts where necessary. Interactive mode is convenient for short operations that require few commands, checking for information on the fly, and error checking.

Batch processing mode is used for automating routine server maintenance and diagnostic tasks. You can write a script or batch file and run it from the command line. Batch processing mode is convenient if you frequently use a particular series of commands, or if a task requires many commands.

Syntax Guidelines

In general, use the same syntax for entering ESSCMD commands as you do for other calculation commands. However, there are differences between ESSCMD's interactive and batch processing modes in the requirements for quotation marks and the semicolon statement terminator. Use the guidelines in this section when creating script or batch files.

Quotation Marks

Quotation marks ("") enclose character parameters and responses to commands.

- In interactive ESSCMD, using quotes is optional. Be sure to use quotes when a parameter has an embedded space; for example:

  ```
  CALC “Calc All;”
  ```

- In an ESSCMD script file, always enclose all character parameters and responses to commands in quotation marks; for example:

  ```
  LOGIN “Localhost” “user1” “Password”;
  ```

- Numeric parameters and responses do not require quotation marks.
- Do not enclose quotation marks within quotation marks.

Semicolon Statement Terminator

The ; (semicolon) statement terminator signals the end of a command; for example:

```
SELECT “SAMPLE” “BASIC”;
```

- In interactive ESSCMD, pressing the Enter key signals ESSCMD that the command is complete. The statement terminator is optional.
- In an ESSCMD script file, you should use the terminator, even though it is optional, if a command has many parameters. This is especially important in order to signal the end of the parameter list if some of the parameters are optional.

  If you omit some optional parameters and do not use a semicolon to end the list, ESSCMD looks for the remaining values in the next command in the file, leading to unpredictable results.

The SETAPPSTATE and SETDBSTATE commands, defined in the *Technical Reference* in the `docs` directory, are examples of commands which you should terminate with a semicolon to prevent any confusion in processing.

Note: All syntax examples in this chapter use quotation marks and semicolon terminators.

Running ESSCMD on Different Operating System Platforms

ESSCMD operates independently of any Essbase client interface, including the Application Manager, Spreadsheet Add-in, or custom-built application programs.

ESSCMD is available on the platforms listed in the *Essbase Installation Guide*, in the server platform system requirements section.

Canceling ESSCMD Operations

When running ESSCMD, you can cancel an asynchronous operation, such as a calculation, export, or restructure operation, by pressing and holding the Esc key until ESSCMD responds.

Referencing Files

Some commands require that you precede object or file names with a numeric parameter, from 1 to 4, that tells Essbase where to look for the object or file. The parameter directs ESSCMD to look for files in other applications, databases, or systems.

The following table lists each value for the numeric parameter (*Number*), the file location to which it applies, and the information that ESSCMD requests when you use each parameter setting. *appName* is the application name and *dbName* is the database name.

Number	**File**	**Essbase prompts user for:**
1	Local or client-based file	Windows: Files in the `\essbase\client\`*`appname`*`\`*`dbName`* directory UNIX: Files in the `essbase/client/`*`appName`*`/`*`dbName`* directory
2	Remote or server-based file	Windows: files in the `\essbase\app\`*`appName`*`\`*`dbName`* directory UNIX: Files in the `essbase/app/`*`appName`*`/`*`dbName`* directory
3	File	Fully-qualified path to the file, unless file is in the current ESSCMD directory
4	SQL table	Full network and database information for the SQL table

For example, the LOADDATA command can load a data file that resides on the client or OLAP Server. The command requires the numeric parameter to tell Essbase where to look for the data file. This example causes ESSCMD to prompt for the fully-qualified path name of the file to load:

```
LOADDATA 3
```

File extensions are usually optional in both interactive and batch processing modes, except when using commands that require a numeric parameter that indicates the location of files:

- If you use file option 3 (File), you must enter the file extension in both interactive and batch processing modes.
- If the object is in the directory from which you started ESSCMD, you do not need to enter a path.

Multi-User Considerations

Because ESSCMD supports multiple login instances on OLAP Server, you can access more than one database in a single session. Even when you log in to multiple databases, you use only one port on your server license.

Case-Sensitivity

The OLAP Server creates application and database names exactly as the user specifies. Case is not changed for any platform.

For backward compatibility, the OLAP Server searches for existing application and database names using the exact case first. However, if it cannot find the file, OLAP Server searches all possible case combinations to find the existing application and database names.

Essbase does not allow you to create application and database names that differ only in case. For example, Essbase would display an error message if you tried to create an application MyData if you already had an application mYdATA.

You can choose to make member names case sensitive. For details, see "Making Members Case-Sensitive" on page 188.

Getting Help

To display a list of all currently available ESSCMD commands, enter `HELP?`. To see documented descriptions and syntax for individual ESSCMD commands, see the online *Technical Reference*, located in the `ESSBASE/DOCS` directory.

Before You Start ESSCMD

Before you start ESSCMD, make sure that the following items are properly installed and running:

- Essbase
- Communications protocol (Named Pipes or TCP/IP) on Essbase

For information on protocols supported by Essbase, see the *Essbase Installation Guide*.

Starting and Quitting ESSCMD

The Essbase installation places the `esscmd.exe` and `esscmd.hlp` files (`esscmd` and `esscmd.hlp` on UNIX platforms) in the `BIN` directory of your application server.

Once you start the application, a command prompt like this one displays:

```
:::[n]->
```

where *n* is the value of the active login instance. Each subsequent, successful login increments this value by one. When you start ESSCMD, the instance number is zero (0).

Note: Use the SETLOGIN command to toggle between active login instances. Use the LISTLOGINS command to view active login instances. For more information, see the online *Technical Reference*, located in the `essbase/docs` directory.

➤ To start ESSCMD, do one of the following:

- Enter `ESSCMD` at the operating system command prompt. ESSCMD runs within the operating system command line window.
- Run `esscmd.exe` (`esscmd` on UNIX platforms), located in the `bin` directory

➤ To quit ESSCMD, Enter EXIT at the prompt and press Enter. ESSCMD disconnects from the application server, and terminates the session.

Using Interactive Mode

In interactive mode, you enter commands and respond to prompts. This is useful when you are performing simple tasks that require few commands. If you are performing more complex tasks that require many commands, consider creating a script file or batch file; see "Using Script and Batch Files for Batch Processing" on page 1265 for information.

For syntax conventions when working in interactive mode, see "Syntax Guidelines" on page 1258.

Logging into the Server

After starting ESSCMD, you must connect to OLAP Server so that you can enter commands. Follow these steps:

1. At the ESSCMD prompt, log in to the server with the LOGIN command. For more information about this command, see the online *Technical Reference*, located in the `essbase/docs` directory.
2. Enter the server name. When you connect from the server console, the server name depends on your network setup. For example, the name could be `aspen`.
3. Enter your user name.
4. Enter your password.

The ESSCMD prompt is displayed as follows:

```
aspen:::userName[1]->
```

userName is your login name.

You can enter any valid ESSCMD command. For a complete listing of commands, type HELP.

Note: To load an application into memory and select a database on the Essbase server, use the SELECT command to select a database from an application that resides on the server. For more information, see the online *Technical Reference*, located in the *`ARBORPATH`*`/docs` directory.

The ESSCMD prompt is displayed as follows:

```
aspen:appName:dbName:userName[1]->
```

appName is the name of the application. *dbName* is the name of the database to which you are connected.

Entering Commands

There are two ways to enter commands in interactive mode. Choose either of the following methods to enter commands:

- Type the command and press Enter.

  ESSCMD prompts you for each of the command parameters. For example, the SELECT command has two parameters, as shown in the command syntax:

  ```
  SELECT "appName" "dbName";
  ```

  If you enter only SELECT and press Enter, ESSCMD prompts you for the first parameter, the application name (*appName*). After you enter the application name and press Enter, ESSCMD prompts you for the database name (*dbName*).

- Type the commands and all parameters, then press Enter.

  Using SELECT as the example, you would type:

  ```
  SELECT "Sample" "Basic";
  ```

Whichever method you use, the interactive prompt now reflects the application and database names. For example, the following prompt tells you that the Sample application and Basic database are selected:

```
aspen:Sample:Basic:User[1]->
```

In this case, you can enter other commands without the application or database name parameters that it normally requires.

Canceling Operations

While ESSCMD is running, you can cancel an asynchronous operation, such as a calculation, export, or restructure operation, by pressing and holding the Esc key until ESSCMD responds.

CAUTION: Do not pause or suspend your system while Essbase is processing a command. Pausing the system may prevent Essbase from correctly completing the command.

Using Script and Batch Files for Batch Processing

If you use a series of commands frequently or you must enter many commands to complete a task, consider automating the task with a script or batch file. These files are useful for batch data loads and complex calculations.

For syntax conventions when working in batch processing mode, see "Syntax Guidelines" on page 1258.

- A *script file* contains ESSCMD commands. You can run a script file from the operating system command line or from within an operating system batch file, and the script file is processed by ESSCMD. By default, an ESSCMD script file has a `*.SCR` file extension. You can use a different extension.
- A *batch file* is an operating system file that calls multiple ESSCMD scripts, and can also include operating system commands. You can use a batch file to run multiple sessions of ESSCMD. You can run a batch file on OLAP Server from the operating system prompt; the file is processed by the operating system. On Windows NT, batch files have `*.BAT` file extensions.

Note: On UNIX, a batch or script file is written as a shell script. A shell script usually has the file extension `.sh` (Bourne or Korn shell) or `.csh` (C shell).

When you run a script or batch file, ESSCMD executes the commands in order until it reaches the end of the file.

Existing script or batch files might be affected by changes to some commands. To ensure that your files work in Essbase 6, check on your use of changed or deleted commands. See the *Essbase Installation Guide* for information about new and changed commands.

Writing Script Files

ESSCMD script files automate an often-used or lengthy series of commands. Each script file must be a complete ESSCMD session, with login, application and database selection, logout, and termination commands.

➤ To define a script file:

1. Enter ESSCMD commands in any editor that saves data in ASCII text.
2. Save the file with the `.SCR` ESSCMD script file extension.

   For example, the following script file, `TEST.SCR`, was created in Notepad:

```
LOGIN "localhost" "User1" "password";
SELECT "Sample" "Basic";
GETDBSTATE
EXIT;
```

When run from the operating system command line, this script logs User1 into the Essbase localhost server, selects the Sample application and Basic database, gets database statistics, and quits the ESSCMD session.

Running Script Files

➤ To run script files in ESSCMD:

1. Enter the following command at the operating system prompt:

   ```
   ESSCMD scriptFileName.SCR
   ```

2. Replace *scriptFileName* with the name of the script file. For example, type the following if the script file is in the current directory:

   ```
   ESSCMD TEST.SCR
   ```

3. If the script file is in another directory, include the path. For example:

   ```
   ESSCMD C:\WORK\SCRIPTS\TEST.SCR (an absolute path on
   Windows NT)
   ```

   or

   ```
   ESSCMD..\SCRIPTS\TEST.SCR (a relative path on Windows NT)
   ```

Handling Command Errors in a Script File

ESSCMD’s error-handling features provide error checking and handling for your script files. You can write error-handling commands into your script file to check for errors and, if necessary, branch to an appropriate error-handling response.

After each ESSCMD command is executed, a number is stored in an internal buffer. If the command executes successfully, `0` is returned to the buffer; If the command is unsuccessful, the error number is stored in the buffer; this is called *non-zero status*.

For error checking within an ESSCMD script file, ESSCMD provides the following error-handling commands:

- IFERROR checks the previously executed command for a non-zero return status (failure to execute). If the status is not zero, processing skips all subsequent commands and jumps to a user-specified point in the file, where it resumes. The script file can branch to an error-handling routine or the end of the file.
- RESETSTATUS reverts all saved status values to 0 (zero) in preparation for more status checking.
- GOTO forces unconditional branching to a user-specified point in the file, whether or not an error occurred.

In this `LOAD.SCR` example file, if the LOADDATA command does not execute successfully, ESSCMD branches to the end of the file to avoid attempting to calculate and run a report script on the empty database.

```
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

EXIT;
```

Note: You can use the OUTPUT command to log errors to a text file.

For the syntax and usage of ESSCMD error commands, see the online *Technical Reference*, located in the `ESSBASE/DOCS` directory.

Sample Script Files

The following script files demonstrate common Essbase batch operations. All samples are based on the Sample Basic database that comes with your Essbase program. The scripts for these examples are available in your \ARBORPATH\\app\\sample\\basic directory. On UNIX systems, the examples are available from `/home/hyperion/essbase/app/Sample/Basic`.

Sample Script: Importing and Calculating Data

Suppose you need a file that executes the following actions:

- Logs in to OLAP Server.
- Selects an application and database.
- Prevents other users from logging on and making changes to the database.
- Imports data from a text file.
- Calculates the database.
- Exits ESSCMD.

The following script file does the job:

```
LOGIN "Poplar" "TomT" "Password";
SELECT "Sample" "Basic";
DISABLELOGIN;
IMPORT 2 "ACTUALS" 4 "Y" 2 "ACTUAL" "N";
CALCDEFAULT;
ENABLELOGIN;
EXIT;
```

On Windows, this script file, `SAMPLE1.SCR`, is available in the `\ESSBASE\APP\SAMPLE\BASIC` directory. On UNIX platforms, `SAMPLE1.SCR` is in the `/essbase/app/Sample/Basic` directory.

Sample Script: Building Dimensions and Importing/Calculating Data from a SQL Source

Suppose you need a script file that executes the following actions:

- Logs in to OLAP Server
- Selects an application and database
- Prevents other users from logging on and making changes to the database
- Updates the outline from an SQL data source
- Imports data from SQL
- Calculates the database
- Exits ESSCMD

The following script file does the job:

```
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;
```

On Windows, this script file, `SAMPLE2.SCR`, is available in the `\ESSBASE\APP\SAMPLE\BASIC` directory. On UNIX systems, `SAMPLE2.SCR` is in the `/essbase/app/Sample/Basic` directory.

Sample Script: Scheduling Report Printing

Suppose you need a file that executes the following actions:

- Logs in to OLAP Server
- Selects an application and database
- Assigns reports that output to files for later printing
- Exits ESSCMD

The following script file does the job:

```
LOGIN "Poplar" "TomT" "Password";
SELECT "Sample" "Basic";
RUNREPT 2 "REP1" "REP1.OUT";
RUNREPT 2 "REP2" "REP2.OUT";
RUNREPT 2 "REP3" "REP3.OUT";
EXIT;
```

On Windows, this script file, `SAMPLE3.SCR`, is available in the `\ESSBASE\APP\SAMPLE\BASIC` directory. On UNIX platforms, `SAMPLE3.SCR` is in the `/essbase/app/Sample/Basic` directory.

Writing Batch Files

You can write a batch file that runs one or more ESSCMD scripts, and includes operating system commands. See your operating system instructions to learn the syntax for writing batch files.

For an example of a batch file, see “Handling Command Errors in a Script File” on page 1267.

Handling Command Errors in Batch Files

For the operating system batch file, you can use ESSCMD command return values to control the flow of scripts that the batch file executes.

An ESSCMD program returns an integer value upon exiting. This value represents the status of the last executed command, usually whether the command succeeded or failed. You can set up your batch file to test for this value, and if the test fails, branch to an error-handling response. This process is similar to creating a script file. For information about handling errors in script files, see “Handling Command Errors in a Script File” on page 1267.

For example, a batch file could contain three scripts: an ESSCMD batch file that loads data, a calculation script that performs calculations on the data, and a report script that reports on the results of the calculation. If the load batch file fails, the calculations and reporting also fail. In this case, it would be best to stop the batch file after the failure of the load file and correct the error that caused the failure before going on. If your batch file tests for the return value of the load process, and this return value indicates failure, the batch file can jump to the end of the file and stop or execute some other error-handling procedure, rather than attempting to calculate data that did not load.

The following example shows an NT operating system batch file and the contents of one of the ESSCMD scripts it runs, `LOAD.SCR`. Because error-checking requirements vary, the syntax in this example may not correspond to that of your operating system. See your operating system documentation for error checking in batch files.

An operating system batch file could contain commands like this:

```
ESSCMD LOAD.SCR
If not %errorlevel%==goto Error
ESSCMD CALC.SCR
If not %errorlevel%==goto Error
ESSCMD REPORT.SCR
If not %errorlevel%==goto Error
Echo All operations completed successfully
EXIT

:Error
Echo There was a problem running the script
```

Part

X

Optimizing and Troubleshooting Essbase

This part describes how to optimize and troubleshoot Essbase:

- Chapter 46, “Monitoring Performance,” describes how to use diagnostic tools to analyze and solve problems, check Hyperion Essbase configuration, and analyze Hyperion Essbase performance.
- Chapter 47, “Improving Essbase Performance,” supplies a list of settings and suggested and default values. These settings all affect performance.
- Chapter 48, “Optimizing Essbase Caches,” describes how to size the Hyperion Essbase index cache, data file cache, and data cache.
- Chapter 49, “Optimizing Database Restructuring,” explains the conditions under which Hyperion Essbase must perform database restructuring, and how you can design your outline to minimize this.
- Chapter 50, “Optimizing Data Loads,” describes how to debug problems with data loads and how to optimize your data loads so that Hyperion Essbase can load the data more quickly.
- Chapter 51, “Optimizing Calculations,” describes how to make your calc scripts execute more quickly.
- Chapter 52, “Optimizing with Intelligent Calculation,” describes how to use intelligent calculation to make your calc scripts execute more quickly.
- Chapter 53, “Optimizing Reports and Other Types of Retrieval,” describes ways to generate your reports more quickly, and how to optimize data extraction.
- Appendix B, “Error Handling and Troubleshooting for Essbase,” provides some suggestions for how to identify problems and correct them.

Note: For information about performance and partitions, see these resources:

- "Performance Considerations for Replicated Partitions" on page 322
- "Performance Considerations for Transparent Partitions" on page 330

Chapter

46

Monitoring Performance

This chapter describes the Essbase OLAP Server diagnostic information and tells you how to display that information. Use this information to monitor Essbase configuration or performance.

This chapter contains the following sections:

For information about logs, see Chapter 43, "Using Essbase Logs." For information about error messages, see Appendix B, "Error Handling and Troubleshooting for Essbase."

Finding Diagnostic Information

Essbase provides information dialog boxes at the application, server, and database level. Use Application Manager to view important performance information before completing any of these tasks:

- Preparing to migrate data
- Adding users
- Analyzing performance problems
- Performing other administrative tasks

The following sections provide instructions for how to open the dialog boxes from Application Manager, and lists other ways to obtain the same information if other ways are available. For a quick reference table to help you find specific information, see "Quick Reference to Diagnostic Information" on page 1294.

Remember these hints as you work with the information dialog boxes for server, application, and database:

- You cannot use these dialog boxes to change settings; the purpose of the dialog box is to help you identify activities or operations that may be affecting performance.
- Essbase displays information on a snapshot basis; to see the latest information, click the Refresh button for a new snapshot. The Refresh button, if it is displayed in a dialog box, refreshes every tab in the dialog box, not just the current tab.
- Many of the Essbase Application Manager tasks can be accomplished using ESSCMD, MaxL, or Essbase Administration Services. This document lists the related ESSCMD and MaxL tasks and tells you how to get more information. You can also read Chapter 45, "Automating the Production Environment" for quick-start information about using ESSCMD.
- For detailed information about a dialog box, display it and press the Help button.
- For information about server, application, and outline logs, see "Understanding Server and Application Logs" on page 1208.

Viewing Server Information

The Server Information dialog box displays information about server license, configuration, operating system, disk drives, and applications for the server where you are connected.

➤ To open the Server Information dialog box, use this procedure:

1. Open Application Manager and connect to a server.
2. From the Application Manager menu bar, select Server Information. A dialog box is displayed with several tabs.

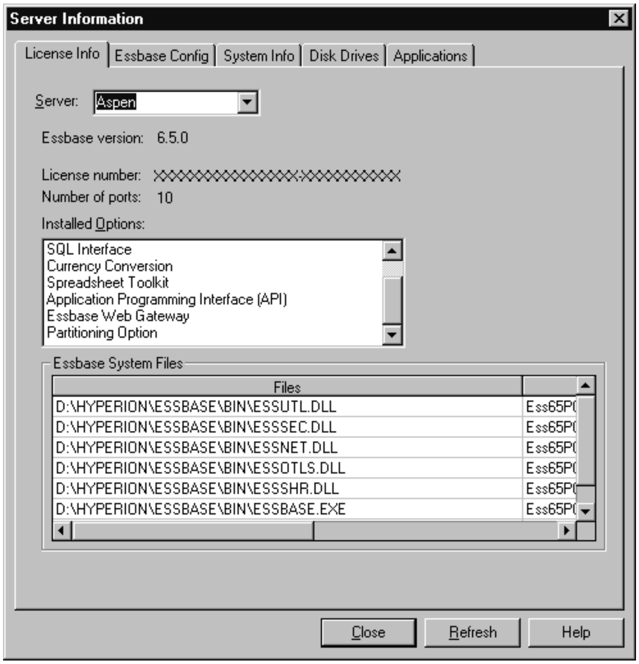
Each of these sections describes one of the tabs:

- "Viewing License Information" on page 1278
- "Viewing Configuration Information" on page 1279
- "Viewing System Information" on page 1280
- "Viewing Disk Drive Information" on page 1281
- "Viewing Application Status Information" on page 1282

Viewing License Information

In Application Manager, use the License Info tab of the Server Information dialog box to view information about the Essbase installation. The information presented includes version and license information, installed options, and a list of Essbase system files:

Figure 526: License Info Tab of the Server Information Dialog Box



Server: Aspen

Essbase version: 6.5.0

License number: XXXXXXXXXXXXXXXX-XXXXXXXXXXX

Number of ports: 10

Installed Options:

SQL Interface
Currency Conversion
Spreadsheet Toolkit
Application Programming Interface (API)
Essbase Web Gateway
Partitioning Option

Essbase System Files

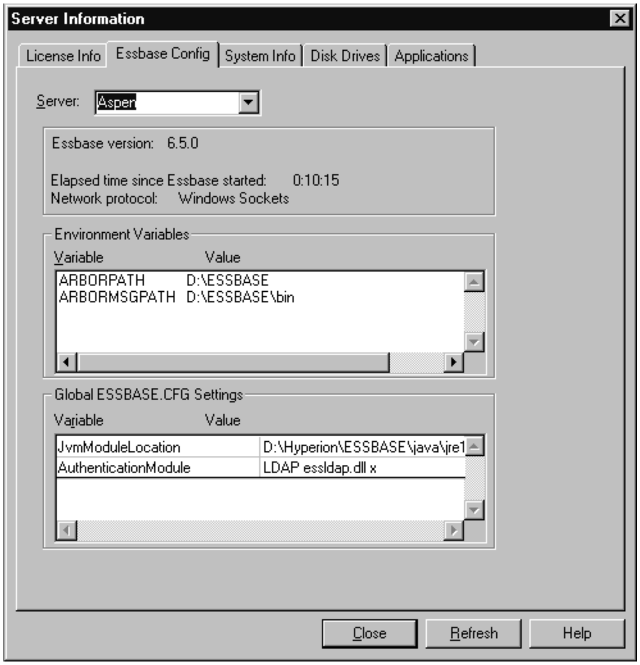
Files	
D:\HYPERION\ESSBASE\BIN\ESSUTL.DLL	Ess65P(
D:\HYPERION\ESSBASE\BIN\ESSSEC.DLL	Ess65P(
D:\HYPERION\ESSBASE\BIN\ESSNET.DLL	Ess65P(
D:\HYPERION\ESSBASE\BIN\ESSOTLS.DLL	Ess65P(
D:\HYPERION\ESSBASE\BIN\ESSSHR.DLL	Ess65P(
D:\HYPERION\ESSBASE\BIN\ESSBASE.EXE	Ess65P(

You can also select the License tab from the OLAP Server Properties window in Essbase Administration Services. For more information, see *Essbase Administration Services Online Help*.

Viewing Configuration Information

In Application Manager, use the Essbase Config tab of the Server Information dialog box to see information about Essbase status and configuration:

Figure 527: Essbase Config Tab of the Server Information Dialog Box



Server: Aspen

Essbase version: 6.5.0

Elapsed time since Essbase started: 0:10:15
Network protocol: Windows Sockets

Environment Variables

Variable	Value
ARBORPATH	D:\ESSBASE
ARBORMSGPATH	D:\ESSBASE\bin

Global ESSBASE.CFG Settings

Variable	Value
JvmModuleLocation	D:\Hyperion\ESSBASE\java\jre1
AuthenticationModule	LDAP essldap.dll x

- The Environment Variables list box displays operating system environment variables, as defined during installation. You can use this list to verify the paths set by environment variables.
- The Global `ESSBASE.CFG` Settings list box displays `ESSBASE.CFG` server file settings, if you have created an `ESSBASE.CFG` file. If you have not, the Global `ESSBASE.CFG` Settings list box is blank.

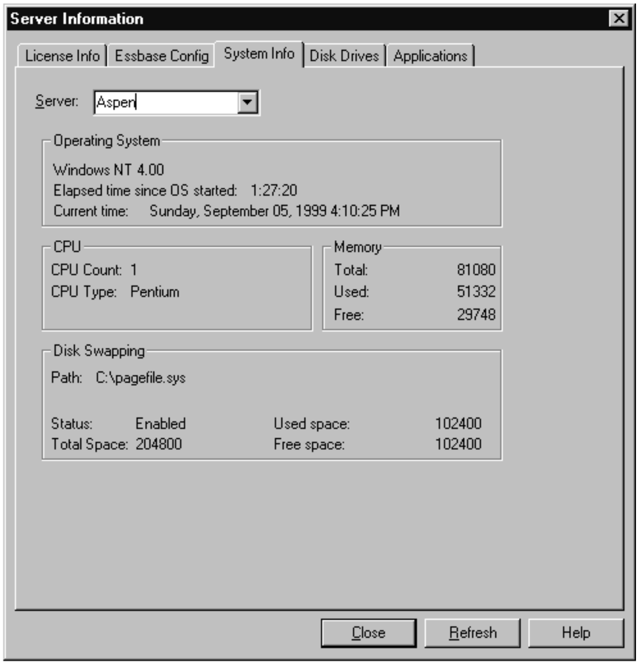
Note: See the *Technical Reference* in the `docs` directory for information on how to create an `ESSBASE.CFG` file.

You can also see information about Essbase status and configuration using Essbase Administration Services. For more information, see *Essbase Administration Services Online Help*.

Viewing System Information

Use the System Info tab of the Server Information dialog box to review information about the operating system and about resource usage. You can use this information to determine whether machine or operating system constraints are affecting Essbase performance.

Figure 528: System Info Tab of the Server Information Dialog Box



Server: Aspen

Operating System
Windows NT 4.00
Elapsed time since OS started: 1:27:20
Current time: Sunday, September 05, 1999 4:10:25 PM

CPU
CPU Count: 1
CPU Type: Pentium

Memory
Total: 81080
Used: 51332
Free: 29748

Disk Swapping
Path: C:\pagefile.sys
Status: Enabled
Total Space: 204800
Used space: 102400
Free space: 102400

For example, if Essbase is running too slowly, check the Memory and Disk Swapping groups to see how much free space is available. If the Free space value in Disk Swapping is very low, you could take one of the following actions, depending your platform:

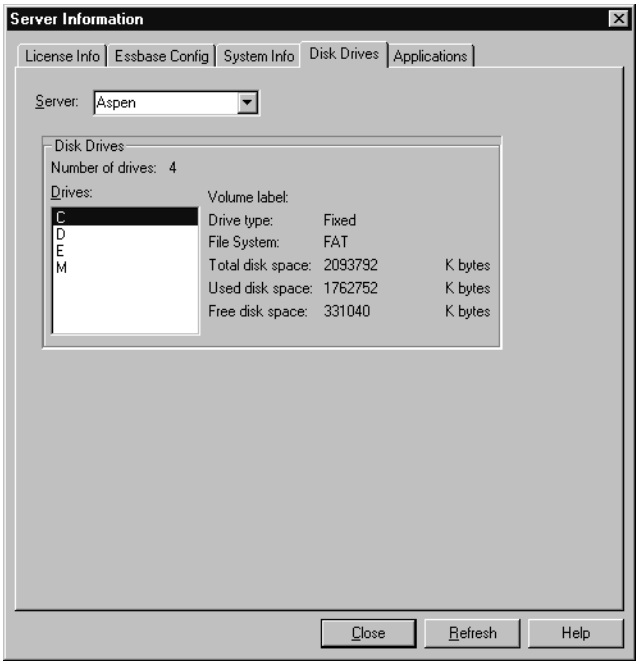
- On Windows platforms, increase the maximum virtual memory or add more volumes for disk swapping.
- On UNIX platforms, increase swap space or add another swapping device.

You can also review information about the operating system and about resource usage with Essbase Administration Services. For more information, see *Essbase Administration Services Online Help*.

Viewing Disk Drive Information

The Disk Drives tab of the Server Information dialog box contains information about disk drive types, disk drive usage, and file system types:

Figure 529: Disk DriveT tab of the Server Information Dialog Box



Server: Aspen

Disk Drives

Number of drives: 4

Drives:

C
D
E
M

Volume label:		
Drive type:	Fixed	
File System:	FAT	
Total disk space:	2093792	K bytes
Used disk space:	1762752	K bytes
Free disk space:	331040	K bytes

Use this information to determine whether lack of disk space or incompatibility of file system types is affecting Essbase performance.

For example, if information on the System Info tab indicates that swap space is low, you can use the Disk Drives tab to see which drives have space available.

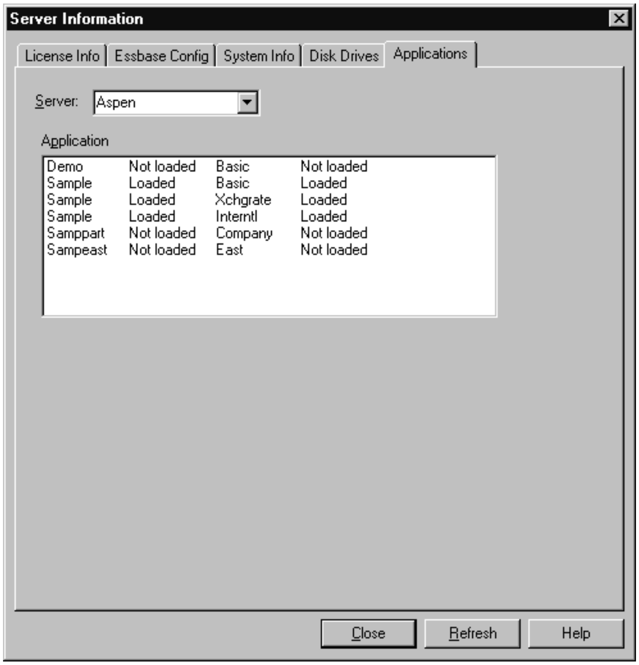
Also, if you want to allocate space on another drive, use the Disk Drives tab to see which drives have space available. See “Specifying Disk Volumes” on page 1116 for information about disk spanning using the Disk Volumes setting.

You can also see information about disk drive types, disk drive usage, and file system types with Essbase Administration Services. For more information, see *Essbase Administration Services Online Help*.

Viewing Application Status Information

Use the Applications tab of the Server Information dialog box to see which applications and databases are loaded.

Figure 530: Applications Tab of the Server Information Dialog Box



Server: Aspen

Application

Demo	Not loaded	Basic	Not loaded
Sample	Loaded	Basic	Loaded
Sample	Loaded	Xchgrate	Loaded
Sample	Loaded	Interntl	Loaded
Samppart	Not loaded	Company	Not loaded
Sampeast	Not loaded	East	Not loaded

Essbase lists only the applications that you are authorized to use.

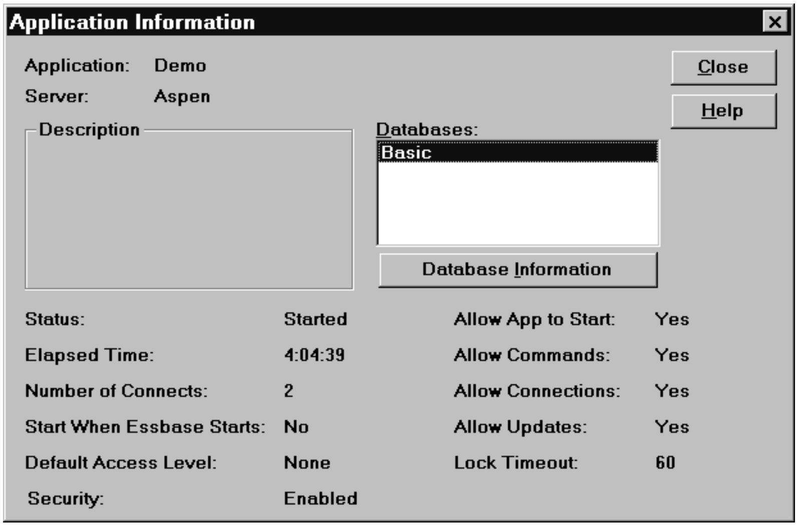
Viewing Application Information

Use the Application Information dialog box to identify which databases are running in the application and to check access, security, and start-up information.

➤ To display the Application Information dialog box, use this procedure:

1. Open the Application Manager and click on an application name to select it.
2. Select Application>Information from the menu bar. The Application Information dialog box is displayed:

Figure 531: Application Information Dialog Box



Application Information

Application: Demo

Server: Aspen

Description

Databases:
Basic

Status:	Started	Allow App to Start:	Yes
Elapsed Time:	4:04:39	Allow Commands:	Yes
Number of Connects:	2	Allow Connections:	Yes
Start When Essbase Starts:	No	Allow Updates:	Yes
Default Access Level:	None	Lock Timeout:	60
Security:	Enabled		

3. Click the Database Information button to display the Database Information dialog box. This is the same information described in “Viewing Database Information” on page 1284.

To customize your application, change the settings in the Application Settings dialog box, as described in Chapter 15, “Managing Security for Users and Applications.”

Tip: You can access application information without using Application Manager:

Tool	Instructions	For More Information
Essbase Administration Services	Use the Application Properties window.	*Essbase Administration Services Online Help*
MaxL	**display application**	*Technical Reference* in the `docs` directory
ESSCMD	GETAPPSTATE GETPERFSTATS	*Technical Reference* in the `docs` directory

Viewing Database Information

The Database Information dialog box provides information about database storage, database statistics, and lock contention.This information may help you identify activities or operations that affect performance.

➤ To see the Database Information dialog box, use this procedure:

1. From the Application Manager, connect to a server.
2. Select an application and database from the lists that is displayed.
3. From the Application Manager menu bar, select Database > Information. The Database Information dialog box appears, with information sorted on several tabs. The relevant server, application, and database name is displayed on every tab.

Each of these sections describes one of the tabs:

- “Viewing General Database Information” on page 1286
- “Viewing Database Storage Information” on page 1287
- “Viewing Currency Database Information” on page 1288
- “Viewing Database Statistics” on page 1289
- “Viewing Run-Time Information” on page 1291
- “Viewing Database File Information” on page 1292
- “Viewing a Record of Database Modifications” on page 1293

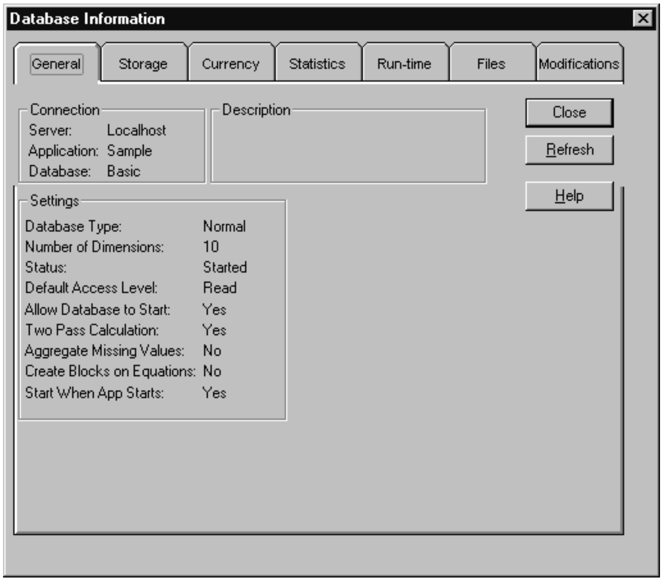
These dialog boxes are for viewing only. For information on changing database settings, see “Customizing Essbase Kernel Settings” on page 1108.

Tip: You can access database information without using Application Manager:

Tool	Instructions	For More Information
Essbase Administration Services	Use the Database Properties window.	*Essbase Administration Services Online Help*
MaxL	**display database**	*Technical Reference* in the `docs` directory
ESSCMD	GETDBINFO	

Viewing General Database Information

Use the General tab of the Database Information dialog box to check database status and start-up information.



Connection

Server: Localhost
Application: Sample
Database: Basic

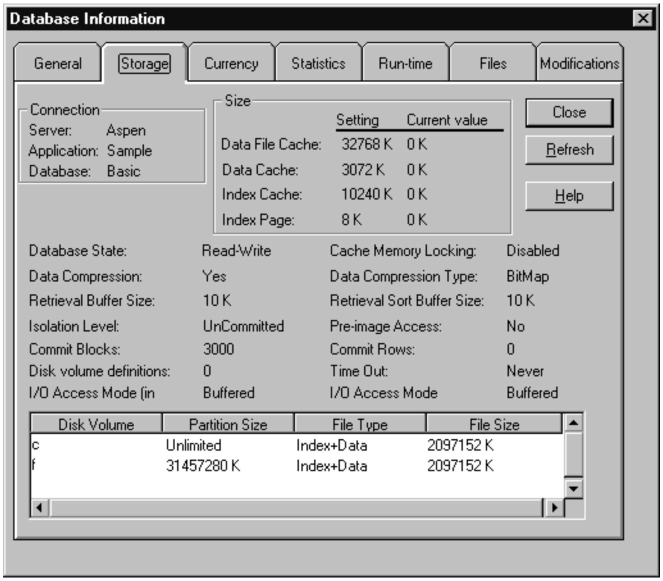
Description

Settings

Database Type:	Normal
Number of Dimensions:	10
Status:	Started
Default Access Level:	Read
Allow Database to Start:	Yes
Two Pass Calculation:	Yes
Aggregate Missing Values:	No
Create Blocks on Equations:	No
Start When App Starts:	Yes

Viewing Database Storage Information

Use the Storage tab of the Database Information dialog box to check data storage information, including cache sizes and the data compression setting.



Connection
Server: Aspen
Application: Sample
Database: Basic

Size

	Setting	Current value
Data File Cache:	32768 K	0 K
Data Cache:	3072 K	0 K
Index Cache:	10240 K	0 K
Index Page:	8 K	0 K

Database State:	Read-Write	Cache Memory Locking:	Disabled
Data Compression:	Yes	Data Compression Type:	BitMap
Retrieval Buffer Size:	10 K	Retrieval Sort Buffer Size:	10 K
Isolation Level:	UnCommitted	Pre-image Access:	No
Commit Blocks:	3000	Commit Rows:	0
Disk volume definitions:	0	Time Out:	Never
I/O Access Mode (in	Buffered	I/O Access Mode	Buffered

Disk Volume	Partition Size	File Type	File Size
c	Unlimited	Index+Data	2097152 K
f	31457280 K	Index+Data	2097152 K

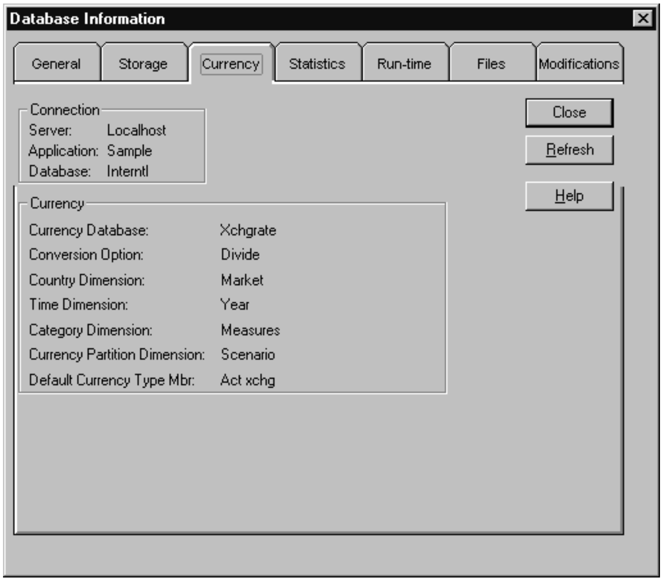
Tip: You can access database storage information without using Application Manager:

Tool	Instructions	For More Information
Essbase Administration Services	Use the Database Properties window. Select the Storage tab.	*Essbase Administration Services Online Help*
MaxL	**display database**	*Technical Reference* in the `docs` directory
ESSCMD	GETDBSTATE	

Viewing Currency Database Information

Use the Currency tab of the Database Information dialog box to examine the configuration of the currency database associated with the selected database, if the selected database is linked to a currency database.

Note: Currency conversion is an optional module, licensed separately. It may not be installed on your server.



Connection
Server: Localhost
Application: Sample
Database: Interntl

Currency
Currency Database: Xchgrate
Conversion Option: Divide
Country Dimension: Market
Time Dimension: Year
Category Dimension: Measures
Currency Partition Dimension: Scenario
Default Currency Type Mbr: Act xchg

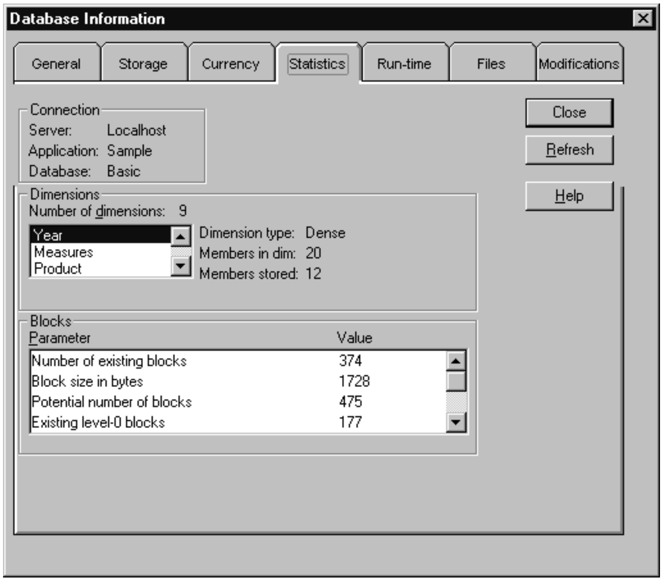
Tip: You can access Currency Database information without using Application Manager:

Tool	Instructions	For More Information
Essbase Administration Services	Use the Database Properties window. Select the Currency tab.	*Essbase Administration Services Online Help*
MaxL	**display database**	*Technical Reference* in the `docs` directory
ESSCMD	GETCRDBINFO	

Viewing Database Statistics

Use the Statistics tab of the Database Information dialog box to view dimension information and to see characteristics of data blocks in the selected database.

Figure 532: Statistics Tab of the Database Information Dialog Box



Connection

Server: Localhost

Application: Sample

Database: Basic

Dimensions

Number of dimensions: 9

Year

Measures

Product

Dimension type: Dense

Members in dim: 20

Members stored: 12

Blocks

Parameter	Value
Number of existing blocks	374
Block size in bytes	1728
Potential number of blocks	475
Existing level-0 blocks	177

To check on block density and usage, in the Blocks list, scroll down to see the Block Density and Percentage Of Maximum Blocks Existing fields.

Block density is the number of cells containing data in a datablock expressed as a percentage of total cells in the datablock. Percentage Of Maximum Blocks Existing Is A Percentage Comparison between existing blocks and potential blocks.

If you have a low block density and a high percentage of maximum blocks, you may want to increase the number of dense dimensions. See Chapter 3, “Basic Architectural Elements,” for more information about dense and sparse dimensions.

To check block size, look at the “Block size in bytes” field. This field reports the size of the block without data compression, whether or not you have data compression enabled. Ideally, this value should be between eight kilobytes and 100 kilobytes. To alter block size, you need to change at least one dimension from dense to sparse or from sparse to dense .

The Compression Ratio indicates the ratio of compressed block size to expanded block size. For more information about data compression, see “Data Compression” on page 1124.

If the database is not started when you choose the Statistics tab, prompts you to start the database.

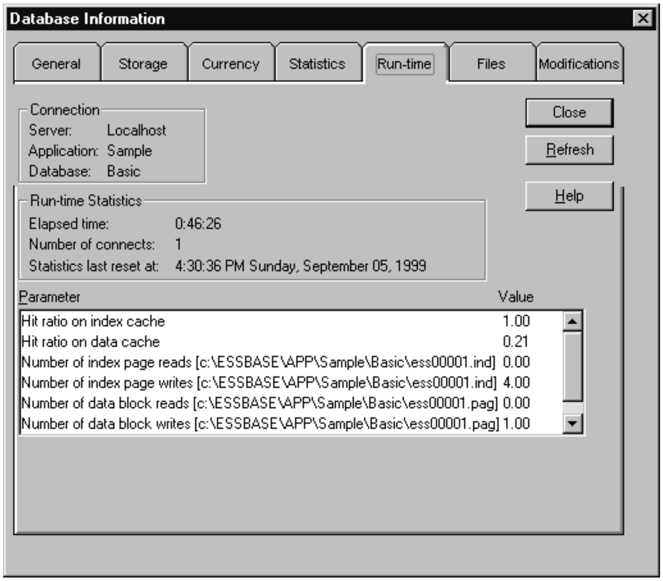
Tip: You can access database statistics information without using Application Manager:

Tool	Instructions	For More Information
Essbase Administration Services	Click the Statistics tab from the Database Properties window.	*Essbase Administration Services Online Help*
MaxL	**display database**	*Technical Reference* in the `docs` directory
ESSCMD	GETDBSTATS	

Viewing Run-Time Information

The Run-time tab of the Database Information dialog box displays run-time information such as lock contention, cache size problems (buffer shortage), hit ratio on index searches (percentage of searches that did not involve retrieving from disk), and the number of read and write operations.

Figure 533: Run-Time Tab of the Database Information Dialog Box



Connection
Server: Localhost
Application: Sample
Database: Basic

Run-time Statistics
Elapsed time: 0:46:26
Number of connects: 1
Statistics last reset at: 4:30:36 PM Sunday, September 05, 1999

Parameter	Value
Hit ratio on index cache	1.00
Hit ratio on data cache	0.21
Number of index page reads [c:\ESSBASE\APP\Sample\Basic\ess00001.ind]	0.00
Number of index page writes [c:\ESSBASE\APP\Sample\Basic\ess00001.ind]	4.00
Number of data block reads [c:\ESSBASE\APP\Sample\Basic\ess00001.pag]	0.00
Number of data block writes [c:\ESSBASE\APP\Sample\Basic\ess00001.pag]	1.00

If the database is not started when you choose the Run-time tab, prompts you to start the database.

To find out if your index cache size is large enough, for example, you could check the Hit Ratio On Index Cache Field (You may need to scroll down in the Parameters list box to see it). This value indicates the success rate of Essbase in locating index data in memory without having to retrieve it from the disk. Ideally, this value should be near 100 percent (1.0). If the hit ratio is very low, the index cache (buffer) is too small.

For more information about caches and hit ratios, see “Fine Tuning Cache Settings” on page 1339. For information on sizing the index cache, see “Sizing the Index Cache” on page 1320.

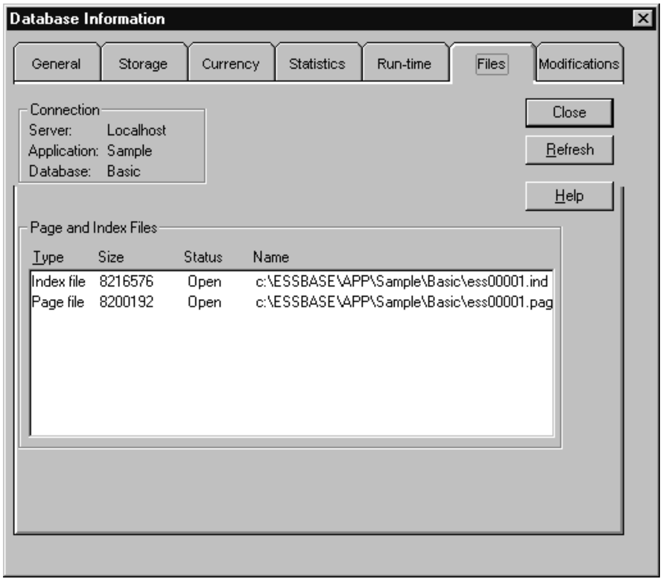
Tip: You can access database statistics information without using Application Manager:

Tool	Instructions	For More Information
Essbase Administration Services	Use the Database Properties window	*Essbase Administration Services Online Help*
MaxL	**Display Database**	*Technical Reference* in the `docs` directory
ESSCMD	GETDBINFO	

Viewing Database File Information

The Files tab of the Database Information dialog box to view information about the index and data (`.PAG`) files associated with the selected application and database:

Figure 534: Files Tab of the Database Information Dialog Box



Connection

Server: Localhost
Application: Sample
Database: Basic

Page and Index Files

Type	Size	Status	Name
Index file	8216576	Open	c:\ESSBASE\APP\Sample\Basic\ess00001.ind
Page file	8200192	Open	c:\ESSBASE\APP\Sample\Basic\ess00001.pag

Use this tab to see where index and data (`.PAG`) files are stored and to see whether the index and page files span multiple volumes.

Check the Applications tab of the Server Information dialog box (see "Viewing Application Status Information" on page 1282) to see if the database is loaded. If the database in the Database field is loaded, the index files and data files should display the status Open.

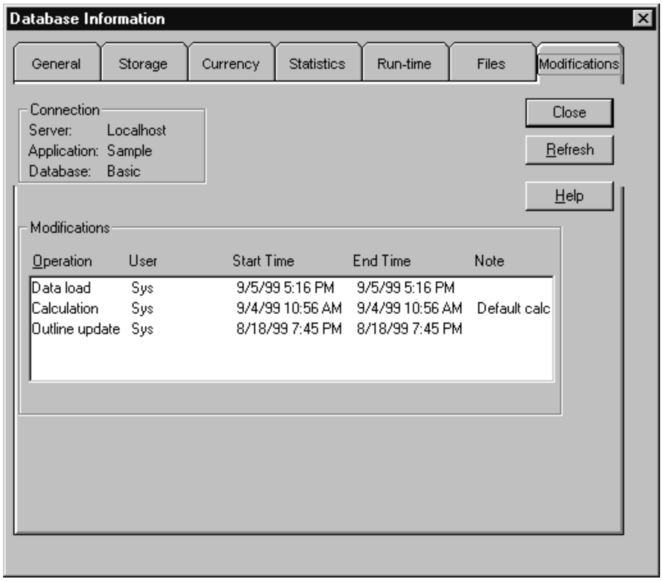
If the files related to the open database display the status Closed, click the Refresh button. If the files still display the status Closed, a problem may have occurred when the database started. Check your application and server logs (see "Understanding the Contents of the Application Log" on page 1212 and "Understanding the Contents of the OLAP Server Log" on page 1209) to see whether you can determine the problem. If you cannot, call the technical support department of your software provider.

You can also use the Database Properties window in Essbase Administration Services to view database file information. For more information, see *Essbase Administration Services Online Help*.

Viewing a Record of Database Modifications

Use the Modifications tab of the Database Information dialog box to see information about the last successful data load, about calculation, and about outline update operations for the current application.

Figure 535: Modifications Tab of the Database Information Dialog Box



Connection
Server: Localhost
Application: Sample
Database: Basic

Modifications

Operation	User	Start Time	End Time	Note
Data load	Sys	9/5/99 5:16 PM	9/5/99 5:16 PM	
Calculation	Sys	9/4/99 10:56 AM	9/4/99 10:56 AM	Default calc
Outline update	Sys	8/18/99 7:45 PM	8/18/99 7:45 PM	

Start Time and End Time include the necessary preparation, such as locking the data, that Essbase does prior to the operation.

To see the duration of the actual operation itself, see the “Elapsed time” entry in the application log. “Understanding the Contents of the Application Log” on page 1212 describes this log and tells you where to find it.

You can also use the Database Properties window in Essbase Administration Services to view a record of database modifications. For more information, see *Essbase Administration Services Online Help*.

Quick Reference to Diagnostic Information

Table 76 provides detailed Application Manager procedures for accessing information about server, applications, and databases that are commonly used to diagnose performance or other issues.

Table 76: Quick Reference to Diagnostic Information

Task	Instructions
To see a list of the **servers you are currently connected to**	Select Server > Information to display the Server Information dialog box. Select the License Info tab to view the Server list box.
To view the **version** of Essbase that is currently running on the server	Select Server > Information to display the Server Information dialog box. Select the License Info tab to view the Essbase version information.
To check the **license number** of your copy of Essbase	Select Server > Information to display the Server Information dialog box. Select the License Info tab to view the License number item.
To see the **total number of ports** that can connect to the server	Select Server > Information to display the Server Information dialog box. Select the License Info tab to view the Number of ports item.
To see a **list of all features** in place when Essbase was installed	Select Server > Information to display the Server Information dialog box. Select the License Info tab to view the features list under Installed Options.
To see a **list of the system files** currently in the server memory, as well as the locations and version numbers for the files	Select Server > Information to display the Server Information dialog box. Select the License Info tab to view the Hyperion Essbase System Files list box.

Table 76: Quick Reference to Diagnostic Information (Continued)

Task	Instructions
To see **elapsed time since Hyperion Essbase started**	Select Server > Information to display the Server Information dialog box. Select the Hyperion Essbase Config tab to view the Elapsed time since Essbase started item.
To check the **type of network protocol** that the server is using	Select Server > Information to display the Server Information dialog box. Select the Essbase Config tab to see the Network protocol.
To view **environment variables**, including source path information	Select Server > Information to display the Server Information dialog box. Select the Essbase Config tab to see the Environment Variables list box.
To view the **version of the operating system** on which Hyperion Essbase is running	Select Server > Information to display the Server Information dialog box. Select the System Info tab to view the Operating System item.
To see **the time that has elapsed since the operating system was started or rebooted** (in hours:minutes:seconds format)	Select Server > Information to display the Server Information dialog box. Select the System Info tab to view the Elapsed time since OS started item.
To see **the time and date** the server information was retrieved, according to the server	Select Server > Information to display the Server Information dialog box. Select the System Info tab to view the Current time item.
To see the **number and type of CPUs** in the system on which the server is operating	Select Server > Information to display the Server Information dialog box. Select the System Info tab to view the CPU group.
To see the **amount of available memory** on the operating system	Select Server > Information to display the Server Information dialog box. Select the System Info tab to view the Memory group.
To see the **disk swap path**	Select Server > Information to display the Server Information dialog box. Select the System Info tab and look in the Disk Swapping box to view the Path item.
To check if **disk swapping is enabled**	Select Server > Information to display the Server Information dialog box. Select the System Info tab and look in the Disk Swapping box to view the Status item.
To see the **total swap space**	Select Server > Information to display the Server Information dialog box. Select the System Info tab and look in the Disk Swapping box to view Total Space.

Table 76: Quick Reference to Diagnostic Information (Continued)

Task	Instructions
To see the **swap file size**	Select Server > Information to display the Server Information dialog box. Select the System Info tab and look in the Disk Swapping box to view Used space.
To see **available swap space**	Select Server > Information to display the Server Information dialog box. Select the System Info tab and look in the Disk Swapping box to view Free space.
To view the **drive type** (fixed, removable, RAM, remote, or unknown)	Select Server > Information to display the Server Information dialog box. Select the Disk Drives tab and select a drive from the Drives list box. The Drive type item displays the drive type.
To view the **file system used by a drive** (FAT, HPFS, NTFS, etc.)	Select Server > Information to display the Server Information dialog box. Select the Disk Drives tab and select a drive from the Drives list box. The File System item displays the type of file system used by the drive.
To view the **total amount of space on a drive**	Select Server > Information to display the Server Information dialog box. Select the Disk Drives tab and select a drive from the Drives list box. The Total disk space item displays the total disk space.
To view the **amount of space being used** on a drive	Select Server > Information to display the Server Information dialog box. Select the Disk Drives tab and select a drive from the Drives list box. The Used disk space item displays the amount of space currently being used by the selected drive.
To view the **amount of space available** on a drive	Select Server > Information to display the Server Information dialog box. Select the Disk Drives tab and select a drive from the Drives list box. The Free disk space item displays the amount of space currently available on the selected drive.

Table 76: Quick Reference to Diagnostic Information (Continued)

Task	Instructions
To see **which applications and databases are loaded**	Select Server > Information to display the Server Information dialog box. Select the Applications tab. The Application list displays the following information for each application/database combination: • Application name (only those you are authorized to use) • The status of the listed applications (Loaded, Not Loaded, Loading, and Unloading) • Database name • The status of the database (Loaded, Not Loaded, Loading, and Unloading)
To view your **database connections**	Select Database > Information to display the Database Information dialog box. Select the General tab and look in the Connection box for the name of the server, application, and database currently loaded.
To view your **database type**	Select Database > Information to display the Database Information dialog box. Select the General tab and look in the Settings box for the database type.
To view the **number of dimensions** in a database	Select Database > Information to display the Database Information dialog box. Select the General tab and look in the Settings box for the number of dimensions.
To view the **status of a database**	Select Database > Information to display the Database Information dialog box. Select the General tab and look in the Settings box for the status (for example, Stopped).
To view the **default access level to a database**	Select Database > Information to display the Database Information dialog box. Select the General tab and look in the Settings box for the default access level.
To see **whether users may start the database**	Select Database > Information to display the Database Information dialog box. Select the General tab and look in the Settings box for the Allow Database to Start status.
To see **if your database aggregates missing values**	Select Database > Information to display the Database Information dialog box. Select the General tab and look in the Settings box for the Aggregate Missing Values status.

Table 76: Quick Reference to Diagnostic Information (Continued)

Task	Instructions
To see **if your database creates blocks on equations**	Select Database > Information to display the Database Information dialog box. Select the General tab and look in the Settings box for the Create Blocks on Equations status.
To see **if a database is set to start automatically** when an application is started	Select Database > Information to display the Database Information dialog box. Select the General tab. The Start When App Starts status is displayed in the Settings group
To view the **data file cache size** of a database (in kilobytes)	Select Database > Information to display the Database Information dialog box. Select the Storage tab and look in the Size box for the Setting and Current Value of the data file cache.
To view the **data cache size** of a database (in kilobytes)	Select Database > Information to display the Database Information dialog box. Select the Storage tab and look in the Size box for the Setting and Current value of the data cache.
To view the **index page size** of a database (in kilobytes)	Select Database > Information to display the Database Information dialog box. Index page size is fixed at 8K.
To view the **index cache size** of a database (in kilobytes)	Select Database > Information to display the Database Information dialog box. Select the Storage tab and look in the Size box for the Setting and Current value of the index cache.
To see **if the cache memory locking option is enabled** in a database	Select Database > Information to display the Database Information dialog box. Select the Storage tab and look at the Cache Memory Locking status.
To see **if the compression option is enabled** in a database	Select Database > Information to display the Database Information dialog box. Select the Storage tab and look at the Data Compression status.
To see **if a selected database is linked to a currency database**	Select Database > Information to display the Database Information dialog box. Select the Currency tab and look in the Currency box for the Currency Database. If it is the same name as the selected database, you are connected to the currency database itself.
To see **if the currency conversion option is on** in a currency database	Select Database > Information to display the Database Information dialog box. Select the Currency tab and look in the Currency box for the Conversion option.

Table 76: Quick Reference to Diagnostic Information (Continued)

Task	Instructions
To see **if a dimension is set to a specific country** in a currency database	Select Database > Information to display the Database Information dialog box. Select the Currency tab and look in the Currency box for the Country Dimension.
To view **the time dimension setting** in a currency database	Select Database > Information to display the Database Information dialog box. Select the Currency tab and look in the Currency box for the Time Dimension.
To view **the category dimension setting** in a currency database	Select Database > Information to display the Database Information dialog box. Select the Currency tab and look in the Currency box for the Category Dimension.
To view **the dimension setting of the currency partition**	Select Database > Information to display the Database Information dialog box. Select the Currency tab and look in the Currency box for the Currency Partition Dimension.
To see the **default type of currency** that a member is set to in a currency database	Select Database > Information to display the Database Information dialog box. Select the Currency tab and look in the Currency box for the Default Currency Type Mbr.
To view the **number of dimensions** in a database	Select Database > Information to display the Database Information dialog box. Select the Statistics tab and look in the Dimensions box for the Number of dimensions.
To view a **list of the dimensions** in a database	Select Database > Information to display the Database Information dialog box. Select the Statistics tab and look in the Dimensions box for a list of all the dimensions.
To see **if a dimension is designated as sparse or dense**	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. From the Dimensions box, select the specific dimension from the list. The Dimension type (either Sparse or Dense) is displayed.
To see the **number of members** in a dimension	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. From the Dimensions box, select the specific dimension from the list. The Members in dim item is displayed.
To see the **number of stored members** in a dimension	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. From the Dimensions box, select the specific dimension from the list. The Members stored item is displayed.

Table 76: Quick Reference to Diagnostic Information (Continued)

Task	Instructions
To view the **number of existing blocks** in a database	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. In the Blocks box, view the value for Number of existing blocks.
To view the **block size in bytes**	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. In the Blocks box, view the value of Block size in bytes. (Block size varies widely from block to block.)
To view the **potential number of blocks** in a database	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. In the Blocks box, view the value of Potential number of blocks.
To view the **number of existing level-0 blocks** in a database	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. In the Blocks box, view the value of Existing level-0 blocks.
To view the **number of existing upper-level blocks** in a database	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. In the Blocks box, view the value of Existing upper-level blocks.
To view **block density** in a database	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. In the Blocks box, view the value of Block density. (Density varies widely from block to block.)
To view a **comparison between the number of existing blocks and the maximum number of possible blocks** in a database	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. In the Blocks group, view the value of Percentage of maximum blocks existing.
To view the **compression ratio of blocks** in a database	Select Database > Information to display the Database Information dialog box. Select the Statistics tab. In the Blocks box, view the value of Compression Ratio.
To view the **run-time that has elapsed since the database was started**, according to the server (in hours:minutes:seconds format).	Select Database > Information to display the Database Information dialog box. Select the Run-time tab. In the Statistics box, view the value of Elapsed time.

Table 76: Quick Reference to Diagnostic Information (Continued)

Task	Instructions
To view the **number of users connected** to the database	Select Database > Information to display the Database Information dialog box. Select the Run-time tab. In the Statistics box, view the value of the Number of connects.
To view the **hit ratio on the data file, data, and index caches** (the success rate of Hyperion Essbase in locating data in the appropriate cache, without having to retrieve the data from disk)	Select Database > Information to display the Database Information dialog box. Select the Run-time tab. and view the values for Hit ratio on index cache, Hit ration on data cache, and Hit ratio on data file.
To view the **number of index page and data block read and writes** (the number of times Hyperion Essbase read index page or data block information from disk or wrote index page or data block information to disk).	Select Database > Information to display the Database Information dialog box. Select the Run-time tab. and view the values for Number of index page read and writes.
To view the **size and status of a data (page) file or an index file**	Select Database > Information to display the Database Information dialog box. Select the Files tab. In the Page and Index Files box, view the values for Index file or Page (data) file.
To view **details about modifications** to a database	Select Database > Information to display the Database Information dialog box. Select the Modifications tab. In the Modifications box, view the values for the Operation (type of modification), the User, the Start Time and the End Time.

Monitoring Applications from the Operating System

Each application that is loaded is an open task or process in the operating system. You can use the operating system to view application tasks or processes:

- On Windows platforms, the application is displayed in an Essbase server window. You can view application activities as they occur in this window. When the Agent starts an application, a new icon is displayed in the Taskbar. You can double-click the icon to view the server window.
- On UNIX platforms, the application server is often a background process. When the application starts, `ESSBASE` starts the ESSSVR process. In order to see activities, you can route all messages to a file with the `tail -f` *`log`* command, where *log* is the name of a file that you specify. You can also use these clients to view a snapshot of the server or application log:
    - Administration Services. See *Essbase Administration Services Online Help* for details about viewing, filtering, searching, and analyzing logs.
    - Application Manager. See *Essbase Application Manager Online Help* for details about viewing logs.

See "Understanding Server and Application Logs" on page 1208 for more information about these types of logs.

Object Locks

Essbase uses a check-out facility for server-based objects to ensure that no more than one user attempts to modify an object at one time. By default, whenever a user opens a server-based object, Essbase locks the object. If a second user attempts to open the same object, a message is displayed. The message indicates that the object is already locked and notes who has locked it.

Overriding a File Lock

You can override the lock error by unchecking Lock file in the Application Desktop window before attempting to open an object. This action essentially opens the file in read-only mode and you cannot save any changes you make.

Unlocking Objects

There are two ways to unlock a previously locked object:

- Save the locked file to the server.
- Select File > Unlock. This action unlocks a file that you previously locked but did not save to the server.

You can unlock an object only if you originally locked it. If you have supervisor authority you can unlock files that are locked by other users.

Tip: You can unlock an object without the Application Manager:

Tool	Instructions	For More Information
Essbase Administration Services	From Enterprise View, select the object to be unlocked. Right-click and select Unlock from the menu.	*Essbase Administration Services Online Help*
ESSCMD	UNLOCKOBJECT	*Technical Reference* in the `docs` directory

Note: File locking is not the same as block locking. For information about how Essbase locks data blocks, see Chapter 40, “Ensuring Data Integrity.”

Chapter 47

Improving Essbase Performance

You can improve Essbase performance with these basic techniques:

- "Using Basic Design for Optimization" on page 1305
- "Reset Database to Increase Performance" on page 1306
- "Database Settings Quick-Reference Table" on page 1306
- "Eliminating Fragmentation" on page 1311
- "Finding Other Optimization Information" on page 1313

Using Basic Design for Optimization

Use the following list to identify basic design issues that affect optimization outside this volume:

- For an introduction to basic concepts and how they relate to optimized performance, see Chapter 3, "Basic Architectural Elements."
- For hints about how to review a database design while it is still on paper, and how this analysis can aid optimization, see "Planning and Analyzing" on page 104.
- To understand how basic database outline issues affect performance, see "Designing an Outline to Optimize Performance" on page 124.

Reset Database to Increase Performance

You can periodically reset your database using the ESSCMD command RESETDB, and then reload it.Even if you reload your database very often, your main database files, `.pag` files, can grow unless you use RESETDB.

Database Settings Quick-Reference Table

You can customize Essbase for maximum performance, using database settings at the database level:

- Database Settings dialog box in Application Manager
- Database Properties window in Administration Services, ESSCMD, or MaxL.

Table 77 lists each Database > Storage setting in Application Manager or the equivalent in ESSCMD or MaxL that can affect performance, and describes how to adjust each setting.

For information on database settings in Essbase Administration Services, see *Essbase Administration Services Online Help*.

➤ Before you use the table with Application Manager, use this procedure:

1. Start Essbase.
2. Start Application Manager.
3. From Application Manager, select the Server > Connect menu command to connect to the appropriate server.
4. Selected the application and database whose performance you wish to improve.
5. Selected Database > Settings and display the Settings dialog box.

If you plan to use the ESSCMD or MaxL commands instead of Application Manager, see the *Technical Reference* in the `docs` directory for instructions.

The MaxL command for all of these is **alter database**. See the *MaxL Language Reference* for more information.

Note: If you are migrating your database, see the *Essbase Installation Guide* for information about the default settings after migration.

Table 77: Database Settings and Defaults

Setting	Application Manager Database > Setting	ESSCMD Command	Default Value and Comments
Index cache size	Storage	`SETDBSTATEITEM 12`	10 MB (default), 1 MB (minimum) Recommendation: Combined size of all ESS*n*.IND files if possible; otherwise, as large as possible. See "Sizing the Index Cache" on page 1320.
Data file cache size	Storage	`SETDBSTATEITEM 27`	32 MB (default), 8 MB (minimum) Recommendation: Combined size of all ESS*n*.PAG files if possible; otherwise, as large as possible. See "Sizing the Data File Cache" on page 1322.
Data cache size	Storage	`SETDBSTATEITEM 5`	3 MB (default and minimum) Recommendation: 0.125 * data file cache size value. However, some calculations require a larger data cache size. In general, if you must choose between allocating memory to the data file cache and allocating memory to the data cache, choose the data file cache. See "Sizing the Data Cache" on page 1324.
Index page size	Storage	`SETDBSTATEITEM 13`	Fixed size is 8 KB. You cannot change the index page size.

Table 77: Database Settings and Defaults (Continued)

Setting	Application Manager Database > Setting	ESSCMD Command	Default Value and Comments
Cache memory locking	Storage	`SETDBSTATEITEM 26`	Disabled (default). Use the default if you do not need to give priority usage of the system RAM to the Essbase kernel. Recommendation: If you want to give priority usage of the system RAM to the Essbase kernel, then enable cache memory locking. Be careful to leave at least one third of the system RAM for operations other than the Essbase Kernel.
Isolation level	Transaction	`SETDBSTATEITEM 18`	Uncommitted (default). See Chapter 40, "Ensuring Data Integrity," for more information.
Commit Blocks	Transaction	`SETDBSTATEITEM 21`	3,000 (default). Setting (default or other) ignored unless the isolation level is uncommitted. A zero value means that no synchronization point occurs; Essbase commits all affected blocks at the end of a transaction. If either Commit Blocks or Commit Rows has a non-zero value, a synchronization point occurs when the first threshold is reached. For example, assume Commit Blocks is 10 and Commit Rows is 0. When you load data, a synchronization point occurs after 10 blocks are updated.

Table 77: Database Settings and Defaults (Continued)

Setting	Application Manager Database > Setting	ESSCMD Command	Default Value and Comments
Commit Rows	Transaction	`SETDBSTATEITEM 22`	0 (default, no implicit commit); synchronization point occurs at the end of a transaction. Commit Rows setting is ignored unless isolation level is uncommitted See the note under Commit Blocks.
Wait for write access to locked data block	Transaction	`SETDBSTATEITEM 20`	Indefinite wait in Application Manager or `-1` in ESSCMD. Setting ignored unless the isolation level is committed.
Pre-image access	Transaction	`SETDBSTATEITEM 19`	Enabled. (default). Setting ignored unless the isolation level is committed.

Disk Volumes: volume name	Storage	`SETDBSTATEITEM 23`
		`SETDBSTATEITEM 24`
			If you do not specify a volume name, the Essbase Kernel uses only the volume that ARBORPATH points to and fills the entire volume as needed.
			Replaces DISKVOLUMES *`volume_name`* in `essbase.cfg`. *`volume_name`* is then used only for initial migration.

Disk volumes: partition size	Storage	`SETDBSTATEITEM 23`
		`SETDBSTATEITEM 24`
			Unlimited (default). Uses all available space on the specified volume.

Disk volumes: file type	Storage	`SETDBSTATEITEM 23`	Index and data (default).

Table 77: Database Settings and Defaults (Continued)

Setting	Application Manager Database > Setting	ESSCMD Command	Default Value and Comments
Disk volumes: maximum file size	Storage	`SETDBSTATEITEM 23`	2 GB (default).
Retrieval buffer size	General	`SETDBSTATEITEM 16`	10,240 B or, 10 KB. **Note:** In Version 4, this setting is specified as REPTKBYTEBUF in `essbase.cfg`.
Retrieval sort buffer size	General	`SETDBSTATEITEM 17`	10,240 B or, 10 KB. **Note:** In Version 4, this setting is specified as REPTKBYTESORTBUF in the `essbase.cfg` file.

Data compression	Storage	`SETDBSTATEITEM 14`
		`SETDBSTATEITEM 15`
			Bitmap compression enabled. (default).
			In Essbase Application Manager, you use a single setting to choose bitmap compression, RLE (run-length encoding) compression, or no compression.
			In ESSCMD, you use `SETDBSTATEITEM 14` to enable or disable compression and `SETDBSTATEITEM 15` to control compression type.

Eliminating Fragmentation

Fragmentation is unused disk space. Fragmentation is created when Essbase writes a data block to a new location on disk and leaves unused space in the former location of the data block. Block size increases because data from a data load or calculation is appended to the blocks; the blocks must therefore be written to the end of a data file.

The Essbase Kernel merges adjacent fragments into increasingly larger fragments so that unused space is more likely to be re-used.

In some cases, fragmentation cannot be reduced completely. Fragmentation is likely to occur with the following:

- Read/write databases that users are constantly updating with data
- Databases that execute calculations around the clock
- Databases that frequently update and recalculate dense members
- Data loads that are poorly designed
- Databases that contain a significant number of Dynamic Calc and Store members
- Databases that use an isolation level of uncommitted access with commit block set to zero

If you experience performance slow-downs, you can check to see if there is too much fragmentation of your database, and if there is, you can take steps to reduce the fragmentation:

- "Measuring Fragmentation" on page 1311
- "Preventing or Removing Fragmentation" on page 1313

Measuring Fragmentation

You can measure fragmentation using the average clustering ratio or average fragmentation quotient statistic from ESSCMD:

- "Using the Average Fragmentation Quotient" on page 1312
- "Using the Average Clustering Ratio" on page 1312

Using the Average Fragmentation Quotient

In ESSCMD, look at the Average Fragmentation Quotient that is returned when you execute GETDBSTATS command. Use this table to evaluate whether or not the level of fragmentation is likely to be causing performance problems:

Database Size	Fragmentation Quotient threshold
Small (up to 200 Mb)	60% or higher
Medium (up to 2 Gb)	40% or higher
Large (greater than 2Gb)	30% or higher

Any quotient above the high end of the range indicates that reducing fragmentation may help performance, with the following qualifications:

- The reported value of the Fragmentation Quotient is more accurate when there are no other write transactions running on the database.
- For databases less than 50Mb using the Direct IO access mode, the fragmentation quotient tends to be high. This does not necessarily indicate a need to reduce fragmentation, because the free space is created in 8Mb chunks and all of it might not get used right away.

Using the Average Clustering Ratio

The average clustering ratio database statistic indicates the fragmentation level of the data (.pag) files. The maximum value, 1, indicates no fragmentation. You can use any of the following tools to view the average clustering ratio for a database.

Tool	Instructions	For more information
Application Manager	Select Database > Information and click the Statistics tab	*Essbase Application Manager Online Help*
Administration Services	Click the Statistics tab on the Database Properties window	*Essbase Administration Services Online Help*
ESSCMD	GETDBSTATS	*Technical Reference* in the `docs` directory

Preventing or Removing Fragmentation

You can prevent and remove fragmentation:

- To prevent fragmentation, optimize data loads by sorting load records based upon sparse dimension members. See Chapter 20, “Introducing Data Loading.”
- To remove fragmentation, perform an export of the database, delete all data in the database with CLEARDATA, and reload the export file. See Chapter 44, “Backing Up and Restoring Data.”
- To remove fragmentation, force a dense restructure by adding or deleting a dummy dense member. See Chapter 49, “Optimizing Database Restructuring.”

Finding Other Optimization Information

Table 77 on page 1307 provides general-purpose information and does not account for the wide variety of configuration possibilities. For more information about performance and server, application, or other settings, see these chapters:

- For information on optimizing performance for a particular function, see these chapters:
    - Chapter 50, “Optimizing Data Loads”
    - Chapter 51, “Optimizing Calculations”
    - Chapter 52, “Optimizing with Intelligent Calculation”
    - Chapter 53, “Optimizing Reports and Other Types of Retrieval”
- For more detailed information on the Essbase Kernel, see the following chapters:
    - Chapter 38, “Managing Essbase Kernel Settings”
    - Chapter 39, “Allocating Storage and Compressing Data”
    - Chapter 40, “Ensuring Data Integrity”
    - Appendix C, “Estimating Disk and Memory Requirements”
    - Chapter 48, “Optimizing Essbase Caches”

Chapter

48

Optimizing Essbase Caches

This chapter describes the memory caches that Essbase uses and provides recommendations for cache-related settings.

This chapter includes the following sections:

About Essbase Caches

Essbase uses five memory caches to coordinate memory usage:

Table 78: Essbase Caches

Cache	Description
Index cache	The index cache is a buffer in memory that holds index pages. How many index pages are in memory at one time depends upon the amount of memory allocated to the cache.
Data file cache	The data file cache is a buffer in memory that holds compressed data files (`.PAG` files). Essbase allocates memory to the data file cache during data load, calculation, and retrieval operations, as needed. The data file cache is used only when direct I/O is in effect.

Table 78: Essbase Caches (Continued)

Cache	Description
Data cache	The data cache is a buffer in memory that holds uncompressed data blocks. Essbase allocates memory to the data cache during data load, calculation, and retrieval operations, as needed.
Calculator cache	The calculator cache is a buffer in memory that Essbase uses to create and track data blocks during calculation operations.
Dynamic calculator cache	The dynamic calculator cache is a buffer in memory that Essbase uses to store all of the blocks needed for a calculation of a Dynamic Calc member in a dense dimension (for example, for a query).

Essbase provides default size settings for each cache. You can adjust the size of any of these five caches as needed for each database. Appropriate cache size is affected by many factors, including database size, block size, index size, and available memory on the server. Cache size settings can effect database and general server performance significantly.

Use these topics to properly size caches for performance:

1. “Before You Size: Using Cache Memory Locking” on page 1317
2. “Sizing Caches” on page 1319
3. “Fine Tuning Cache Settings” on page 1339

Before You Size: Using Cache Memory Locking

Before setting cache sizes, you need to choose to enable cache memory locking or leave the default, that cache memory locking is disabled.

The setting for cache memory locking controls whether the memory used for the index cache, data file cache, and data cache is locked into physical memory, giving the Essbase kernel priority use of system RAM.

To use cache memory locking, you must be using direct I/O (buffered I/O is the default I/O access mode), and direct I/O requires a larger index cache size than buffered I/O. For more information, see “Migrating and Upgrading Databases” in the *Essbase Installation Guide*, and Chapter 38, “Managing Essbase Kernel Settings.”

Locking improves performance for a Essbase database because the system memory manager does not need to swap the memory used by the caches when swapping the memory used by OLAP Server. By default, cache memory locking is turned off.

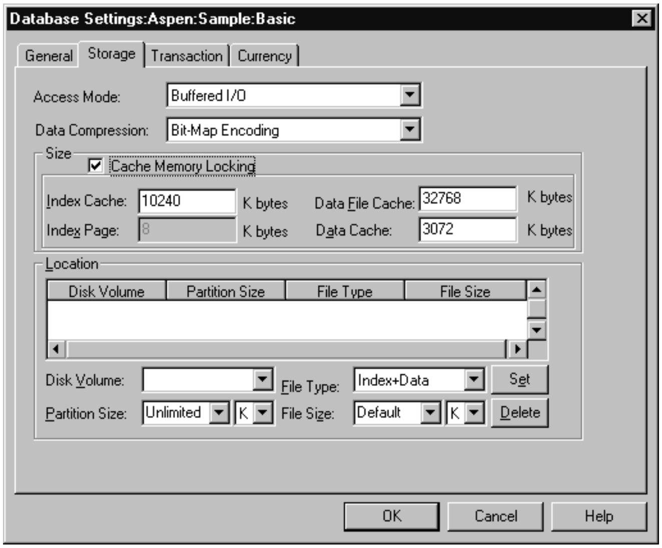
Enabling cache memory locking gives the Essbase Kernel priority use of system RAM. If you enable cache memory locking, leave at least one-third of the system RAM available for non-Essbase Kernel use. If you do not want to give the Essbase Kernel priority usage of system RAM, do not enable cache memory locking.

➤ To enable cache memory locking, use this procedure:

1. Install Essbase.
2. If you are running on any system other than Solaris, skip this step. If you are running Essbase on Solaris, run the Bourne shell script, *root.sh*. This script sets the server to run in Superuser mode so that it can lock memory. For more information, see the *Essbase Installation Guide*.
3. Start Essbase and open the Application Manager.

4. Select Database > Settings from the menu bar and click the Storage tab on the window which is displayed:

Figure 536: Database Settings Window—Changing Cache Memory Locking



Database Settings:Aspen:Sample:Basic

Access Mode: Buffered I/O

Data Compression: Bit-Map Encoding

Size

☑ Cache Memory Locking

Index Cache: 10240 K bytes　Data File Cache: 32768 K bytes

Index Page: 8 K bytes　Data Cache: 3072 K bytes

Location

Disk Volume	Partition Size	File Type	File Size

Disk Volume:　File Type: Index+Data

Partition Size: Unlimited K　File Size: Default K

5. Select the Cache Memory Locking check box.

Note: You must restart the database to initialize the new cache memory locking setting.

Tip: You can enable memory cache locking without Essbase Application Manager:

Tool	Instructions	For More Information
Administration Services	Database Properties window > Caches tab	*Essbase Administration Services Online Help*
MaxL	**alter database enable cache_pinning**	*Technical Reference* in DOCS directory
ESSCMD	SETDBSTATEITEM 26	

Sizing Caches

The settings that you should use for each of the caches that you can configure depends on data distribution and the dense/sparse configuration of the database.

If memory resources are restricted, you can optimize performance by adjusting the cache settings relative to the memory available on the machine which contains your database.

The needs for each site and even for a particular database can vary. Depending on the complexity and type of each operation, Essbase allocates as much memory for the data file cache and the data cache as needed. Use the recommended values in this section to estimate enough memory for *optimal* performance.

If you are using Essbase for the first time, cache sizes are automatically set to the default values discussed in the following sections. If you are migrating from Essbase Release 5.x, the data file cache is set to the default value and the other cache settings from that version are retained when you migrate. See the *Essbase Installation Guide* for migration information.

Note: Changes made to cache sizes take effect the next time you start the database.

Use these topics to find and understand recommendations for each cache size:

- "Sizing the Index Cache" on page 1320
- "Changing the Index Cache Size" on page 1321
- "Sizing the Data Cache" on page 1324
- "Changing the Data Cache Size" on page 1326
- "Sizing the Calculator Cache" on page 1327
- "Sizing Dynamic Calculator Caches" on page 1335

Note: The size of index pages is fixed at 8 K. This is to reduce input-output overhead, as well as to simplify database migration.

Sizing the Index Cache

The index is stored in index files on disk. When a database is active, the most recently accessed index pages are held in the index cache, which is a memory area that is allocated for index pages. How much of the index can be held in memory at one time depends upon the amount of memory you allocate to the index cache.

When a data block is requested, Essbase looks at the index pages in the index cache to find its location on disk. If the block location is not found in index pages in the index cache, the index page containing the block location is pulled into the index cache from disk. If the index cache is full, the least recently used index page in the cache is dropped to make room for the index page containing the location of the data block.

The effectiveness of the index cache size depends on the nature of the calculation you are performing. For example, if you were reloading and recalculating an entire database (such as a database that is refreshed each month), a high index cache size is not helpful because Essbase is creating new blocks rather than searching the index cache for existing blocks during calculation.

Table 79 shows default and recommended settings for the index cache.

Table 79: Index Cache Size Settings

Minimum Value	Default Value	Recommended Value
1 MB (1048576 bytes)	Buffered I/O: 1 MB (1048576 bytes) Direct I/O: 10 MB (10485760 bytes)	Combined size of all `ESSn.IND` files, if possible; as large as possible otherwise. Do not set this cache size higher than the total index size, as no performance improvement results. (To determine the total index size, see "Index Files" on page 1476.)

In general, if you are using direct I/O, make the index cache as large as system resources allow, up to 2 GB. If you are using buffered I/O, make them as small as possible.

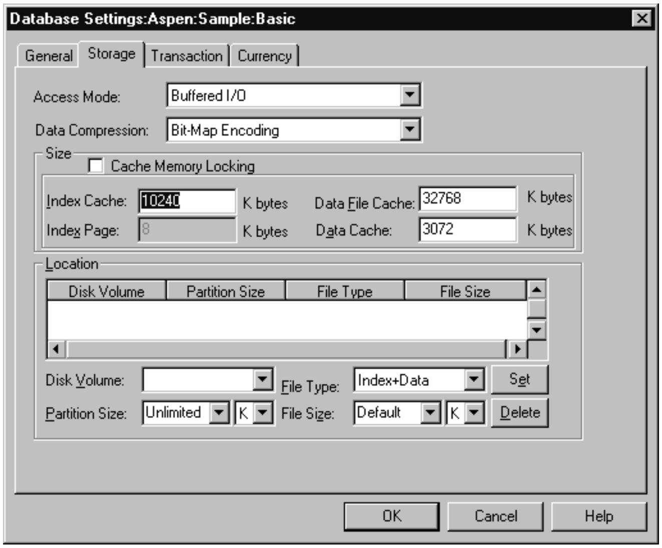
For information on testing and fine tuning your cache settings, see "Fine Tuning Cache Settings" on page 1339.

Changing the Index Cache Size

➤ To set the size of the index cache:

1. Start Application Manager.
2. Select Database > Settings and click the Storage tab:

Figure 537: Database Settings Window—Changing Index Cache Size



Database Settings:Aspen:Sample:Basic

Access Mode: Buffered I/O

Data Compression: Bit-Map Encoding

Size

☐ Cache Memory Locking

Index Cache: 10240 K bytes

Data File Cache: 32768 K bytes

Index Page: 8 K bytes

Data Cache: 3072 K bytes

Location

Disk Volume	Partition Size	File Type	File Size

Disk Volume:

File Type: Index+Data

Partition Size: Unlimited K

File Size: Default K

3. In the Index Cache text box, enter the desired value in kilobytes.
4. Click OK.
5. Restart the database to start using the new setting value.

Tip: You can change the index cache size without Application Manager:

Tool	Instructions	For More Information
Administration Services	Database Properties window > Caches tab.	*Essbase Administration Services Online Help*
MaxL	**alter database set index_cache_size**	*Technical Reference* in the `docs` directory
ESSCMD	SETDBSTATEITEM 12 SETDBSTATE	

Sizing the Data File Cache

The data file cache holds data files (`.PAG` files) in memory, if you are using direct I/O. If you are not using direct I/O, the data file cache is not used. How much of the data within data files can fit into memory at one time depends on the amount of memory you allocate to the data file cache.

In general, if you have to choose whether to allocate memory to the data cache or to the data file cache, choose the data file cache if you are using direct I/O.

Table 80 shows default and recommended settings for the data file cache.

Table 80: Data File Cache Size Settings

Minimum Value	Default Value	Recommended Value
Direct I/O: 10 MB (10485760 bytes)	Direct I/O: 32 MB (33554432 bytes)	Combined size of all `ESS`*`n`*`.PAG` files, if possible; otherwise as large as possible. This cache setting not used if Essbase is set to use buffered I/O.

In general, if you are using direct I/O, make the data file cache as large as system resources allow, up to 2 GB. If you are using buffered I/O, the data file cache is not used.

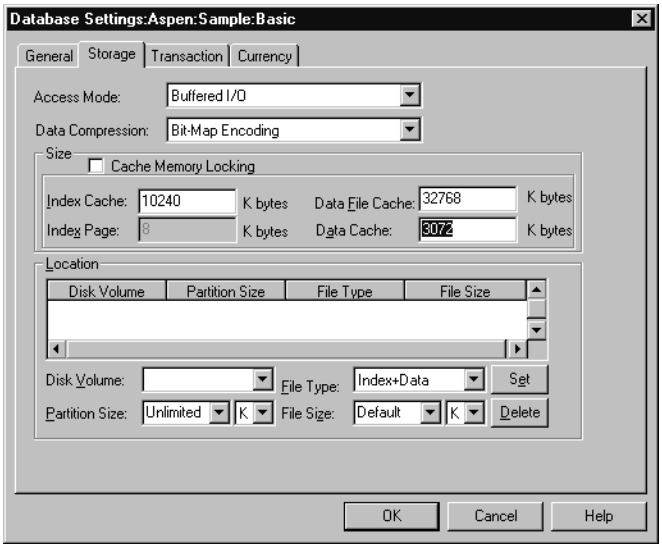
For information on testing and fine tuning your cache settings, see “Fine Tuning Cache Settings” on page 1339.

Changing the Data File Cache Size

➤ To set the size of the data file cache:

1. Start Application Manager.
2. Select Database > Settings and click the Storage tab:

Figure 538: Database Settings Window—Changing the Data File Cache Size



Access Mode: Buffered I/O

Data Compression: Bit-Map Encoding

☐ Cache Memory Locking

Index Cache: 10240 K bytes

Data File Cache: 32768 K bytes

Index Page: 8 K bytes

Data Cache: 3072 K bytes

Disk Volume:

File Type: Index+Data

Partition Size: Unlimited K

File Size: Default K

3. In the Data File Cache text box, enter the desired value in kilobytes.
4. Click OK.
5. Restart the database to start using the new setting value.

Tip: You can change the data file size without Application Manager:

Tool	Instructions	For More Information
Administration Services	Database Properties window > Caches tab.	*Essbase Administration Services Online Help*
MaxL	**alter database set data_file_cache_size**	*Technical Reference* in the `docs` directory
ESSCMD	SETDBSTATEITEM 27	

Sizing the Data Cache

Data blocks reside on physical disk and in memory. The data cache is the memory area that is allocated to hold uncompressed data blocks. The number of blocks that can be held in the data cache at one time depends on the amount of memory you allocate to the data cache.

When a block is requested, Essbase searches the data cache for the block. If Essbase finds the block in the cache, it is accessed immediately. If the block is not found in the cache, Essbase searches the index for the appropriate block number and then uses the index entry of the block to retrieve it from the appropriate data file on disk. Retrieving a requested block from the data cache is faster, and therefore improves performance.

In general, if you have to choose whether to allocate memory to the data cache or to the data file cache, choose the data file cache if you are using direct I/O.

This table shows default and recommended settings for the data cache.

Table 81: Data Cache Size Settings

Minimum Value	Default Value	Recommended Value
3 MB (3145728 bytes)	3 MB (3145728 bytes)	0.125 * the value of data file cache size. Increase value if either of these conditions exist: • Many concurrent users are accessing different data blocks. • Calc scripts contain functions on sparse ranges, and the functions require all members of a range to be in memory (for example, when using @RANK and @RANGE) • For data load, the number of threads specified by the DLTHREADSWRITE setting is very high and the expanded block size is large. See "Managing Parallel Data Load Processing" on page 1365 for more information.

Make the data cache as small as possible whether you are using buffered I/O or direct I/O.

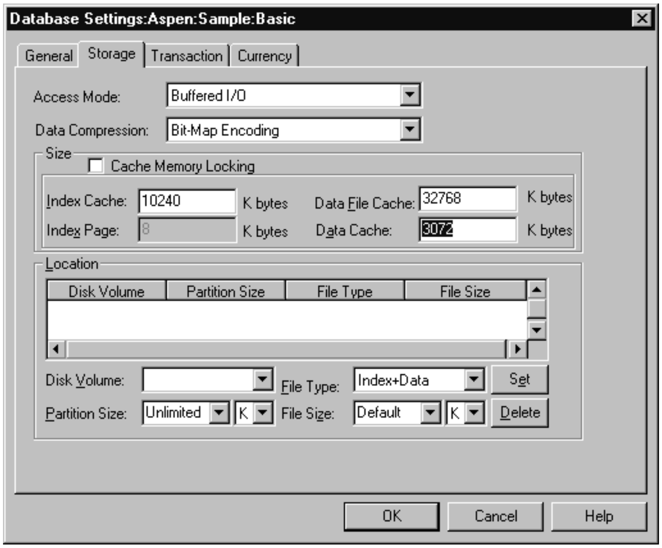
For information on testing and fine tuning cache settings, see "Fine Tuning Cache Settings" on page 1339.

Changing the Data Cache Size

➤ To set the size of the data cache, use this procedure:

1. Start Application Manager.
2. Select Database > Settings and click the Storage tab:

Figure 539: Database Settings Window—Changing the Data Cache Size



Access Mode: Buffered I/O

Data Compression: Bit-Map Encoding

☐ Cache Memory Locking

Index Cache: 10240 K bytes

Data File Cache: 32768 K bytes

Index Page: 8 K bytes

Data Cache: 3072 K bytes

Disk Volume:

File Type: Index+Data

Partition Size: Unlimited K

File Size: Default K

3. In the Data Cache text box, enter the desired value in kilobytes.
4. Click OK.
5. Restart the database to start using the new setting value.

Tip: You can change the data cache size without Application Manager:

Tool	Instructions	For More Information
Administration Services	Database Properties window > Caches tab	*Essbase Administration Services Online Help*
MaxL	**alter database set data_cache_size**	*Technical Reference* in the `docs` directory
ESSCMD	SETDBSTATEITEM 5 SETDBSTATE	

Sizing the Calculator Cache

Essbase can create a bitmap, whose size is controlled by the size of the calculator cache, to record and track data blocks during a calculation. Determining which blocks exist using the bitmap is faster than accessing the disk to obtain the information, particularly if calculating a database for the first time or calculating a database when the data is very sparse.

Essbase uses the calculator cache bitmap if the database has at least two sparse dimensions, and either of these conditions are also met:

- You calculate at least one, full sparse dimension
- You specify the SET CACHE ALL command in a calc script.

The best size for the calculator cache depends on the number and density of the sparse dimensions in your outline. Use these topics to understand the calculator cache bitmap, size the calculator cache, and change the size of the calculator cache (and therefore the largest possible size for the bitmap), if required:

- "Understanding the Calculator Cache Bitmap" on page 1328
- "Calculating the Calculator Cache Size" on page 1330
- "Changing the Calculator Cache with Calc Scripts" on page 1335
- "Calculating the Database for the First Time" on page 1334
- "Changing the Calculator Cache for Large, Flat Database Outlines" on page 1335

Understanding the Calculator Cache Bitmap

For the calculator cache, Essbase separates sparse dimensions in the database into two groups:

- Bitmap dimensions: the sparse dimensions from the database outline that Essbase fits into the bitmap until the bitmap is full. Each member combination of the sparse dimensions placed in the bitmap occupies 1 bit of memory, and there must be enough space in the bitmap for every member combination of a sparse dimension for it to be placed in the bitmap.
- Anchoring dimensions: the remaining one or more sparse dimensions in the database outline that do not fit into the bitmap.

Essbase starts with the first sparse dimension in the database outline and fits as many sparse dimensions as possible into the bitmap. The dimensions that fit are the bitmap dimensions. Essbase stops the process when it cannot fit another complete sparse dimension into the bitmap. Because the calculator cache controls the size of the bitmap, the number of sparse dimensions that can fit in the bitmap depends on the size of the calculator cache (and the number and size of the sparse dimensions).

The remaining sparse dimensions are the anchoring dimensions. For anchoring dimensions, Essbase cannot use the bitmap and must to determine whether or not blocks exist.

To see which dimensions are anchoring dimensions and which are bitmap dimensions, use the SET MSG DETAIL calculation command to display bitmap information in the application log. For more information on this command, see the *Technical Reference* in the `docs` directory.

Carefully order the sparse dimensions in your outline so that as many dimensions as possible can be placed into the bitmap. Start with the dimension that contains the fewest members, and continue until the dimension with the most members is last. This order allows more dimensions to fit into the bitmap and results in improved calculation performance.

Note: The order of sparse dimensions in the outline also affects query performance. To optimize the outline for query performance, see "Designing an Outline to Optimize Performance" on page 124.

Essbase uses a single bitmap if there is more than one anchoring dimension or if the calculator cache is not large enough to support multiple bitmaps, and uses two or more bitmaps if there is a single anchoring dimension.

A single bitmap has these properties:

- A single bitmap is used to track child blocks.
- A single bitmap uses the least memory but is less efficient than multiple bitmaps.

Multiple bitmaps have these properties:

- Two or more bitmaps are used, one to track child blocks and one to track parent blocks.
- Multiple bitmaps use more memory but are faster than using a single bitmap. The performance improvement is particularly high when you are calculating the database for the first time.
- The number of bitmaps used is determined by the maximum number of dependent parents for any of the members in the anchoring dimension. A member has one dependent parent, unless it has a shared member. For example, consider the Product dimension of the Sample Basic database. The member Cola (100-10) has one parent, which is Colas (100). However, Diet Cola (100-20) has two parents, which are Diet Drinks (Diet) and Colas (100). No members of Product have more than two dependent parents. Therefore, if Product is the anchoring dimension, the maximum number of dependent parents is 2.

Essbase chooses one of three options for the calculation:

Table 82: Options for calculator cache

Option	Method	Performance Rating
1	Single anchoring dimension, multiple bitmaps.	1
2	Single anchoring dimension, single bitmap	2
3	Multiple anchoring dimensions, single bitmap	3

Essbase chooses the optimal performance method for a database calculation, based on the size of the calculator cache. If the calculator cache size is too small for any of the above options, Essbase does not use a calculator cache. Calculation performance may be significantly impaired.

Enabling parallel calculation may change the which calculator cache option is used. See “Calculator Cache” on page 1385 for details.

CAUTION: If you are calculating the database for the first time, the size of the calculator cache is particularly significant for calculation performance. If possible, ensure that the calculator cache is large enough for Essbase to use the optimal calculator cache option.

Calculating the Calculator Cache Size

The optimum size of the calculator cache depends on the amount of memory the system has available. It also depends on the nature and configuration of the database. You can calculate the calculator cache size that is required for each of the three options in Table 82 on page 1329.

Consider an example database with five sparse dimensions (S1 to S5):

Sparse Dimension	# of Members	Dependent Parents
S1	20	Not applicable
S2	20	Not applicable
S3	50	Not applicable
S4	50	Not applicable
S5	200	3

Use this example information for these sample calculations:

- “Option 1: Single Anchoring Dimension, Multiple Bitmaps” on page 1331
- “Option 2: Single Anchoring Dimension, Single Bitmap” on page 1332
- “Option 3: Multiple Anchoring Dimensions, Single Bitmap” on page 1333

Option 1: Single Anchoring Dimension, Multiple Bitmaps

For this example calculation, assume the following facts about a database (from Table 82 on page 1329):

Bitmap dimensions	S1, S2, S3, S4
Anchoring dimension	S5
Dependent parents in anchoring dimension	3

Now perform this calculation:

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**

In order for Essbase to use multiple bitmaps for this database with a single anchoring dimension, the calculator cache needs to be 625,000 bytes.

Option 2: Single Anchoring Dimension, Single Bitmap

For this example calculation, assume the following facts about a database (from Table 82 on page 1329):

Bitmap dimensions	S1, S2, S3, S4
Anchoring dimension	S5
Dependent parents in anchoring dimension	Not applicable

Now perform this calculation:

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**

In order for Essbase to use a single bitmap for this database with a single anchoring dimension, the calculator cache needs to be 125,000 bytes.

Option 3: Multiple Anchoring Dimensions, Single Bitmap

For this example calculation, assume the following facts about a database (from Table 82 on page 1329):

Bitmap dimensions	S1, S2, S3
Anchoring dimensions	S4, S5
Dependent parents in anchoring dimensions	Not applicable

Now perform this calculation:

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**

In order for Essbase to use a single bitmap for this database with a single anchoring dimension, the calculator cache needs to be 2,500 bytes.

Choosing a Calculator Cache Size for a Database

The following table shows which calculator cache option Essbase uses, depending on the calculator cache size specified:

Minimum Size Specified	Option Selected
625,000 bytes	Option 1 (provides optimal performance)
125,000 bytes	Option 2
2,500 bytes	Option 3

If you specify a calculator cache size of less than 2,500 bytes, Essbase does not use a calculator cache during the calculation. Calculation performance may be significantly impaired.

You can check which calculator cache option Essbase is able to use on a database by using the SET MSG SUMMARY command in a calc script. Run the following calc script on the empty database:

```
SET MSG SUMMARY;
CALC ALL;
```

Essbase displays the calculator cache setting in the ESSCMD window or in the application log. For more information, see "Monitoring and Tracing Calculations" on page 1373.

The maximum calculator cache size that you can specify is 200,000,000 bytes. The default is 200,000 bytes. The calculator cache size that you choose depends on how much memory is available and the configuration of the database.

Note: The sizes of the calculator, index, data file, and data caches usually have a greater effect on performance if the database calculation is based more on aggregations and less on formula calculations.

Calculating the Database for the First Time

If you are calculating the database for the first time, the size of the calculator cache is particularly significant. If possible, ensure that the calculator cache is large enough for Essbase to use the optimal calculator cache option. For more information, see "Calculating the Calculator Cache Size" on page 1330.

Changing the Calculator Cache with Calc Scripts

You can use the default calculator cache size, or you can set the size of the calculator cache within a calc script. If you set the size from a calc script, the setting is used only for the duration of the calc script. For more information, review information about the calc script SET CACHE command and the CALCCACHE configuration setting in the *Technical Reference* in the `docs` directory.

Changing the Calculator Cache for Large, Flat Database Outlines

You can use the SET CALCHASHTBL command to optimize how Essbase uses the calculator cache when it calculates large, flat database outlines (for example, where one member has more than 5000 children). To use SET CALCHASHTBL, you must enable the calculator cache.

When you enable SET CALCHASHTBL, Essbase uses a hash table to optimize use of the calculator cache for large, flat databases. Using this feature may significantly improve the performance of a CALC ALL of the database or a CALC DIM of the dimension containing the member with over 5000 children.

For more information, see the SET CALCHASHTBL command and the CALCOPTCALCHASHTBL and CALCHASHTBLMEMORY configuration settings in the *Technical Reference* in the `docs` directory.

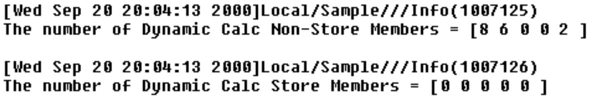
Sizing Dynamic Calculator Caches

Hyperion Essbase uses a separate dynamic calculator cache for each open database. The `DYNCALCCACHEMAXSIZE` setting in the configuration file, `ESSBASE.CFG`, specifies the maximum size of each dynamic calculator cache on the server. By default, the maximum size is 20 MB. Hyperion Essbase allocates area in a dynamic calculator cache for data blocks until the maximum memory area specified by the `DYNCALCACHEMAXSIZE` setting is allocated.

For more information about `DYNCALCACHEMAXSIZE` and other dynamic calculator cache settings, see “Changing the Dynamic Calculator Cache Size” on page 1336.

Reviewing Dynamic Calculator Cache Usage

For each database, Essbase writes two messages to the application log for each data retrieval:



```
[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 ]
```

The first message describes the total amount of time required for the retrieval. If a dynamic calculator cache is used, the second message displays the number of blocks calculated within the data calculator cache (DCC = *n*) and the number of blocks calculated in general memory (non-DCC = *n*).

Changing the Dynamic Calculator Cache Size

Four configuration file settings are relevant to dynamic calculator caches. The optimum values for these four dynamic calculator cache settings depend on the amount of memory on the server machine, the configuration of all databases on the server machine, and the nature of user queries.

Table 83 describes each setting and includes recommendations on how to determine values for your system. To match your site's unique requirements, you may need to test and adjust the settings.

Table 83: `essbase.cfg` Settings for Dynamic Calculator Caches

DYNCALCCACHEMAXSIZE	
Description	This setting specifies the maximum size Hyperion Essbase can allocate to each dynamic calculator cache on the server.
Recommended Setting	Recommended setting value = C * S * U. • C is the value of the appropriate `CALCLOCKBLOCK` setting in the `ESSBASE.CFG` file. (The `SET LOCKBLOCK` command specifies which `CALCLOCKBLOCK` setting to use.) • S is the size of the largest expanded block across all databases on the machine. To calculate the expanded block size, multiply the number of members (including Dynamic Calc members) in each dense dimension together, and then multiply the result by the size of each member cell, 8 bytes. For example, for the dense dimensions in Sample Basic, 12 (Year) * 8 (Measures) * 3 (Scenario) * 8 bytes = 2304 bytes. • U is the maximum number of expected concurrent users on the database that has the largest number of concurrent users. Assigning the value 0 (zero) to `DYNCALCACHEMAXSIZE` tells Hyperion Essbase not to use dynamic calculator caches. The default size is 20 megabytes (20,971,520 bytes).

Table 83: `essbase.cfg` *Settings for Dynamic Calculator Caches (Continued)*

DYNCALCCACHEWAITFORBLK	
Description	If Essbase uses all of the area allocated for a dynamic calculator cache, this setting tells Essbase whether to wait until space becomes available in the cache or to immediately write and calculate the blocks in memory outside the dynamic calculator cache. If the dynamic calculator cache is too small, it is possible for more than one thread to be in queue, each thread waiting to calculate its data blocks.
Recommended Setting	Recommended setting value = FALSE (default value). Before setting to `TRUE`, try these alternatives: • Add physical memory to the server machine • Increase the value of `DYNCALCCACHEMAXSIZE`, test, and repeat until you verify that you cannot use any more memory for the dynamic calculator cache.
DYNCALCCACHEBLKTIMEOUT	
Description	If Essbase is to wait for available space in the dynamic calculator cache, this setting defines how long it waits.
Recommended Setting	Recommended setting value = WT / B. • WT is the maximum tolerable wait time for a query; for example, 5 seconds. • B is the total number of logical blocks accessed in the largest query. To determine the value of B, check the messages in the application log for the largest number of Dyn.Calc.Cache “Big Block Allocs” for a query, as shown in Figure 416 on page 791.
DYNCALCCACHEBLKRELEASE	
Description	If Essbase has waited the specified time and space is still not available in the dynamic calculator cache, this setting tells Essbase whether to write and calculate the blocks immediately outside the dynamic calculator cache or to create space in the dynamic calculator cache by swapping out blocks and temporarily compressing the swapped blocks in a dynamic calculator cache compressed-block buffer.
Recommended Setting	Recommended setting value = FALSE (default value). Set to TRUE only if you are experiencing severe memory shortage problems.

Table 83: `essbase.cfg` Settings for Dynamic Calculator Caches (Continued)

DYNCALCCACHECOMPRBLKBUFSIZE	
Description	If Essbase has waited the specified wait time and the DYNCALCCACHEBLKRELEASE setting is TRUE, this setting is the size of the dynamic calculator cache compressed-block buffer.
Recommended Setting	Recommended setting value = (C * S) / 2. • C is the value of the current `CALCLOCKBLOCK` setting in the `ESSBASE.CFG` file. The `SET LOCKBLOCK` command specifies which `CALCLOCKBLOCK` configuration setting is current. • S is the size of the largest expanded block across all databases on the machine. Calculate S as described for the `DYNCALCCACHEMAXSIZE` setting.

Note: After changing any parameter in the `essbase.cfg` file, you must close and restart Essbase to use the new values.

For more information about dynamic calculator cache settings, see the *Technical Reference* in the `docs` directory.

Fine Tuning Cache Settings

After using a database at your site with typical data, user access, and standard environment (including server machines, network, etc.), check to see how Essbase performs. It is difficult to predict optimal cache sizes without testing. You may need to adjust cache settings.

Understanding Cache Settings

The sizes of the index cache and the data file cache (when direct I/O is used) are the most critical Essbase cache settings. In general, the larger these caches, the less swapping activity occurs; however, it does not always help performance to set cache sizes larger and larger. Read this entire section to understand cache size considerations.

Index Cache

The advantages of a large index cache start to level off after a certain point. Whenever the index cache size equals or exceeds the index size (including all index files on all volumes), performance does not improve. However, to account for future growth of the index, you can set the index cache size larger than the current index size. See “Index Files” on page 1476 for an example of estimating index size.

Data File Cache

If possible, set the data file cache to equal the size of the stored data, which is the combined size of all `ESS*.PAG` files. Otherwise, the data file cache should be as large as possible. If you want to account for future growth of stored data, you can set the data file cache size larger than the current size of stored data.

Note: The data file cache is used only if you are using direct I/O.

Data Cache

The data cache should be about 0.125 times the data file cache. However, certain calculations require a larger data cache size. If many concurrent users are accessing different data blocks, this cache should be larger.

In general, if you have to choose between allocating memory to the data file cache or allocating it to the data cache, choose the data file cache if you are using direct I/O. If you are migrating from a previous version of Essbase, see the *Essbase Installation Guide* for important migration information.

Testing Cache Settings

You can check Essbase performance in several ways. One way is to check the Run-time page of the Database Information dialog box. A second method is to click the Statistics tab on the Database Properties window in Administration Services.

➤ To check values in the Run-time page of the Database Information dialog box, select Database > Information from the Application Manager menu, and check the hit ratios:

- The cache hit ratio indicates the percentage of time that a requested piece of information is already in the cache. A higher hit ratio indicates that the data is in the cache more often. This improves performance because the requested data does not have to be retrieved from disk for the next process. A hit ratio of 1.0 indicates that every time data is requested, it is found in the cache. This is the maximum performance possible from a cache setting.
- The Hit Ratio on Index Cache setting indicates the Essbase Kernel's success rate in locating index information in the index cache without having to retrieve another index page from disk.
- The Hit Ratio on Data File Cache setting indicates the Essbase Kernel's success rate in locating data file pages in the data file cache without having to retrieve the data file from disk.
- The Hit Ratio on Data Cache setting indicates Essbase's success rate in locating data blocks in the data cache without having to retrieve the block from the data file cache.

6. Check memory allocation. Add smaller amounts of memory at a time, if needed, because a smaller increment may have the same benefit as a large one. Large, incremental allocations of memory usually result in very little gain in the hit ratio.

Determining Which Settings to Change

The sizes of the index cache and the data file cache are the most critical Essbase cache settings. In general, the larger these buffers, the less swapping activity occurs; however, it does not always help performance to set cache sizes larger and larger. Read this entire section to understand the cache size considerations.

The advantages of a large index cache start to level off after a certain point. At any given time, when the index cache size equals or exceeds the index size (including all index files on all volumes), performance does not improve. However, to account for future growth of the index, you can set the index cache size larger than the current index size.

Because the index cache is filled with index pages, for optimum use of storage, set the size of the index cache to be a multiple of the size of the index page (8 KB).

If possible, set the data file cache to equal the size of the stored data, which is the combined size of all `ESS*.PAG` files. Otherwise, the data file cache should be as large as possible. If you want to account for future growth of stored data, you can set the data file cache size larger than the current size of stored data.

The data cache should be about 0.125 times the data file cache. However, certain calculations require a larger data cache size. If many concurrent users are accessing different data blocks, this cache should be larger.

Note: The data file cache is used only if you are using direct I/O.

In general, if you have to choose between allocating memory to the data file cache or allocating it to the data cache, choose the data file cache. If you are migrating from a previous version of Essbase, see the *Essbase Installation Guide* for important migration information.

Checking Performance

You can check cache statistics for a database by using the `GETPERFSTATS` command in ESSCMD. For more information on the `GETPERFSTATS` command, see the *Technical Reference* in the `docs` directory.

Chapter 46, “Monitoring Performance,” provides more information about ways to check performance.

Running Test Calculations

Because calculations are the most processor-intensive operations on a Essbase database, you should run test calculations and examine how various cache sizes affect memory use on OLAP Server.

Chapter

49

Optimizing Database Restructuring

This chapter describes how changes to a database outline affect Essbase:

In addition to the information in this chapter, look for information and instructions concerning restructuring in the following topics:

- "Designing an Outline to Optimize Performance" on page 124
- "Sorting Dimensions and Members" on page 184
- "Moving Members" on page 186
- "Using Attribute Dimensions" on page 237
- "Using Alternative Design Approaches" on page 237
- "Outline Performance Considerations" on page 238
- "Calculation and Retrieval Performance Considerations" on page 259

Understanding Database Restructuring

As your business changes, you change the Essbase database outline to capture new product lines, provide information on new scenarios, reflect new time periods, etc. Some changes to a database outline affect the data storage arrangement, forcing Essbase to restructure the database.

Because changes that require restructuring the database are very time-consuming, (unless you discard your data before restructuring), you may wish to make decisions about these kinds of changes, based on how much they affect performance. This section provides the information you need to understand how restructuring affects performance, and describes tasks you can perform related to database restructuring:

- "Types of Database Restructuring" on page 1344
- "Conditions Affecting Database Restructuring" on page 1345
- "Temporary Files Used During Restructuring" on page 1346
- "Understanding a Full Restructure" on page 1346
- "Understanding a Sparse Restructure" on page 1347

Note: For more information about clearing data, and thus avoiding some restructuring, see.

Types of Database Restructuring

Essbase uses three types of restructure operations:

- Full restructure: If a member of a dense dimension is moved, deleted, or added, Essbase restructures the blocks in the data files and creates new data files. When Essbase restructures the data blocks, it regenerates the index automatically so that index entries point to the new data blocks. Essbase marks all restructured blocks as dirty, so after a full restructure you need to recalculate the database. Full restructuring is the most time-consuming of the restructures and, for large databases, can take a very long time to complete.
- Sparse restructure: If a member of a sparse dimension or a member of an attribute dimension is moved, deleted, or added, Essbase restructures the index and creates new index files. Restructuring the index is relatively fast; the amount of time required depends on the size of the index.
- Outline-only restructure: If a change affects only the database outline, Essbase does not restructure the index or data files. Member name changes, creation of aliases, and dynamic calculation formula changes are examples of changes that affect only the database outline.

If you use incremental restructuring, Essbase defers full restructuring. If you change a database outline frequently, consider enabling incremental restructuring. See "Incremental Restructuring and Performance" on page 1348 for more information.

Note: Whether a database outline is changed via the Outline Editor or using dimension building does not influence restructuring. Only the type of information change influences what type of restructuring, if any, takes place.

Conditions Affecting Database Restructuring

Intelligent Calculation, name changes, and formula changes affect database restructuring:

- If you use Intelligent Calculation in your database, all restructured blocks are marked as dirty whenever data blocks are restructured. This forces the next default Intelligent Calculation to be a full calculation.
- If you change a name or a formula, Essbase does not mark the affected blocks as dirty. Therefore, you must use a method other than full calculation to recalculate the member or the database.

Use this table to find more information about the topics mentioned in this section:

Table 84: Topics Related To Database Restructuring

Topic	Related Information
Intelligent Calculation	Chapter 52, "Optimizing with Intelligent Calculation"
Sparse and dense dimensions	Chapter 3, "Basic Architectural Elements"
Attribute dimensions	Chapter 9, "Working with Attributes"
Dimension building	Chapter 18, "Introducing Dynamic Dimension Building"
Outline Editor	Chapter 7, "Creating and Changing Database Outlines"

Temporary Files Used During Restructuring

When Essbase restructures both the data blocks and the index, it uses these files:

Table 85: Files Used During Database Restructuring

File	Description
ESS*xxxxx*.PAG	Essbase data file
ESS*xxxxx*.IND	Essbase index file
dbname.ESM	Essbase kernel file that contains control information used for database recovery
dbname.TCT	Transaction control table
dbname.IND	Free fragment file for data and index free fragments
dbname.OTL	Outline file that stores all metadata for a database and defines how data is stored. This outline file does not store data.

Understanding a Full Restructure

To perform a full restructure, Essbase does the following:

1. Creates temporary files that are copies of the .IND, .PAG, .OTL, .ESM, and .TCT files. Each temporary file substitutes either N or U for the last character of the file extension, so the temporary file names are *dbname*.INN, ESS*xxxxx*.INN, ESS*xxxxx*.PAN, *dbname*.OTN, *dbname*.ESN, and *dbname*.TCU.
2. Reads the blocks from the database files copied in Step 1, restructures the blocks in memory, and then stores them in the new temporary files. This step takes the most time.
3. Removes the database files copied in Step 1, including .IND, .PAG, .OTL, .ESM, and .TCT files.
4. Renames the temporary files to the correct file names: .IND, .PAG, .OTL, .ESM, and .TCT.

Understanding a Sparse Restructure

When Essbase does a sparse restructure (restructures just the index), it uses the following files:

- `ESSxxxxx.IND`
- *dbname*`.OTL`
- *dbname*`.ESM`

To perform a sparse restructure, Essbase does the following:

1. Renames the *dbame*`.ESM` file to *dbname*`.ESR`
2. Renames the `ESS`*xxxxx*`.IND` files to `ESS`*xxxxx*`.INM`.
3. Creates new index files (`ESS`*xxxxx*`.IND`) to store index information that is changed by the restructuring operation.
4. Removes *dbname*`.ESR` and `ESS`*xxxxx*`.INM` created in Step 1.

Optimizing Restructure Operations

If a database outline changes frequently, analyze the outline and the types of changes that you are making. Remember that changes to sparse dimensions or attribute dimensions are relatively fast because only the index needs to change. Changes to dense dimensions are relatively slow because data blocks need to be rebuilt.

These types of restructure operations are listed from fastest to slowest:

- Outline only (no index or data files)
- Sparse (only index files)
- Full (index files and data files) as a result of adding, deleting, or moving members and other operations as listed in Table 86
- Full (index and data files) as a result of changing a dense dimension to sparse or changing a a sparse dimension to dense.

Actions That Improve Performance

There are a number of things you can do to improve performance related to database restructuring:

- If you change a dimension frequently, make it sparse.
- Use incremental restructuring to control when Essbase performs a required database restructuring. See "Incremental Restructuring and Performance" on page 1348.
- Select options when you save a modified outline that reduce the amount of restructuring required. See "Saving a Modified Outline" on page 1349.

Incremental Restructuring and Performance

If you make frequent changes to a database outline, you may want to consider enabling incremental restructuring. When incremental restructuring is enabled, Essbase defers restructuring so that a change to the database outline or to a dimension does not cause structural change. Essbase restructures the index and, if necessary, the affected block the next time the block is accessed.

Understanding Incremental Restructuring

When incremental restructuring is enabled, Essbase defers restructuring for the database changes listed in Table 86, unless otherwise noted in the table.

The following changes override incremental restructuring; that is, they result in immediate restructuring, regardless of whether incremental restructuring is enabled:

- Adding or deleting a non-attribute dimension.
- Deleting a stored member of a sparse dimension.
- Changing a dimension definition from sparse to dense or from dense to sparse.
- If you are using linked reporting objects (LROs) in a database, incremental restructuring is automatically disabled on that database. This does not affect other databases on the server.
- Certain member additions and certain changes to sparse dimensions can also trigger immediate restructuring. For more information, see Table 86.

Note: Recalculate the database after any type of restructure operation.

Using Incremental Restructuring

You can enable incremental restructuring for any of the following:

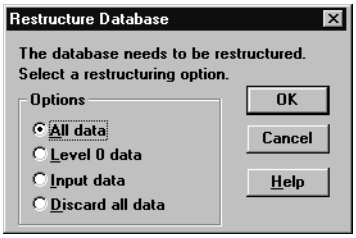
- An individual database in an application
- All databases in an application
- All databases in all applications

To enable incremental restructuring, use the INCRESTRUC parameter in the `ESSBASE.CFG` file. For more information on the INCRESTRUC parameter and syntax, see the *Technical Reference* in the `docs` directory.

Essbase logs outline changes in an internal file, *`dbname`*`.OCL`. Essbase clears the file when it does a full database restructure or when you clear or reset the database. The file *`dbname`*`.OCL` can grow quite large. To clear this file, issue VALIDATE in ESSCMD. VALIDATE causes Essbase to restructure any blocks whose restructure was deferred; thus, the file is cleared. When you issue VALIDATE, make sure that the database is *not* in read-only mode (read-only mode is used for backing up a database). For more information on the VALIDATE command, see "Checking Structural and Data Integrity" on page 1145.

Saving a Modified Outline

Essbase displays this dialog box when you save outline changes that trigger database restructuring (using the Outline Editor):



Restructure Database

The database needs to be restructured.
Select a restructuring option.

Options
- ◉ All data
- ○ Level 0 data
- ○ Input data
- ○ Discard all data

In the Restructure Database dialog box, you must choose one of the options:

- All data—This option preserves all data, but is requires more time than the other options.
- Level 0 data—This option preserves data only for level zero (leaf node) members. If the outline change requires a database recalculation, and if all data required for the calculation is in level zero members, this is the fastest restructure option, and requires the least amount of disk space.

  When you select Level 0 data, all upper level blocks are deleted before restructuring. This option reduces the disk space required to restructure and improves calculation time. Essbase recreates the upper-level blocks when it calculates the database.
- Input data—This option preserves only the blocks that contain loaded data. This option prevents any blocks created by data loading from being deleted, whether they are upper-level or lower-level blocks. If you change the database and need to recalculate, and if you load data into various levels of the outline, this is restructure option is the fastest, and uses the least disk space.

  The Input data option deletes all blocks that contain calculated values before restructuring. This option reduces the disk space required to restructure and improves calculation time when the database is calculated.
- Discard all data—This option preserves no data. Use this option when you expect to reload the data or when the outline is so radically changed that no existing data applies.

If your database contains data, you need enough free disk space on the server to create a backup copy of the database. Backup ensures that any abnormal termination during the restructure process does not corrupt the database.

Essbase may display a “Restructuring not required” message, yet still perform an index-only restructure. This event is most likely to occur if you make changes to a sparse dimension. If you try to cancel a restructure operation, Essbase may issue a “Can’t cancel” message. If such a message is displayed, Essbase is performing final cleanup and it is too late to cancel.

Outline Change Log

If you activate the outline change log, Essbase records all activity that affects the outline (member name changes, member moves, and so on). The more changes you make to the outline, the more updates Essbase must make to the log, thus slowing performance.

By default, Essbase does not log outline changes. To see if outline logging is slowing performance, look for `OUTLINECHANGELOG TRUE` in the `ESSBASE.CFG` file. For more information about the log, see "Understanding and Using the Outline Change Log" on page 1228.

Essbase Partitioning Option

When you use Partitioning, Essbase tracks outline changes so that you can synchronize the database outlines across partitions. Tracking outline changes slows restructuring, particularly when there are many structural changes.

If Essbase restructures data when you are using partitioning, perform the following steps to make sure that data is synchronized across partitions:

1. Validate the partitions. Use the Validate page of the Partition Wizard. See "Validating the Partition" on page 386 for more information.

   Note: To validate a partition, you must have DB Designer or higher privileges.

2. Synchronize the outlines of the partitions. Select Database > Synchronize Outline to open the Synchronize Outline dialog box. See "Synchronizing Outlines" on page 397 for more information.

Tip: You can use ESSCMD to synchronize outlines. See the *Technical Reference* in the `docs` directory for information about these commands. See Chapter 45, "Automating the Production Environment" for information about ESSCMD.

Outline Change Quick Reference

Table 86 shows all outline changes that affect calculation and restructuring, including incremental restructuring (see "Incremental Restructuring and Performance" on page 1348 for more information).

Note: If you are using Partitioning, restructuring affects only the local, target database, not the database to which you are connected.

Table 86: How Actions Affect Databases and Restructuring

Action	Calculation and Standard Restructure Effects	Incremental Restructuring Applies? (If Enabled)
Delete, Add, or Move Member		
Delete member of sparse dimension	Data needs to be recalculated to reflect changes to relationships. Essbase deletes from the index file all pointers to blocks represented by the deleted member. Because the blocks are no longer pointed to, they become free space.	For regular members, no. Essbase restructures the index, overriding incremental restructure. For shared and label-only members, yes, restructuring is deferred.
Delete member of attribute dimension	None	No
Delete member of dense dimension	Data needs to be recalculated to reflect changes to relationships. Essbase restructures the data files to reflect a changed block size. Essbase restructures the index.	Yes. Restructure deferred.

Table 86: How Actions Affect Databases and Restructuring (Continued)

Action	Calculation and Standard Restructure Effects	Incremental Restructuring Applies? (If Enabled)
Delete shared member in sparse or dense dimension	Data needs to be recalculated. The data remains associated with the original member name, but, because the parent of the shared member may have depended on the child data, recalculation is needed. No restructure.	No
Add member to sparse dimension	Data for the new member needs to be loaded or calculated to derive new values. Essbase restructures the index.	Yes. Restructure deferred.
Add member to dense dimension	Data for the new member needs to be loaded or calculated to derive new values. Data needs to be recalculated. Essbase restructures the data files to reflect a changed block size. Essbase restructures the index.	Yes. Restructure deferred.
Add member to attribute dimension	None	No
Add shared member to sparse or dense dimension	Data needs to be recalculated. The new shared member affects the consolidation to its parent. No restructure.	No
Move regular member within a sparse dimension	Data needs to be recalculated to reflect changes in consolidation. Essbase restructures the index file.	No. Essbase restructures the index file, overriding incremental restructure.

Table 86: How Actions Affect Databases and Restructuring (Continued)

Action	Calculation and Standard Restructure Effects	Incremental Restructuring Applies? (If Enabled)
Move regular member within a dense dimension	Data needs to be recalculated to reflect changes in consolidation. Essbase restructures both index and data files.	Yes. Restructure deferred.
Move an attribute dimension member	None	No
Other Member-Related Changes		
Change a member alias or add an alias to a member	None	No
Rename member	None	No
Change member formula	Data needs to be recalculated to reflect formula changes. No restructure.	No
Dynamic Calculation-Related Changes		
Define Dynamic Calc member as Dynamic Calc And Store	For dense dimension members: Essbase restructures both index and data files. For sparse dimension members: no restructure.	Yes. Restructure deferred.
Define Dynamic Calc And Store member as Dynamic Calc	None	No
Define regular dense dimension member as Dynamic Calc And Store	None	No
Define regular dense dimension member as Dynamic Calc	Essbase restructures both index and data files.	Restructure deferred.

Table 86: How Actions Affect Databases and Restructuring (Continued)

Action	Calculation and Standard Restructure Effects	Incremental Restructuring Applies? (If Enabled)
Define sparse dimension Dynamic Calc And Store member or Dynamic Calc member as regular member	No restructure.	No
Define sparse dimension regular member as Dynamic Calc or Dynamic Calc And Store	Essbase restructures both index and data files.	Yes. Restructure deferred.
Define dense dimension Dynamic Calc And Store member as regular member	No restructure.	No
Define dense dimension Dynamic Calc member as regular member	Essbase restructures both index and data files.	Yes. Restructure deferred.
Define dense dimension regular member as Dynamic Calc member	Essbase restructures both index and data files.	Yes. Restructure deferred.
Add, delete, or move sparse dimension Dynamic Calc member	Essbase restructures only index files.	For member add or delete, restructure is deferred. For member move, Essbase restructures only index files, overriding incremental restructure.
Add, delete, or move sparse dimension Dynamic Calc And Store member	Essbase restructures only index files.	For member add, restructure deferred. For member move or delete, Essbase restructures only index files (overrides incremental restructure).
Add, delete, or move dense dimension Dynamic Calc And Store member	Essbase restructures both index and data files.	No

Table 86: How Actions Affect Databases and Restructuring (Continued)

Action	**Calculation and Standard Restructure Effects**	**Incremental Restructuring Applies? (If Enabled)**
Add, delete, or move dense dimension Dynamic Calc member	No restructure.	No
Property and Other Changes		
Change dense-sparse property	Data needs to be recalculated. Essbase restructures both index and data files.	Essbase restructures both index and data files overriding incremental restructure.
Change label only property	Data needs to be recalculated. Essbase restructures both index and data files.	Restructure deferred.
Change shared member property	Data needs to be recalculated to reflect the changed data value of the child. Essbase restructures both index and data files.	Restructure deferred.
Change properties other than dense-sparse, label, or shared	Data may need to be recalculated to reflect changed consolidation properties, such as changing time balance from first to last.	No
Change the order of two sparse dimensions	No calculation or data load impact. Essbase restructures the index.	Essbase restructures the index, overriding incremental restructure.
Change the order of dimensions	Data needs to be recalculated. Essbase restructures both index and data files.	Essbase restructures both index and data files (overrides incremental restructure).
Change the order of attribute dimensions	None	No
Create, delete, clear, rename, or copy an alias table	None	No

Table 86: How Actions Affect Databases and Restructuring (Continued)

Action	Calculation and Standard Restructure Effects	Incremental Restructuring Applies? (If Enabled)
Import an alias table or set a member alias	None	No
Change the case-sensitive setting	None	No
Name a level and generation	None	No
Create, change, or delete a user-defined attribute	None	No

Chapter 50

Optimizing Data Loads

Loading a large data source into a Essbase database can take hours. You can speed up the data loading process by improving two areas:

- Minimizing the time spent reading and parsing the data source
- Minimizing the time spent reading and writing to the database

This chapter contains the following sections:

Understanding Data Loads

To optimize data load performance, you must think in terms of database structure. Essbase loads data block by block. For each unique combination of sparse dimension members, one data block contains the data for all the dense dimension combinations, assuming there is at least one cell containing data. For faster access to block locations, Essbase uses an *index*. Each entry in the index corresponds to one data block. For further explanation of how sparse and dense dimensions affect database structure, see Chapter 3, “Basic Architectural Elements.”

When Essbase loads a data source, it processes the data in three main stages:

- Input stage: Essbase reads a portion of the data source.
- Preparation stage: Essbase arranges the data in preparation for putting it into blocks.
- Write stage: Essbase puts the data into blocks in memory and then writes the blocks to disk, finding the correct block on the disk by using the index, which is composed of pointers based on sparse intersections.

This process is repeated until all data is loaded. By using one or more processing threads in each stage, Essbase can perform some processes in parallel. For more information about parallel data load processing, see "Managing Parallel Data Load Processing" on page 1365.

All examples in this chapter assume that you are already familiar with the discussions about data sources in Chapter 20, "Introducing Data Loading."

Grouping Sparse Member Combinations

The most effective strategy to improve performance is to minimize the number of disk I/Os that Essbase must perform while reading or writing to the database. Because Essbase loads data block by block, organizing the source data to correspond to the physical block organization reduces the number of physical disk I/Os that Essbase must perform.

Arrange the data source so that records with the same unique combination of sparse dimensions are grouped together. This arrangement corresponds to blocks in the database.

The examples in this chapter illustrate various ways you can organize the data following this strategy. These examples use a subset of the Sample Basic database, as shown in Table 87.

Table 87: Dimensions and Values for Examples

Sparse, Nonattribute Dimensions	Dense Dimensions
Scenario (Budget, Actual)	Measures (Sales, Margin, COG, Profit)
Product (Cola, Root Beer)	Year (Jan, Feb)
Market (Florida, Ohio)	

Note: Because you do not load data into attribute dimensions, they are not relevant to this discussion even though they are sparse.

First, consider the data shown in Figure 540. Because it is not grouped by sparse-dimension member combinations, this data has not been sorted for optimization. As Essbase reads each record, it must deal with different members of the sparse dimensions.

Figure 540: Non-Optimized Sequence of Source Data

```
Jan
Actual    Cola          Ohio      Sales    25
Budget    "Root Beer"   Florida   Sales    28
Actual    "Root Beer"   Ohio      Sales    18
Budget    Cola          Florida   Sales    30
```

This data loads slowly because Essbase accesses four different blocks instead of one.

From the same Sample Basic database, Figure 541 shows different records sorted by a unique combination of sparse-dimension members: Actual -> Cola -> Ohio. Essbase accesses only one block to load these four records.

Figure 541: Optimally-Organized Source Data

```
Actual     Cola    Ohio    Jan   Sales      25
Actual     Cola    Ohio    Jan   Margin     18
Actual     Cola    Ohio    Jan   COGS       20
Actual     Cola    Ohio    Jan   Profit      5
```

You can use a data source that loads more than one cell per record. Make sure that records are grouped together by unique sparse-dimension member combinations, then order the records so that the dimension in the record for which you provide multiple values is a dense dimension.

Figure 542 uses a header record to identify the members of the Measures dimension, which is a dense dimension. The data is sorted first by members of the dense dimension Year and grouped hierarchically by members of the other dimensions. Multiple values for the Measures dimension are provided on each record.

Figure 542: Source Data Sorted and Grouped by Dense Dimensions

```
                                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
```

The heading and first data line in this example provide the same data shown in four lines in Figure 541.

For more information about arranging data in source files before loading, see the “Rules for Free-Form Data Sources” on page 582 and “Rules for Rules File Data Sources” on page 576. For more information on dense and sparse dimensions, see Chapter 3, “Basic Architectural Elements.”

Making the Data Source As Small As Possible

Make your data source as small as possible. The fewer fields that Essbase reads in the data source, the less time is needed to read and load the data.

Group the data into ranges. Eliminating redundancy in the data source reduces the number of fields that Essbase must read before loading data values.

Figure 543 shows a file that is not organized in ranges. It includes unneeded repetition of fields. All values are Profit values. Profit needs to be included only at the beginning of the group of data applicable to it. This example contains 33 fields that Essbase must read in order to load the data values properly.

Figure 543: Data Source Without Ranges

```
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
```

Figure 544 shows the same file optimized by grouping members in ranges. By eliminating redundancy, this example contains only 23 fields that Essbase must read in order to load the data values properly.

Figure 544: Data Source Organized in Ranges

```
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 assigns the first value, 4, to Jan->New York->Cola; it assigns the next value, 3, to Jan->New York->Diet Cola and so on.

Although sorted efficiently, the data in Figure 542 still shows a lot of repetition that can slow down the load process. You could further optimize this data by grouping the data into ranges. The optimized data source shown in Figure 545 eliminates the redundant fields, thereby reducing processing time.

Figure 545: Source Data Sorted and Grouped in Ranges

```
                                  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
```

For more information about organizing source data into ranges, see "Formatting Ranges of Member Fields" on page 583.

Making Source Fields As Small As Possible

Making fields in a data source smaller enables Essbase to read and load the data in less time.

Make the fields in the data source as small as possible by:

- Removing excess white space in the data source. For example, use tabs instead of blank spaces.
- Rounding off computer-generated numbers to the precision you need.
  For example, if the data value has nine decimal points and you only care about two, round the number to two decimal points.
- Using `#MI` instead of `#MISSING`.

Positioning Data in the Same Order As the Outline

The index is organized in the same order as the sparse dimensions in the outline. To further optimize the data source, with the sparse data combinations in the data source grouped together, arrange the data so that sparse dimensions are in the same order as the outline.

Essbase pages portions of the index in and out of memory as requested by the data load or other operations. Arranging the source data to match the order of entries in the index speeds up the data load because it requires less paging of the index. Less paging results in fewer I/O operations.

Essbase uses the index cache size to determine how much of the index can be paged into memory. Adjusting the size of the index cache may also improve data load performance.

Note: If the index cache size is large enough to hold the entire index in memory, positioning data in the same order as the outline does not affect the speed of data loads.

For more information about setting the index cache size, see "Sizing the Index Cache" on page 1320.

Loading from the Essbase OLAP Server

Loading the data source from the Essbase OLAP Server is faster than loading from a client computer. To load a data source from the server, move the data source to the server computer and then start the load.

Loading data from the server improves performance because the data does not have to be transported over the network from the client computer to the server computer.

Managing Parallel Data Load Processing

The methods described earlier in this chapter give you the most substantial data load performance enhancements. If you haven't done so, you also need to carefully evaluate your processor speed and memory requirements and upgrade your computers to meet these requirements.

Another method to speed up data loads is to work with the Essbase parallel data load feature to optimize use of processor resources. The parallel data load feature recognizes opportunities to process data load tasks at the same time. Although some opportunities present themselves on single-processor computers, many more opportunities are available on multiple-processor computers.

To enable you to fine tune processor use for specific application and database situations, Essbase provides three `essbase.cfg` configuration settings: DLTHREADSPREPARE, DLTHREADSWRITE, and DLSINGLETHREADPERSTAGE.

Understanding Parallel Data Load Processing

When Essbase loads a data source, it works with a portion of data at a time, in stages. Essbase looks at each stage as a task and uses separate processing *threads* in memory to perform each task.

One form of parallel processing occurs when one thread takes advantage of processor resources that are left idle during another thread's wait time. For example, while a thread performs I/O processing, it must wait for the slower hardware to perform its task. While this thread waits, another thread can use the idle processor resource. Processing staged tasks in parallel can improve processor efficiency by minimizing idle time.

When computers have multiple processors, Essbase can perform an additional form of parallel processing. When a data load stage completes its work on a portion of data, it can pass the work to the next stage and start work immediately on another portion of data. Processing threads perform their tasks simultaneously on the different processors, providing even faster throughput.

Optimizing Parallel Data Load Processing

Even though Essbase uses parallel processing to optimize processor resources across the data load stages, there are still times when processor resources can be idle. To take advantage of these idle times, Essbase can further divide up record processing in the preparation and write stages. To tailor parallel processing to your situation, you can use the DLTHREADSPREPARE and DLTHREADSWRITE `essbase.cfg` settings to tell Essbase to use additional threads during these stages.

Setting Parallel Data Load Settings

As shown in Table 88, Essbase provides three `essbase.cfg` settings that enable you to manage parallel data load processing.

You can specify setting values that apply to all applications on the Essbase OLAP server or you can specify settings multiple times with different values for different applications and databases.

Table 88: Parallel Data Load essbase.cfg Settings

Setting	Description
DLTHREADSPREPARE	Specifies how many threads Hyperion Essbase may use during the data load stage that codifies and organizes the data in preparation to being written to blocks in memory
DLTHREADSWRITE	Specifies how many threads Hyperion Essbase may use during the data load stage that writes data blocks to the disk. High values may require allocation of additional data cache. See "Implications in Sizing the Data Cache" on page 1368.
DLSINGLETHREADPERSTAGE	Specifies that Hyperion Essbase uses a single thread per stage, ignoring the values in the DLTHREADSPREPARE and DLTHREADSWRITE settings

Only when the DLSINGLETHREADPERSTAGE setting is set to FALSE for the specific application and database being loaded does the data load process use the thread values specified in the DLTHREADSPREPARE and DLTHREADSWRITE settings.

See the *Technical Reference* in the `docs` directory for details about these settings and their parameters.

Implications in Sizing the Data Cache

The data cache is the memory area that is allocated to hold uncompressed data blocks. Each thread specified by the DLTHREADSWRITE setting uses an area in the data cache equal to the size of an expanded block.

Depending on the size of the block, the number of threads, and how much data cache is used by other concurrent operations during a data load, it may be possible to need more data cache than is available.In such circumstances, decrease the number of threads or increase the size of the data cache. See "Changing the Data Cache Size" on page 1326.

Testing Different Thread Values

While processing data loads, you can view processor activity. Different operating systems provide different tools for doing this. For example, the Task Manager in Windows/NT and Windows/2000 enables you to view processor and memory usage and processes. Among the tools available on UNIX are top and vmstat. You can also use third-party tools to view and analyze system utilization.

➤ To assess system usage during data load processing;

1. Start with the default parallel data load processing thread values whereby Essbase uses a single thread per stage.
2. Perform and time the data load.
3. Monitor the entire process, identifying the stages during which the processor may be idle.
4. Alter the `essbase.cfg` settings that are described in "Setting Parallel Data Load Settings" on page 1367.
5. Repeat the last three steps until you find values that provide the best performance.

Chapter

51

Optimizing Calculations

This chapter provides information on how to optimize the performance of Essbase calculations:

You can also use any or all of these features to optimize overall database calculations:

- Chapter 13, “Designing Partitioned Applications”
- Chapter 28, “Dynamically Calculating Data Values”
- Chapter 30, “Developing Calculation Scripts”
- Chapter 52, “Optimizing with Intelligent Calculation”

Designing for Calculation Performance

You can configure a database to optimize calculation performance.

The best configuration for your site depends on the nature and size of your database. Use these sections as guidelines only:

- “Block Size and Block Density” on page 1370
- “Order of Sparse Dimensions” on page 1372
- “Incremental Data Loading Considerations” on page 1372
- “Performance for Database Outlines with Two or More Flat Dimensions” on page 1373

Block Size and Block Density

A data block size of 8Kb to 100Kb provides optimal performance in most cases.

If data blocks are much smaller than 8Kb, the index is usually very large, forcing Essbase to write and retrieve the index from disk. This slows down calculation.

If data blocks are much larger than 100Kb, Intelligent Calculation does not work effectively. For more information on how intelligent calculation aids performance improvements, see Chapter 52, “Optimizing with Intelligent Calculation.”

To optimize calculation performance and data storage, you need to balance data block density and data block size. You can create balance by rearranging the dense and sparse dimension configuration of the database. Therefore, keep these suggestions in mind:

- Keep data block size between 8 Kb and 100 Kb with as high a block density as possible.
- Run test calculations of the most promising configurations of a database that contains representative data. Check the results to determine the configuration for the best calculation performance.

You can view information about a database, including the potential and actual number of data blocks and the data block size. Use any of these tools to view this information:

Tool	**Instructions**	**For more information**
Administration Services	Database Properties window > Statistics tab	*Essbase Administration Services Online Help*
Application Manager	Open Database > Information, select Statistics tab.	Application Manager help topic: Main Menu > Database Menu > Settings
ESSCMD	GETDBINFO	*Technical Reference* in the `docs` directory

Order of Sparse Dimensions

You may improve calculation performance by changing the order of standard (not attribute) sparse dimensions in the database outline. Order standard sparse dimensions by the number of members they contain, starting with the dimension that contains the fewest members. This arrangement provides a number of possible improvements, depending on your site:

- The calculator cache functions more effectively, providing approximately a 10% performance improvement if you have a database outline with a very large dimension (for example, a dimension containing 1000 members). For more information, see "Managing Caches to Improve Performance" on page 1403.
- Parallel calculation, if enabled, is more likely to be used if the standard sparse dimension with the most members is the last standard sparse dimension in the outline. For more information, see "Using Parallel Calculation" on page 1380.

Incremental Data Loading Considerations

Many people load data incrementally. For example, a company may load data each month for that month.

To optimize calculation performance when you load data incrementally, make the dimension tagged as time a sparse dimension. If the time dimension is sparse, a different data block contains data for each time period. When you load data by time period, Essbase accesses fewer data blocks because fewer blocks contain the relevant time period. Thus, if you have Intelligent Calculation enabled, only the data blocks marked as dirty are recalculated.

For example, if you load data for March, only the data blocks for March are updated. The data blocks for January and February do not change. With Intelligent Calculation enabled, Essbase recalculates only the data blocks for March and March's dependent parents.

However, making the time dimension sparse when it is naturally dense may significantly increase the size of the index, creating possibly slower performance due to more physical I/O activity to accommodate the large index.

If the dimension tagged as time is dense, you still receive some benefit from Intelligent Calculation when you do a partial data load for a sparse dimension. For example, if Product is sparse and you load data for one product, Essbase recalculates only the blocks affected by the partial load, even though time is dense and Intelligent Calculation is enabled.

For more information on Intelligent Calculation, see Chapter 52, "Optimizing with Intelligent Calculation."

Performance for Database Outlines with Two or More Flat Dimensions

Calculation performance may be affected if a database outline has two or more flat dimensions. A flat dimension has very few parents and each parent has many thousands of children; in other words flat dimensions have many members and few levels.

You can improve performance for outlines with two or more flat dimensions by making any of these changes:

- Adding intermediate levels to the database outline.
- Using the SET CALCHASHTBL command in a calculation script to optimize outline calculation, or adding the CALCOPTCALCHASHTBL setting to the configuration file (`essbase.cfg`) to optimize outline calculation. Hash table settings and commands are ignored for parallel calculation.

  Note: If you use either SET CALCHASHTBL or CALCOPTCALCHASHTBL, set the maximum amount of memory that you want the calculator hash table to use with the CALCHASHTBLMEMORY setting in the configuration file (`essbase.cfg`). For more information, see the *Technical Reference* in the `docs` directory.

Monitoring and Tracing Calculations

You can display information in the application log about how Essbase is calculating the database by using the following commands in a calculation script:

- "SET MSG SUMMARY and SET MSG DETAIL" on page 1374
- "SET NOTICE" on page 1374

SET MSG SUMMARY and SET MSG DETAIL

You can use the SET MSG SUMMARY and SET MSG DETAIL calculation commands in a calculation script to do the following:

- Display calculation settings, for example, whether completion notice messages are enabled
- Provide statistics on the number of data blocks created, read, and written
- Provide statistics on the number of data cells calculated

The SET MSG DETAIL command also provides a detailed information message every time Essbase calculates a data block. SET MSG DETAIL is useful for reviewing the calculation order of data blocks and for testing intelligent recalculations.

CAUTION: Because the SET MSG DETAIL command causes a high processing overhead, use it only during test calculations.

SET MSG SUMMARY causes a processing overhead of approximately 1% to 5%, depending on database size, and is therefore appropriate for all calculations.

For more information on SET MSG SUMMARY and SET MSG DETAIL, see the *Technical Reference* in the `docs` directory.

SET NOTICE

You can use the SET NOTICE calculation command in a calculation script to display calculation completion notices that tell you what percentage of the database has been calculated. You can use the SET MSG SUMMARY command with the SET NOTICE command to show calculation progress between completion notices. Completion notices do not significantly reduce calculation performance, except when used with a very small database. For more information on SET NOTICE, see the *Technical Reference* in the `docs` directory.

Simulating Calculations

You can simulate a calculation using SET MSG ONLY in a calculation script. A simulated calculation produces results that help you analyze the performance of a real calculation based on the same data and outline.

By running a simulated calculation with a command like SET NOTICE HIGH, you can mark the relative amount of time each sparse dimension takes to complete. Then, by performing a real calculation on one or more dimensions, you can estimate how long the full calculation would take, because the time a simulated calculation takes to run is proportional to the time that same actual calculation takes to run.

For example, if the caclulation started at 9:50:00 AM, the first notice was time-stamped at 09:50:10 AM, and the second at 09:50:20 AM, you'd know that each of these parts of the calculation took ten seconds. If you then ran a real calculation on only the first portion and noted it took 30 seconds to run, you'd know that the other portions would also take 30 seconds. If there were only two messages total, then you'd know that the real calculation would take approximately 60 seconds (20 /10 * 30 = 60 seconds).

In this manner, you can estimate the length of time it takes a calculation to run using a simulated calculation.

Use these sections to learn how to perform a simulated calculation and how to analyze results from a simulated calculation:

- "Performing a Simulated Calculation" on page 1376
- "Notes About Using SET MSG ONLY" on page 1378
- "Changing Your Outline Based on Results" on page 1378

Performing a Simulated Calculation

➤ To perform a simulated calculation, use this procedure:

1. Create a data model using all the dimensions and level of detail about which you want information.
2. Load all the data. This procedure only calculates data loaded in the database.
3. Create a calculation script with these entries:

   ```
   SET MSG ONLY;
   SET NOTICE HIGH;
   CALC ALL;
   ```

   If you are using dynamic calculations on dense dimensions, substitute the `CALC ALL` command with the specific dimensions that you need to calculate, for example `CALC DIM EAST`.

   Note: If you try to validate the script, Essbase reports an error. You can disregard this error.

4. Run the script.
5. Find the first sparse calculation message in the application log and note the time in the message.
6. Note the time for each subsequent message.
7. Actually calculate the dense dimensions of the model that are not being dynamically calculated:

   ```
   CALC DIM (DENSE_DIM1, DENSE_DIM2, …);
   ```

8. Calculate the sparse dimensions of the model:

   ```
   CALC DIM (SPARSEDIM1, SPARSEDIM2, …);
   ```

9. Project the intervals at which notices will occur, then verify against sparse calculation results. You can then estimate how long a calculation will take.

➤ To estimate the total time a calculation will take, use this procedure:

1. Note the times of all the intervals between application log messages generated by SET NOTICE HIGH. See Table 89 on page 1377 for an example.
2. Use this calculation to estimate the time for a real calculation:

   Total time required for simulated calculation, divided by the first simulated calculation notice interval, multiplied by the first real calculation time interval.

Table 89: Sample Intervals Between Log Messages

Calculation Notice Number	Simulated Calculation Time Interval	Sparse dimension Calculation Interval
1	7 seconds	45 seconds
2	5 seconds	
3	6 seconds	
4	3 seconds	
5	4 seconds	
6	2 seconds	
7	6 seconds	
8	4 seconds	
9	3 seconds	
10	3 seconds	
Total	43 seconds	

In this example, 43 / 7 * 45 = 276.4 seconds, so the real calculation should take 276.4 seconds.

Notes About Using SET MSG ONLY

This calculation time estimating technique should be validated against later CALC NOTICE intervals. The results of this technique will vary because of this chain of influences:

1. Blocks differ in block density through the real aggregation process, therefore
2. The rate at which Essbase writes blocks to the disk differ, therefore
3. The rate at which blocks are processed in cache differs, therefore
4. Actual results may differ from the predicted calculation time.

This estimating technique, however, improves significantly once Essbase has reached about 30-40 percent of the simulated calculations (30-40 percent of the messages generated by SET NOTICE HIGH).

There is one more factor that will make actual results diverge significantly from predicted, and that is when the model contains 1-2 sparse dimensions that are very large in relation to the other sparse dimensions. This will tend to make the performance of the last 10 to 20th percentile CALC NOTICE much longer than would be expected. Many times, this is the result of 2-3 shared rollups in larger dimensions.

Excluding these issues, the simulated calculation should return a time accurate to about 5%.

Changing Your Outline Based on Results

Once you have estimated and analyzed a simulated calculation, you can make changes in your outline to improve performance.

From top to bottom in your outline, order sparse dimensions to create the fewest percentage increases in upper blocks:

- Level 0 blocks following full model load 100,000
- Upper level blocks after aggregating only sparse dimension 1: 1,000,000
- Upper level blocks after aggregating only sparse dimension 2: 3,000,000
- Upper level blocks after aggregating only sparse dimension 3: 10,000,000
- Upper level blocks after aggregating only sparse dimension 4: 300,000
- Upper level blocks after aggregating only sparse dimension 5: 5,700,000

For example:

- #4 (members = 10,000, 4 levels)
- #1 (members = 500, 2 levels)
- #2 (members = 100, 4 levels)
- #5 (members = 10,000, 4 levels)
- #3 (members = 20, flat)

Use the simulated calculation to generate the upper block count. These numbers may be accurate despite actual dimension sizes as noted next to the items above.

CAUTION: The largest count of dimension members is not always a good predictor.

Estimating Calculations

Given the current number of blocks in a database, you can estimate the number of blocks that will be produced by a CALC ALL.

➤ To estimate the size of a calculation, use this procedure:

1. Load data and issue a CALC ALL command.
2. Log into ESSCMD and select an application and database if you have not already.
3. Issue the ESTIMATEFULLDBSIZE command and note the value returned, the number of blocks.
4. Multiply the number returned by the average size of the blocks in the database. If you are not sure of the average size of blocks for your database, load a small sample of data, perform a CALC ALL, and check the average size of blocks.
5. Results are accurate to a precision of plus or minus 10%.

Be aware of these conditions when you use ESTIMATEFULLDBSIZE:

- You must use ESTIMATEFULLDBSIZE after a CALC ALL. Any other calculation will not produce accurate results.
- You can obtain accurate results with formulas only if they are on sparse dimensions. Estimates on databases that have formulas on dense dimensions are not accurate.
- You cannot obtain accurate results with top down calculations on any member in combination with a lock on data (committed access).
- If you need to estimate partitions, you must use the command ESTIMATEFULLDBSIZE on every partition and add up the results. If you issue this command against just the source database, the estimate will include only the data on that server.

For more information, see the *Technical Reference* in the `docs` directory, in the ESSCMD section for command ESTIMATEFULLDBSIZE.

Using Parallel Calculation

This section provides the information that you need to understand parallel calculation and to decide whether it will improve performance for your site. This section also presents instructions for enabling parallel calculation:

- "Understanding Parallel Calculation" on page 1380
- "Checking Current Parallel Calculation Settings" on page 1388
- "Enabling Parallel Calculation" on page 1389
- "Identifying Additional Tasks for Parallel Calculation" on page 1390

Understanding Parallel Calculation

Essbase provides two ways of invoking a calculation:

- The calculation may be implicitly specified by the outline itself.
- The calculation may be explicitly specified by a calculation script you create, containing formulas and calculation instructions.

Regardless of how a calculation is triggered, Essbase can execute the calculation in one of two modes:

- *Serial calculation* is the default. With serial calculation, each calculation pass is scheduled to run on a single processor. If invoked from a calculation script, the calculations are executed sequentially in the order that they appear in the calculation script.
- *Parallel calculation* breaks down each calculation pass into sub-tasks. The sub-tasks that can run independently of each other are scheduled to run simultaneously on up to four threads. Each thread may be on a different processor.

To change from the default serial calculation to parallel calculation, use either of these methods:

- Change at most two configuration settings and restart the server.
- Add an instruction to your calculation script.

See "Enabling Parallel Calculation" on page 1389 for more details.

CAUTION: Be sure to read all of this chapter before following the instructions in "Enabling Parallel Calculation" on page 1389.

Use the rest of this section to understand parallel calculation in detail:

- "Essbase Analysis of Feasibility" on page 1381
- "Requirements for Parallel Calculation" on page 1382
- "Relationship to Other Essbase Features" on page 1383

Essbase Analysis of Feasibility

Essbase evaluates whether using parallel calculation is possible before each calculation pass for which you have enabled parallel calculation.

Essbase analyzes your outline and the calculation requested for each calculation pass. Remember that a single calculation may require more than one pass. A number of situations may create the need for more than one pass, including dynamic calculation, the presence of a member tagged as two-pass, or calculations that create certain kinds of inter-dependencies. For more information, see "Calculation Passes" on page 764.

If Essbase determines that parallel calculation is possible, Essbase splits the calculation into smaller tasks that are independent of one another. During the calculation, Essbase performs these smaller tasks at the same time.

However, Essbase uses serial calculation even if parallel calculation is enabled if complex interdependencies between formulas that participate in the pass are present, rendering parallel calculation impossible.

Requirements for Parallel Calculation

Your outline structure and application design determine whether enabling parallel calculation can improve calculation performance. Before you enable parallel calculation, review this list of requirements. If your site does not meet these requirements, you may not get the full benefit of parallel calculation:

- Use the uncommitted access isolation level. Parallel calculation is not supported if you use the committed access isolation level. See "Isolation Level" on page 1388 for details.
- One or more formulas present in a calculation may prevent Essbase from using parallel calculation even if it is enabled. For a description of formulas that may force serial calculation regardless of parallel calculation settings, see "Formula Limitations" on page 1384.
- Calculation tasks are usually generated along the last sparse dimension of an outline. Order the sparse dimensions in an outline from smallest to largest, based on actual size of the dimension as reported by the ESSCMD command GETDBSTATS. This recommendation is consistent with recommendations for optimizing calculator cache size and consistent with other outline recommendations. For situations that may need to use more than the last sparse dimension, see "Identifying Additional Tasks for Parallel Calculation" on page 1390.

- Parallel calculation is effective on non-partitioned applications and these partitioned applications:
  - Replicated partitions
  - Linked partitions
  - Transparent partitions if the calculation occurs at the target database. The number of sparse dimensions specified by CALCTASKDIMS in the configuration file `essbase.cfg` or by SET CALCTASKDIMS in a calculation script must be set at 1 (the default value). See "Transparent Partition Limitations" on page 1385 for details about transparent partitions, and see "Identifying Additional Tasks for Parallel Calculation" on page 1390 for details about CALCTASKDIMS or SET CALCTASKDIMS.
- If you have selected incremental restructuring for a database and you have made outline changes that are pending a restructure, do not use parallel calculation. Unpredictable results may occur.

Relationship to Other Essbase Features

Use these sections to help you understand the relationship between parallel calculation and other Essbase functionality:

- "Retrieval Performance" on page 1384
- "Formula Limitations" on page 1384
- "Calculator Hash Tables" on page 1384
- "Calculator Cache" on page 1385
- "Transparent Partition Limitations" on page 1385
- "Restructuring Limitations" on page 1386
- "Commit Threshold Adjustments" on page 1386
- "Isolation Level" on page 1388

Retrieval Performance

Placing the largest sparse dimension at the end of the outline for maximum parallel calculation performance may slow retrieval performance. See "Designing an Outline to Optimize Performance" on page 124 for more information.

Formula Limitations

The presence of some formulas may force serial calculation. These formula placements are likely to force serial calculation:

- A formula on a dense member, including all stored members and any Dynamic Calc members upon which the stored member may be dependent, causes a dependence on a member in the dimension used to identify tasks for parallel calculation.
- A formula contains references to variables declared in a calculation script using @VAR, @ARRAY, or @XREF.
- A formula on a member causes a circular dependence. For example, member A has a formula referring to member B, and member B has a formula referring to member C, and member C has a formula referring to member A.
- A formula on a dense or sparse member with a dependency on the member or members from the dimension used to identify tasks for parallel processing.

If you need to use a formula that might prevent parallel calculation, you can either mark the formula's member as Dynamic Calc or exclude it from the scope of the calculation. To check if a formula is preventing parallel calculation, check the application log. See "Monitoring Parallel Calculation" on page 1391 for details.

Calculator Hash Tables

Calculator hash tables are not used during parallel calculation. You can leave any of these items in your `essbase.cfg` configuration file or calculation script, but they are ignored during a parallel calculation:

- SET CALCHASHTBL (calculation script command)
- CALCOPTCALCHASHTBL (configuration file setting)
- CALCHASHTBLMEMORY (configuration file setting)

Calculator Cache

At the start of a calculation pass, Essbase checks the calculator cache size and the degree of parallelism, then uses the calculator cache bitmap option appropriate for maximum performance. Therefore, the option used for parallel calculation may be different from the one used for serial calculation.

For example, assume Essbase performs a serial calculation using multiple bitmaps and with a single anchoring dimension. That same calculation, without explicitly changing the calculator cache size, with a parallel calculation might be performed with only a single bitmap and single anchoring dimension.

You can determine the calculator cache mode which controls the bitmap options by checking the application log at the start of each calculation pass for an entry similar to this:

```
Multiple bitmap mode calculator cache memory usage has a limit
of [50000] bitmaps.
```

For more information about the calculator cache and calculator cache bitmaps, see "Sizing the Calculator Cache" on page 1327.

Transparent Partition Limitations

Parallel calculation with transparent partitions has these limitations:

- You cannot use parallel calculation across transparent partitions unless the calculation occurs at the target.
- You must set CALCTASKDIMS or SET CALCTASKDIMS to 1 (the default) so that there is only one anchoring dimension.
- You must increase the calculator cache so that multiple bitmaps can be used. You can identify the calculator cache mode that controls the bitmap options by checking the application log at the start of each calculation pass for an entry similar to this:

  ```
  Multiple bitmap mode calculator cache memory usage has a limit
  of [50000] bitmaps.
  ```

  For more information about changing the calculator cache size, see "Sizing the Calculator Cache" on page 1327.

Restructuring Limitations

Do not use parallel calculation if you have selected incremental restructuring. Parallel calculation does not support incremental restructuring.

Commit Threshold Adjustments

Essbase checks the commit threshold specified by the database setting “Number of blocks before internal commit.” If the setting requires less than 10 MB of data be written before an internal commit, then Essbase automatically increases the commit threshold for the duration of the calculation pass to 10 MB. If the setting is greater than 10 MB, Essbase uses the setting value.

Essbase writes a message to the application log noting the temporary increase if it occurs.

If you can allocate more than 10 MB extra disk space for calculation, consider increasing your commit threshold value to a very large number for better performance.

➤ To view the current threshold, use any of these procedures:

Tool	Instructions	For more information
Administration Services	Click the Transactions tab on the Database Properties window	*Essbase Administration Services Online Help*
MaxL	**display database** *dbs_name*	*Technical Reference* in the `docs` directory, list of MaxL statements
ESSCMD	GETDBINFO: Number of blocks modified before internal commit	*Technical Reference* in the `docs` directory, GETDBINFO
Application Manager	Database > Settings > Transactions tab, Commit Blocks value	Application Manager Online Help

➤ To modify the commit threshold, use any of these procedures:

Tool	**Instructions**	**For more information**
Administration Services	Click the Transactions tab on the Database Properties window	*Essbase Administration Services Online Help*
MaxL	**alter database** *dbs_name* **set implicit_commit after <n> blocks**	*Technical Reference* in the `docs` directory, list of MaxL statements
ESSCMD	SETDBSTATEITEM 21	"Specifying settings with ESSCMD" on page 1143
Application Manager	Database > Settings > Transactions tab, Commit Blocks value	"Specifying Settings with Application Manager" on page 1142

For more information about commit thresholds, see "Uncommitted Access" on page 1137.

Isolation Level

You must use uncommitted mode for parallel calculation.

➤ To set the isolation level to uncommitted mode, use any of these procedures:

Tool	Instructions	For more information
Administration Services	Click the Transactions tab on the Database Properties window	*Essbase Administration Services Online Help*
MaxL	**alter database** *dbs_name* **disable committed_mode**	*Technical Reference* in the `docs` directory, list of MaxL statements
ESSCMD	SETDBSTATEITEM 18	"Specifying settings with ESSCMD" on page 1143
Application Manager	Database > Settings > Transactions tab	"Specifying Settings with Application Manager" on page 1142

See "Isolation Levels" on page 1132 for more information about isolation levels and uncommitted mode.

Checking Current Parallel Calculation Settings

You can check either the server configuration file or the calculation script you plan to use to see if parallel calculation is enabled.

➤ To check if parallel calculation has already been enabled in the server configuration file:

1. Open the server configuration file `essbase.cfg` with a text editor.
2. Search for the parameter CALCPARALLEL, and check its specified value. A value of 0 means parallel calculation is not enabled, and values 1–4 mean that parallel calculation is enabled. The number of threads that can simultaneously perform tasks to complete a calculation is specified by the value 1–4. See the *Technical Reference* in the `docs` directory for details.

➤ To check if a calculation script sets parallel calculation, look for the SET CALCPARALLEL command. Review the script carefully, as the script may enable or disable parallel calculation more than once.

Enabling Parallel Calculation

To use parallel calculation, enable it at the server level, application level, or database level using either of these methods:

- Add or edit the appropriate configuration settings to the `essbase.cfg` file. See CALCPARALLEL and CALCTASKDIMS in the *Technical Reference* in the `docs` directory of your Essbase installation, in the configuration settings section.
- Add the appropriate calculation commands to a calculation script. See SET CALCPARALLEL and SET CALCTASKDIMS in the *Technical Reference* in the `docs` directory of your Essbase installation, in the calculator commands section.

Parallel calculation settings use standard precedence rules:

- The database setting takes precedence over the application setting
- The application setting takes precedence over the server setting.

Setting parallel calculation at the server level enables it for all calculations performed on every application and database on the server. You can disable parallel calculation for individual applications or databases by setting it at the server level in the configuration file, then adding application- or database-specific entries in a calculation script.

➤ To enable parallel calculation, use this procedure:

1. Check the current status using the procedure in "Checking Current Parallel Calculation Settings" on page 1388 if you plan to enable parallel calculation in the configuration file, to see if an entry already exists.
2. Add or modify CALCPARALLEL to the server configuration file, or add SET CALCPARALLEL to a calculation script.
3. If needed, enable Essbase to use more than the one sparse dimension to identify tasks for parallel calculation, using the procedure in "Identifying Additional Tasks for Parallel Calculation" on page 1390.

4. Restart the server if you added entries to the configuration file.
5. Run the calculation.

Hyperion recommends that you set the value of CALCPARALLEL to be one less than the number of processors available for calculation. This extra processor can then be used by either the operating system or by the Essbase process responsible for writing out dirty blocks from the cache.

Tip: You can combine the use of CALCPARALLEL and SET CALCPARALLEL if your site requires it. For example, you could set CALCPARALLEL off at the server level, then use a calculation script to enable and disable parallel calculation as often as needed.

Identifying Additional Tasks for Parallel Calculation

By default, Essbase uses the last sparse dimension in an outline to identify tasks that can be performed concurrently. But the distribution of data may cause one or more tasks to be empty, that is, there are no blocks to be calculated in the part of the database identified by a task. This can lead to uneven load balancing, thus reducing the effectiveness of parallel calculation.

To resolve this situation, you can enable Essbase to use additional sparse dimensions in the identification of tasks for parallel calculation. For example, if you had a FIX statement on a member of the last sparse dimension, you could include the next-to-last sparse dimension from the outline as well. Since each unique member combination of these two dimensions is identified as a potential task, more and smaller tasks would be created, increasing the opportunities for parallel processing and providing better load balancing.

➤ To increase the number of sparse dimensions used to identify tasks for parallel calculation, use this procedure:

1. If you are not sure, verify if parallel calculation is already enabled. See "Checking Current Parallel Calculation Settings" on page 1388 for details. Without CALCPARALLEL (or SET CALCPARALLEL in a calculation script), CALTASKDIMS has no effect.
2. Add or modify CALCTASKDIMS in the `essbase.cfg` server configuration file, or use the calculation script command SET CALCTASKDIMS at the top of the script. See the *Technical Reference* in the `docs` directory for details.
3. Restart Essbase if you add or modify CALCTASKDIMS in the `essbase.cfg` server configuration file. If you are using a calculation script, run the script.

Note: In some cases, Essbase uses a lower number of dimensions to identify tasks than is specified by CALCTASKDIMS or SET CALCTASKDIMS. See the *Technical Reference* in the `docs` directory of your Essbase installation for details.

Monitoring Parallel Calculation

You can view events related to parallel calculation in the application log:

For instructions about how to view the application log in Application Manager, see "Viewing the Server and Application Logs" on page 1225.

For each calculation pass, Essbase writes several types of information to the application log to support parallel calculation:

- If you have enabled parallel calculation and Essbase has determined that parallel calculation can be performed, Essbase writes a message in the application log:

  ```
  Calculating in parallel with n threads
  ```

  n represents the number of concurrent tasks specified in CALCPARALLEL or SETCALCPARALLEL.

- For each formula that prevents parallel calculation (forces serial calculation), Essbase writes a message to the application log:

  ```
  Formula on mbr memberName prevents calculation from running
  in parallel.
  ```

  memberName represents the name of the member where the relevant formula exists. You can look in the application log for such messages and consider tagging the relevant member or members as Dynamic Calc if possible so they do not feature in the calculation pass.

- Essbase writes a message to the application log specifying the number of tasks that can be executed concurrently at any one time (based on the data, not the value of CALCPARALLEL or SETCALCPARALLEL):

  ```
  Calculation task schedule [576,35,14,3,2,1]
  ```

  This example message indicates that 576 tasks can be executed concurrently. After these tasks complete, 35 more can be performed concurrently, and so on.

  The benefits of parallel calculation is greatest in the first few steps, and then the benefits taper off because there are fewer and fewer tasks being performed concurrently.

The degree of parallelism depends on the number of tasks shown in the task schedule. The greater the number, the more tasks that can run in parallel, and thus the greater the performance gains.

- Essbase writes a message to the application log indicating how many of the tasks are empty (contain no calculations):

  ```
  [Tue Nov 27 12:30:44 2001]Local/CCDemo/Finance/essexer/
  Info(1012681) Empty tasks [91,1,0,0]
  ```

  In the example log message above, Essbase indicates that 91 of the tasks at level zero were empty.

  If the ratio of empty tasks to the tasks specified in the task schedule is greater than 50, then parallelism may not be giving you improved performance, because of the high sparsity in the data model.

  You can change dense/sparse assignments to reduce the number of empty tasks, and increase the performance gains from parallel calculation.

Using Formulas

You may achieve significant improvements in calculation performance by careful use of formulas in the database outline. For example, you may achieve improved calculation performance by placing formulas on members in the database outline instead of placing the formulas in a calculation script. For more information, see Chapter 25, “Developing Formulas.”

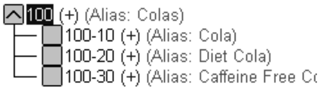
➤ To increase performance with formulas, use any of these methods:

- “Consolidation” on page 1393
- “Simple Formulas” on page 1393
- “Complex Formulas” on page 1394
- “Complex Formula Performance Example” on page 1395
- “Optimizing Formulas on Sparse Dimensions in Large Database Outlines” on page 1395
- “Improving Performance for Constants in a Sparse Dimension” on page 1396
- “Using Caution with a Cross-Dimensional Operator (->)” on page 1397

Consolidation

Using the database outline to roll up values is always more efficient than using a formula to calculate values. For example, consider the following consolidation on the Sample Basic database outline shown in Figure 546.

Figure 546: Consolidation on Sample Basic Outline



```
100 (+) (Alias: Colas)
  100-10 (+) (Alias: Cola)
  100-20 (+) (Alias: Diet Cola)
  100-30 (+) (Alias: Caffeine Free Co
```

Using outline consolidation is more efficient than applying the following formula to the Colas member:

```
100-10 + 100-20 + 100-30
```

Simple Formulas

If you use a simple formula, you can place it on either a sparse or a dense dimension without significantly affecting calculation performance if the block size is not unusually large. The bigger the block size, the more impact simple formulas have on calculation performance. For more information, see "Block Size and Block Density" on page 1370.

A simple formula is, for example, a ratio or a percentage. A simple formula meets all of these requirements:

- Does not reference values from a different dimension (sparse or dense); for example, Product->Jan.
- Does not use range functions, for example, @AVGRANGE, @MAXRANGE, @MINRANGE, or @SUMRANGE.
- Does not use relationship or financial functions; for example, @ANCESTVAL, @NEXT, @PARENTVAL, @SHIFT, @ACCUM, or @GROWTH. For a complete list of relationship and financial functions, see the *Technical Reference* in the `docs` directory.

For information on how formulas affect calculation performance, see "Understanding Bottom-Up and Top-Down Calculation" on page 1400.

Complex Formulas

If you use a complex formula, you can improve performance by following these guidelines:

- If possible, apply the formula to a member in a *dense* dimension.
- Use the FIX command in a calculation script to calculate only the minimum, required data blocks. See Chapter 30, “Developing Calculation Scripts.”
- Increase the density of the database (ratio of existing data blocks to possible data blocks). See “Block Size and Block Density” on page 1370.

A complex formula is any formula that meets any of these requirements:

- References a member or members in a different dimension (sparse or dense); for example, Product->Jan.
- Uses one or more range functions, for example, @AVGRANGE, @MAXRANGE, @MINRANGE, or @SUMRANGE.
- Uses relationship or financial functions; for example, @ANCESTVAL, @NEXT, @PARENTVAL, @SHIFT, @ACCUM, or @GROWTH. For a complete list of relationship and financial functions, see the *Technical Reference* in the `docs` directory.

Complex formulas create more calculation overhead and therefore slow performance when applied to sparse dimension members. This is true because the presence of complex formulas requires Essbase to perform calculations on possible data blocks related to the member with the complex formula (not just the existing blocks). The presence of relationship or financial functions on a sparse dimension member causes Essbase to perform calculations on all possible blocks with any sparse member combinations, increasing the overhead even more.

For more information about how complex formulas affect calculation performance, see “Understanding Bottom-Up and Top-Down Calculation” on page 1400.

Complex Formula Performance Example

When a complex formula, relationship function or financial function is attached to a sparse dimension member, the calculation overhead increases the higher the ratio of existing data blocks to possible data blocks, and therefore the slower the performance.

Two examples illustrate complex formula overhead:

- If a database had 90 existing data blocks and 100 potential blocks, the overhead for complex formulas would not be very large, not more than ten extra blocks to read and to possibly write values to cells in the data blocks.
- If a database had only 10 existing data blocks and 100 potential blocks, the overhead would be as much as ten times what it would be without the complex formula or relationship or financial function (depending on the particular outline structure and other factors), because as many as 90 extra blocks would be read and values possibly written to cells in the data blocks.

In all cases, the lower the ratio of existing data blocks to possible data blocks, the higher the calculation performance overhead.

Optimizing Formulas on Sparse Dimensions in Large Database Outlines

You can use the SET FRMLBOTTOMUP calculation command to optimize the calculation of formulas in sparse dimensions in large database outlines. With this command, you can force a bottom-up calculation on sparse member formulas that would otherwise be calculated top-down. For more information, see "Forcing a Bottom-Up Calculation" on page 1401.

Forcing a bottom-up calculation on a top-down formula enables efficient use of the CALC ALL and CALC DIM commands. For more information, review the discussions of the SET FRMLBOTTOMUP calculation command and the CALCOPTFRMLBOTTOMUP configuration setting in the *Technical Reference* in the `docs` directory.

Improving Performance for Constants in a Sparse Dimension

If you assign a constant to a member in a sparse dimension, Essbase automatically creates a data block for every combination of sparse dimension members that contains the member.

For example, assume that a member or a calculation script formula contains the following expression:

```
California = 120;
```

In this formula, California is a member in a sparse dimension and 120 is a constant value. Essbase automatically creates all possible data blocks for California and assigns the value 120 to all data cells. Many thousands of data blocks may be created. To improve performance, create a formula that doesn't create unnecessary values.

➤ To assign constants in a sparse dimension to only those intersections that require a value, use FIX in a manner similar to this example:

```
FIX(Colas,Misc,Actual)
California = 120;
ENDFIX
```

This example assigns the value 120 to any intersection of California (in the Market dimension), Actual (in the Scenario dimension), Misc (in the Measures dimension), Colas (in the Product dimension), and any member in the Year dimension, because a specific member of Year is not specified in the script.

In the Sample Basic database, Colas is a member of the sparse Product dimension, Actual is a member of the dense Scenario dimension, and Misc is a member of the dense Measures dimension. Essbase creates new data blocks for all combinations of the sparse members, California and Colas. Within the new blocks, Essbase sets Measures and Scenario values (other than those assigned the value 120) to `#MISSING`.

For more information about FIX, see the *Technical Reference* in the `docs` directory.

Using Caution with a Cross-Dimensional Operator (->)

Use caution when using a cross-dimensional operator (->) in the following situations:

- "On the Left Side of an Equation" on page 1397
- "In Equations in a Dense Dimension" on page 1398
- When doing a two-pass formula calculation on a dimension tagged as accounts. For more information about two-pass calculation, see "Using Two-Pass Calculation" on page 1406.

On the Left Side of an Equation

Replace formulas that use a cross-dimensional operator on the left side of an equation with a formula that uses FIX in a calculation script instead for faster performance.

For example, assume you want to increase the Jan -> Sales values by 5% in Sample Basic.To improve performance by calculating only the relevant combinations of members, use the FIX command in a calculation script:

```
FIX(Jan)
   Sales = Sales * .05;
ENDFIX
```

With the FIX command instead of a cross-dimensional operator, Essbase calculates the formula only for the specified member combinations, in this example, Jan.

Compare this technique to using the slower cross-dimensional operator approach. For the previous example, you would place this formula on the Sales member in the database outline:

```
Sales(Sales -> Jan = Sales -> Jan * .05;)
```

As Essbase cycles through the database, it calculates the formula for every combination of members.Thus Essbase calculates the formula for every member in the dimension tagged as time (Jan, Feb, Mar, etc.), even though only January members need to be calculated.

For more information on calculation scripts and the FIX command, see Chapter 30, "Developing Calculation Scripts" and the *Technical Reference* in the `docs` directory.

In Equations in a Dense Dimension

When you use a cross-dimensional operator in an equation in a dense dimension, Essbase does not automatically create the required blocks if both of these conditions apply:

- Resultant values are from a dense dimension.
- The operand or operands are from a sparse dimension.

When the blocks are not created automatically, performance is slower than necessary. You can prevent this performance issue by using a DATACOPY command in the calculation script that contains the relevant equation.

For example, assume this equation must be applied to a member in a dense dimension:

result = member -> operand

“result” is the member of a dense dimension and “operand” is the member of a sparse dimension.

Now assume you need to apply this equation in Sample Basic, to create budget sales and expense data from existing actual data. Sales and Expenses are members in the dense Measures dimension. Budget and Actual are members in the sparse Scenario dimension.

Create a calculation script for the above formulas using a DATACOPY command:

```
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 copies the data and creates the required blocks. Essbase creates blocks that contain the Budget values for each corresponding Actual block that already exists.

Compare this calculation script to the solution which has the slowest performance:

```
FIX(Budget)
  (Sales = Sales -> Actual * 1.1;
Expenses = Expenses -> Actual * .95;)
ENDFIX
```

The calculation script above does not create the required data blocks. Budget data values are not calculated for blocks that do not already exist. The resultant values are dense dimension members (Sales and Expenses). The operand is a sparse dimension member (Actual).

Other solutions

If you cannot use DATACOPY, you can avoid it without slowing performance by ensuring that the resultant members are not from a dense dimension:

```
FIX(Sales)
      Budget = Actual * 1.1;
ENDFIX
FIX(Expenses)
      Budget = Actual * .95;
ENDFIX
```

Or, you can use a member formula that contains the dense member equations:

```
FIX(Sales, Expenses)
  Budget (Sales = Sales -> Actual * 1.1;
  Expenses = Expenses -> Actual * .95;)
ENDFIX
```

Using Bottom-Up Calculation

A top-down calculation is less efficient than a bottom-up calculation because more blocks are calculated than is necessary. Although a top-down calculation is less efficient than a bottom-up calculation, top-down calculations are necessary in some cases to ensure that calculation results are correct.

Use these sections to understand bottom-up and top-down calculation, and then find instructions for forcing a bottom-up calculation only if it is appropriate for your site:

- "Understanding Bottom-Up and Top-Down Calculation" on page 1400
- "Forcing a Bottom-Up Calculation" on page 1401

Understanding Bottom-Up and Top-Down Calculation

Essbase uses one of two calculation methods to do a full calculation of a database outline: bottom-up calculation or top-down calculation. By default, Essbase does a bottom-up calculation of a database. However, if the database outline contains a complex member formula, Essbase performs a top-down calculation for that member.

Use this information to learn more about simple and complex formula interactions with bottom-up and top-down calculation:

- "Bottom-Up Calculations and Simple Formulas" on page 1400
- "Top-Down Calculations and Complex Formulas" on page 1400
- "A Complex Formula Example" on page 1401

Bottom-Up Calculations and Simple Formulas

For a bottom-up calculation, Essbase determines which data blocks need to be calculated before it calculates the database. Essbase then calculates only the blocks that need to be calculated. The calculation begins with the existing block with the lowest block number and works up through each block in number order until the last existing block is reached. For more information on block calculation order, see Chapter 27, "Defining the Calculation Order."

For simple formulas, Essbase does a bottom-up calculation to determine which blocks need to be calculated prior to running the full calculation. For example, for a simple formula on a member (such as `A = B + C`), A is calculated only if B or C exists in the database. That is, the dependency of the formula on B and C is known before the calculation is started.

Top-Down Calculations and Complex Formulas

Before starting a calculation, Essbase searches the database outline and marks complex formulas that require a top-down calculation, for example, a member formula that contains a cross-dimensional reference. When Essbase reaches a member with a top-down formula, it does a top-down calculation for that member.

When a formula on a member is complex, all possible blocks for the member must be examined to see if an existing block needs to be changed or a new block needs to be created; it is difficult to determine the dependency blocks may have on other blocks prior to the start of the calculation. The top-down method slows down calculation performance because Essbase must search for the appropriate blocks to calculate in order to execute the formula.

When a formula is compiled, if the formula is to be calculated top-down, Essbase logs a message in the application log file.

For more information about complex formulas, see "Complex Formulas" on page 1394.

A Complex Formula Example

Consider this complex formula:

```
A = B -> D + C -> D
```

Essbase must examine every possible combination of A to see whether B -> D or C -> D exists. Unlike a simple formula, the dependencies for a complex formula cannot be determined easily prior to the calculation, so Essbase uses a top-down calculation by default.

Forcing a Bottom-Up Calculation

If it is appropriate for your site, you can force a bottom-up calculation on a top-down formula. Use any of these methods to force a bottom-up calculation:

Tool	Instructions	For More Information
Calculation Function	@CALCMODE in a formula	*Technical Reference* in the `docs` directory
Calculation Script Command	SET FRMLBOTTOMUP	
`essbase.cfg` file setting	CALCOPTFRMLBOTTOMUP or CALCMODE	

Forcing a bottom-up calculation on a formula, which ordinarily increases performance over a top-down calculation, may produce results that are inconsistent with a top-down calculation if the formula contains complex functions (for example, range functions), or if the formula's dependencies are not straightforward.

CAUTION: Before changing the setting CALCOPTFRMLBOTTOMUP or using the calculation script command SET FRMLBOTTOMUP in a production environment, check the validity of calculation results by comparing results to a top-down calculation of the same data.

Designing Calculation Scripts for Performance

You may achieve significant improvements in calculation performance by carefully grouping formulas and dimensions in a calculation script. In this way, you can ensure that Essbase cycles through the data blocks in the database as few times as possible during a calculation.

Order commands in your calculation scripts to make the database calculation as simple as possible. Consider applying all formulas to the database outline and using a default calculation (CALC ALL). This method may improve calculation performance.

For more information about developing calculation scripts, see Chapter 30, “Developing Calculation Scripts.” For more information about calculation passes, see “Calculation Passes” on page 764

Managing Caches to Improve Performance

When calculating the database, Essbase uses approximately 30 bytes of memory per member in the database outline. So if the database has 5,000 members, Essbase needs approximately 150K of memory to calculate the database.

When you run concurrent calculations, each calculation uses separate memory space. For example, if you are running two calculation scripts concurrently, Essbase requires 60 bytes: 30 bytes per member, per script. Concurrent calculations do share the caches.

Note: You can avoid the excess memory use by combining calculation scripts, and get good performance by using parallel calculation with the single calculation script. For more information about parallel calculation, see "Using Parallel Calculation" on page 1380.

Essbase uses memory to optimize calculation performance, especially for large calculations. The amount of memory used is not controllable, except by altering the size of the database outline. However, you can ensure that the memory cache sizes enable Essbase to optimize the calculation.

Essbase uses four memory caches to coordinate memory usage:

- The calculator cache. Ensure that the calculator cache is large enough to optimize calculation performance.
- The index cache. If the database is large, the default index cache is not large enough to provide optimum calculation performance.
- The data cache.
- The data file cache.

Note: If you are calculating the database for the first time, the size of the calculator cache is particularly significant for calculation performance. If possible, ensure that the calculator cache is large enough for Essbase to use the optimal calculator cache option.

For information about sizing caches, see "Sizing Caches" on page 1319. Make sure you read the entire chapter before making any changes.

Locking Blocks During Calculation

When a block is calculated, Essbase locks the block and all blocks that contain its children. Essbase calculates the block and then releases both the block and the blocks containing its children.

By default, Essbase locks up to 100 blocks concurrently when calculating a block. This number of blocks is sufficient for most database calculations. If you are calculating a formula in a sparse dimension, Essbase works most efficiently if it can lock all required children concurrently. Therefore, when calculating a formula in a sparse dimension, you may want to set a number higher than 100 if you are consolidating very large numbers of children (for example, more than 100 children). By increasing the number, you ensure that Essbase can lock all required blocks and therefore performance is not impaired.

Essbase locking behavior depends on the isolation level setting. For more information, see “Isolation Levels” on page 1132.

Note: For aggregations in a sparse dimension, block locking is not a consideration because Essbase does not need to lock all blocks containing the children concurrently.

How to Change Locking for Performance

You can use the SET LOCKBLOCK command in a calculation script along with the CALCLOCKBLOCK setting in the `essbase.cfg` file to specify the maximum number of blocks that Essbase can lock concurrently when calculating a block. Do this if the default 100 blocks is not sufficient during calculation, or your calculation may be slower.

For more information, see the *Technical Reference* in the `docs` directory.

Multiple Users and Locking

Essbase uses the block locking system to manage concurrent access to users. This system ensures that only one user at a time can update or calculate a particular data block. How Essbase handles locking blocks and committing data depends on the isolation level setting.

When Essbase calculates a data block, it creates an exclusive lock. Thus, no other user can update or calculate the data block. However, other users can have read-only access to the block. When Essbase finishes the calculation, it releases the block. Other users can then update the block if they have the appropriate security access.

When a user is updating a data block, the block is locked. If a database calculation requires a data block that is being updated by another user, the calculation waits for one of the following:

- For the data block to be released if the Isolation Level setting is Uncommitted Access.
- For the calculation to complete if the Isolation Level setting is Committed Access.

Note: You can view the isolation level settings in the Application Manager by selecting Database > Settings from the main menu when the server containing the application is active.

Essbase does not provide a message to say that the calculation is waiting for the data block to be released.

You can prevent calculation delays caused by waiting for locked blocks by using Essbase security options to do either of the following:

- Deny access to other users
- Disconnect users from Essbase

For more information on these security options, see Part IV, “Designing and Building a Security System.” For information on how Essbase handles locks and transactions, see Chapter 40, “Ensuring Data Integrity.”

Note: When Essbase locks a block for calculation, it does not put an exclusive lock on the dependent child blocks. Thus, another user can update values in the child blocks. If necessary, you can use the above security options to prevent such updates.

Using Two-Pass Calculation

You can improve performance significantly by tagging an accounts dimension member as two-pass in the database outline, if it is appropriate for your application. Your combination of data and calculation needs may require the use a calculation script to calculate a formula twice, instead of two-pass tagging to preserve accuracy.

Use these sections to understand more about two-pass calculation, and decide whether you can tag an accounts dimension member as two-pass to improve performance or whether you must use a calculation script to calculate a formula twice. This section also provides information about how to enable two-pass calculation or create a calculation script for two-pass calculation:

- "Understanding Two-Pass Calculation" on page 1406
- "A Two-Pass Calculation Example" on page 1407
- "Understanding Two-Pass Calculation and Intelligent Calculation" on page 1408
- "Choosing Two-Pass Calculation Tag or Calculation Script" on page 1410
- "Enabling Two-Pass on a Default Calculation" on page 1411
- "Creating a Calculation Scripts for Two-Pass and Intelligent Calculation" on page 1412

For information about the interaction of two-pass calculation and attribute members, see Table 14 on page 232.

Understanding Two-Pass Calculation

You can use a two-pass calculation on member formulas that need to be calculated twice to produce the correct value.

Whenever possible, Essbase calculates two-pass formulas at the data block level, calculating the two-pass formulas at the same time as the main calculation. Thus, Essbase does not need to do an extra calculation pass through the database. However, in some situations, Essbase needs an extra calculation pass through the database.

How Essbase calculates the two-pass formulas depends on whether there is a dimension tagged as time as well as a dimension tagged as accounts. It also depends on the dense-sparse configuration of the time and account dimensions.

A Two-Pass Calculation Example

For example, consider this calculation required for Profit%:

```
Profit % = Profit % Sales
```

Assume that the following table shows a subset of a data block with Measures and Year as dense dimensions. Measures is tagged as accounts, and Year is tagged as time. The AGGMISSG setting is turned off (the default).

Data values have been loaded into the input cells. Essbase calculates the shaded cells. The numbers in bold show the calculation order for the cells. Cells with multiple consolidation paths are darkly shaded.

Measures -> Year	Jan	Feb	Mar	Qtr1
Profit	75	50	120	**5**
Sales	150	200	240	**6**
Profit%	**1**	**2**	**3**	**0%** **4125% 7**

Note: For detailed information on how cell calculation order depends on database configuration, see Chapter 27, “Defining the Calculation Order.”

Essbase uses this calculation order:

1. Essbase calculates the formula `Profit % Sales` for Profit % -> Jan, Profit % -> Feb, Profit % -> Mar, and Profit % -> Qtr1 (1, 2, 3, 4 above).
2. Essbase calculates Profit -> Qtr1 and Sales -> Qtr1 by adding the values for Jan, Feb, and Mar (5, 6 above).
3. Essbase calculates Profit % -> Qtr1 by adding the values for Profit % -> Jan, Profit % -> Feb, and Profit % -> Mar (7 above). This addition of percentages produces the value %125, not the correct result.

4. If you tag Profit % as two-pass in the database outline, Essbase uses the `Profit % Sales` formula to recalculate the Profit % values and produce the correct results.

Measures/Year	Jan	Feb	Mar	Qtr1
Profit	75	50	120	**245 (5)**
Sales	150	200	240	**590 (6)**
Profit%	**50% (1)**	**25% (2)**	**50% (3)**	**0% (4)** **125% (7)** **42% (8)**

For more information about calculation passes, see "Calculation Passes" on page 764.

Understanding Two-Pass Calculation and Intelligent Calculation

Two scenarios are described in detail in the following sections. If you are using Intelligent Calculation, use the scenario that matches the configuration of your database; each scenario tells you how to ensure that Essbase calculates two-pass formulas accurately.

These scenarios require that you understand the concepts of Intelligent Calculation. For more information, see Chapter 52, "Optimizing with Intelligent Calculation."

Scenario A

Scenario A demonstrates two key approaches:

- "No Extra Calculation Pass For Two-pass Formulas" on page 1409
- "All Data Blocks Marked As Clean" on page 1409

In this scenario, you place formulas in the outline and then, as appropriate, tag specific formulas as two-pass for best performance.

No Extra Calculation Pass For Two-pass Formulas

Essbase calculates the two-pass formulas while it is calculating the data block. Thus, Essbase does not need to do an extra calculation pass through the database.

All Data Blocks Marked As Clean

After the calculation, all data blocks are marked as clean for the purposes of Intelligent Calculation. For more information, see Chapter 52, “Optimizing with Intelligent Calculation.”

When you tag a member formula as two-pass in the outline, Essbase does the two-pass calculation while each data block is being calculated. However, when you repeat a formula in a calculation script, Essbase has to read the data blocks and write them to memory in order to recalculate the formula.

Scenario B

Scenario B illustrates two key approaches:

- “Extra Calculation Pass For Two-pass Formulas” on page 1409
- “Data Blocks For Two-pass Formulas Not Marked As Clean” on page 1409

In this scenario, you create a calculation script to perform the formula calculation for best performance.

Extra Calculation Pass For Two-pass Formulas

Essbase calculates the database and then does an extra calculation pass to calculate the two-pass formulas. Even though all data blocks are marked as clean after the first database calculation, Essbase ignores the clean status on the blocks that are relevant to the two-pass formula and recalculates these blocks.

Data Blocks For Two-pass Formulas Not Marked As Clean

After the first calculation, Essbase has marked all the data blocks as clean for the purposes of Intelligent Calculation. In a second calculation pass through the database, Essbase recalculates the required data blocks for the two-pass formulas. However, because the second calculation is a partial calculation of the database, Essbase does not mark the recalculated blocks as clean. When you recalculate the database with Intelligent Calculation turned on, these data blocks may be recalculated unnecessarily.

If the database configuration allows Essbase to use Scenario B, consider using a calculation script to perform two-pass formula calculations. If you use a calculation script, Essbase still does an extra calculation pass through the database; however, you can ensure that Essbase has marked all the data blocks as clean after the calculation. For more information, see “Creating a Calculation Scripts for Two-Pass and Intelligent Calculation” on page 1412.

Choosing Two-Pass Calculation Tag or Calculation Script

Even though tagging an accounts member as two-pass may bring performance benefits, some applications cannot use this method. Check these qualifications to see whether you should apply a two-pass tag or create a calculation script that performs a calculation twice for best performance and accuracy:

- You can tag a member as two-pass if it is in a dimension tagged as accounts. When you perform a default calculation on the database, Essbase automatically recalculates any formulas tagged as two-pass if they are in the dimension tagged as accounts in the database outline.
- You can tag a member as two-pass if it is a Dynamic Calc or Dynamic Calc And Store member of any dimension. For more information about dynamic calculation, see Chapter 28, “Dynamically Calculating Data Values.”
- You may need to use a calculation script to calculate a two-pass formula to obtain accurate results, even if the two-pass tag would provide performance benefits. For more information, see “Creating a Calculation Scripts for Two-Pass and Intelligent Calculation” on page 1412.
- Use a calculation script instead of the two-pass tag to ensure efficient use of Intelligent Calculation. For more information, see “Understanding Two-Pass Calculation and Intelligent Calculation” on page 1408.
- You need to use a calculation script to calculate a formula twice if the database configuration means that Essbase uses Scenario A, as described in “Scenario A” on page 1408, and if the formula references values from another data block.
- You may want to use a calculation script to calculate two-pass formulas if the database configuration means that Essbase uses Scenario B, as described in “Scenario B” on page 1409.

Enabling Two-Pass on a Default Calculation

When you perform a default calculation on a database with two-pass calculation enabled (the default), Essbase automatically attempts to calculate any formulas tagged as two-pass in the dimension tagged as accounts in the database outline. This is true even if you have customized the default calculation script.

➤ You can perform a default calculation using any of these methods:

Tool	Instructions	For More Information
Application Manager	Select Database > Calculate, and choose Default	Application Manager Help topic: Main Menu > Database Settings > Calculate
MaxL	**execute calculation**	*Technical Reference* in the `docs` directory
ESSCMD	CALCDEFAULT	*Technical Reference* in the `docs` directory

➤ To enable two-pass calculation, use any of these methods:

Tool	Instructions	For More Information
Application Manager	Select Database > Settings, General tab, and select Two-Pass Calculation	*Essbase Application Manager Online Help* topic: Main Menu > Database Settings > General
Administration Services	Database Properties window > General tab.	*Essbase Administration Services Online Help*
MaxL	**alter database**	*Technical Reference* in the `docs` directory
ESSCMD	SETDBSTATE	*Technical Reference* in the `docs` directory

Creating a Calculation Scripts for Two-Pass and Intelligent Calculation

Use these methods to create calculation scripts to perform two-pass calculations with Intelligent Calculation, so that your calculation is accurate and as fast as possible:

- Before the calculation script command that recalculate a two-pass formula, add the SET UPDATECALC OFF command to disable Intelligent Calculation.If you have Intelligent Calculation enabled (the default), Essbase calculates only the data blocks that are not marked as clean, but when you perform a default calculation of the database with Intelligent Calculation enabled, all data blocks are marked as clean, so Essbase does not perform the two-pass formula recalculation.
- When you use a calculation script, Essbase does not automatically recalculate two-pass formulas. Use the CALC TWOPASS command.
- If you have changed the default calculation from the default CALC ALL and Intelligent Calculation is enabled, the data blocks may not be marked as clean after the first calculation. For more information, see Chapter 52, "Optimizing with Intelligent Calculation." You can check the default calculation setting by selecting Database > Set Default Calc in Essbase Application Manager.

To obtain the performance benefits of Intelligent Calculation when performing the first, full calculation of the database, use one of these methods, depending on your calculation needs and outline structure:

- "Intelligent Calculation with a Large Index" on page 1413
- "Intelligent Calculation with a Small Index" on page 1415
- "Turning Off Intelligent Calculation for a Two-Pass Formula" on page 1416

These three options all use the following example situation:

> The outline has a dimension tagged as accounts, and it is a dense dimension. You want to calculate sales for each product as a percentage of sales for all products. Assume this formula should calculate the dimension:
>
> ```
> Sales % Sales -> Product
> ```
>
> When Essbase calculates the data block for each product, it has not yet calculated the value Sales->Product, so the results for the sales of each product as a percentage of total sales are incorrect.

Intelligent Calculation with a Large Index

If the index is quite large and you want the benefit of using Intelligent Calculation, you can use any of the following options for the best performance:

- "Use a Calculation Script" on page 1413
- "Use a Calculation Script and the Two-Pass Tag" on page 1414
- "Use a Client and a Calculation Script" on page 1414

All three of these options perform the same tasks:

1. Enable Intelligent Calculation.
2. Calculate the full database and marks the data blocks as clean.
3. Disable Intelligent Calculation.
4. Mark the recalculated blocks as clean, even though this calculation is a partial calculation of the database. If you do not use the command SET CLEARUPDATESTATUS AFTER, Essbase marks data blocks as clean only after a full calculation of the database.
5. Essbase cycles through the database calculating only the formula for the relevant member (Share of Sales in our example), or calculating all formulas tagged as two-pass in the database outline.

Use a Calculation Script

Use this model to create a calculation script that performs a full calculation of the database with Intelligent Calculation enabled:

```
SET UPDATECALC ON;
CALC ALL;
SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
"Share of Sales" = Sales % Sales -> Product;
```

Use a Calculation Script and the Two-Pass Tag

Use this procedure to tag a member as two-pass, and use a calculation script to calculate first the full database, then the two-pass member:

1. Place a formula in the database outline and tag it as two-pass.
2. Place the formula on the appropriate member in the dimension tagged as accounts, in our example, Share of Sales.
3. Create a calculation script that performs a full database calculation and then a two-pass calculation:

```
SET UPDATECALC ON;
CALC ALL;
SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
CALC TWOPASS;
```

Use a Client and a Calculation Script

Use this procedure to perform a default calculation from a client and then use a calculation script to perform the formula calculation:

1. Enable Intelligent Calculation if this default has been changed.
2. Perform a full calculation, using any of the clients listed in Table 90 on page 1415.
3. Use a calculation script similar to this example to disable Intelligent Calculation and calculate the formula:

```
SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
"Share of Sales" = Sales % Sales -> Product;
```

or:

```
SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
CALC TWOPASS;
```

Table 90: Methods for Performing a Full Calculation

Tool	Instructions	For More Information
Administration Services	Calculate Database dialog box	*Essbase Administration Services Online Help*
MaxL	**execute calculation**	*Technical Reference* in the `docs` directory
ESSCMD	CALCDEFAULT	*Technical Reference* in the `docs` directory
Application Manager	Select Database > Calculate, and choose Default, or your customized default calculation script.	Application Manager help topic: Main Menu > Database > Calculate

For more information on Intelligent Calculation, see Chapter 52, “Optimizing with Intelligent Calculation.”

For more information on developing formulas and calculation scripts, see Chapter 25, “Developing Formulas,” and Chapter 30, “Developing Calculation Scripts.”

Intelligent Calculation with a Small Index

If the index is small and you want the benefit of using Intelligent Calculation, use this procedure:

1. Create a calculation script to calculate the database, but tell Essbase not to mark the calculated data blocks as clean
2. Mark all data blocks as clean, and do not recalculate the data blocks.

```
SET CLEARUPDATESTATUS OFF;
CALC ALL;
CALC TWOPASS;
SET CLEARUPDATESTATUS ONLY;
CALC ALL;
```

With the example script, Essbase performs these tasks:

1. The SET CLEARUPDATESTATUS OFF command tells Essbase not to mark the calculated data blocks as clean.
2. The first CALC ALL command causes Essbase to cycle through the database calculating all dirty data blocks. Essbase does not mark the calculated data blocks as clean. Essbase does not automatically recalculate the formulas tagged as two-pass in the database outline.
3. The CALC TWOPASS command causes Essbase to cycle through the database recalculating the formulas that are tagged as two-pass in the dimension tagged as accounts in the database outline. Essbase recalculates the formulas because the required data blocks are not marked as clean by the previous CALC ALL. Essbase does not mark the recalculated data blocks as clean.
4. The SET CLEARUPDATESTATUS ONLY command tells Essbase to mark the data blocks as clean but not to calculate the data blocks. This command disables calculation.
5. The last CALC ALL command causes Essbase to cycle through the database and mark all the data blocks as clean. Essbase searches through the index and marks the data blocks as clean. It does not calculate the data blocks.

Turning Off Intelligent Calculation for a Two-Pass Formula

Create a calculation script that performs these tasks:

1. Disables Intelligent Calculation.
2. Performs a full calculation.
3. Repeats the two-pass formula:

```
SET UPDATECALC OFF;
CALC ALL;
"Share of Sales" = Sales % Sales -> Product;
```

Choosing Between Member Set Functions and Performance

Queries and calculations which reference a member that has been tagged as Dynamic Calc or Dynamic Calc and Store may be significantly slower than queries and calculations involving the same members, if the member has formulas involving any of these functions:

- @CURRMBR
- @PARENT
- @SPARENTVAL
- @ANCEST
- @SANCESTVAL

If you are experiencing slow performance, you may wish to either remove the dynamic calculation tag or remove these functions from the attached formula.

Aggregating #MISSING Values

If no data value exists for a combination of dimension members, Essbase gives the combination a value of #MISSING. Essbase treats #MISSING values and zero (0) values differently.

Understanding #MISSING calculation

This table shows how Essbase calculates #MISSING values. In this table, X represents any number:

Table 91: How Essbase Treats #MISSING Values

Calculation/Operation	**Result**
X + #MISSING	X
X – #MISSING #MISSING – X	X -X
X * #MISSING	#MISSING

Table 91: How Essbase Treats #MISSING Values (Continued)

Calculation/Operation	**Result**
X / #MISSING #MISSING / X X / 0	#MISSING #MISSING #MISSING
X % #MISSING #MISSING % X X % 0	#MISSING #MISSING #MISSING
X == #MISSING	FALSE, unless X is #MISSING
X != #MISSING X <> #MISSING	TRUE, unless X is #MISSING TRUE, unless X is #MISSING
(X <= #MISSING)	(X <= 0)
(X >= #MISSING)	(X >= 0) or (X == #MISSING)
(X > #MISSING)	(X > 0)
(X < #MISSING)	(X < 0)
X AND #MISSING: Y AND #MISSING, where Y represents any nonzero value	#MISSING
0 AND #MISSING #MISSING AND #MISSING	0 #MISSING
X OR #MISSING: Y OR #MISSING, where Y represents any nonzero value 0 OR #MISSING #MISSING OR #MISSING	 1 #MISSING #MISSING
IF (#MISSING)	IF (0)
f (#MISSING)	#MISSING for any Essbase function of one variable
f (X)	#MISSING for any X not in the domain of *f* and any Essbase function of more than one variable (except where specifically noted)

By default, Essbase does not aggregate #MISSING values.However, if you always load data at level 0 and never at parent levels, then you should enable the setting for aggregating #MISSING values. Use of this setting provides a calculation performance improvement of between 1% and 30%. The performance improvement varies, depending on database size and configuration.

CAUTION: The default, not aggregating #MISSING values must be in effect if you load data at parent, rather than child, levels.if any child member combinations have #MISSING values. If all child member combinations have any other values, including zero (0), then Essbase aggregates the child values and overwrites the parent values correctly, so you can safely change the default.

Changing Aggregation for Performance

To aggregate, enable the setting for aggregating #MISSING values by using one of the methods described above. The degree of performance improvement you achieve depends on the ratio between upper level blocks and input blocks in the database. For more information, see Chapter 27, “Defining the Calculation Order.”

➤ You can change the way Essbase aggregates #MISSING values using any of these methods:

Tool	**Instructions**	**For More Information**
Administration Services	Database Properties window > General tab	*Essbase Administration Services Online Help*
Calculation Script	Use SET AGGMISSG	*Technical Reference* in the `docs` directory
MaxL	**alter database**	*Technical Reference* in the `docs` directory
ESSCMD	SETDBSTATEITEM	*Technical Reference* in the `docs` directory
Application Manager	Database > Settings, then check Aggregate Missing Values	*Essbase Application Manager Online Help* topic:

Note: If you enable the setting for aggregating `#MISSING` values, the cell calculation order within a data block changes. For more information, see Chapter 27, “Defining the Calculation Order.”

When the setting for aggregating `#MISSING` values is disabled, note that the performance overhead is particularly high in the following two situations:

- When you have a low ratio of calculated data blocks to input data blocks
- When you load many data values at parent levels on sparse dimensions; for example, in the Sample Basic database, if you load many data values into East in a sparse Market dimension

In these situations, the performance overhead is between 10% and 30%. If calculation performance is critical, you may want to reconsider the database configuration or reconsider how you load data.

For more information on setting the behavior for aggregating `#MISSING` values, see “Aggregating #MISSING Values” on page 1417, Chapter 27, “Defining the Calculation Order,” and the *Technical Reference* in the `docs` directory.

Removing #MISSSING Blocks

CLEARDATA changes the value of cells in a block to #MISSING. It does not remove the data blocks. These extra blocks can slow performance.

If the `#MISSING` blocks are slowing performance, perform either of these tasks:

- Use the CLEARBLOCK command to remove the data blocks. See the *Technical Reference* in the `docs` directory.
- Export the data and import it again. See “Exporting Data” on page 1253 and “Reloading Exported Data” on page 1254.

Identifying Other Calculation Optimization Issues

This map identifies the location in other sections that discuss the relationship between calculation and performance:

- "Benefitting from Dynamic Calculations" on page 772
- "Reducing the Impact on Retrieval Time" on page 784
- "Choosing Between Dynamic Calc and Dynamic Calc And Store" on page 778
- "Writing Calculation Scripts for Partitions" on page 872
- "Dynamically Calculating Data in Partitions" on page 797
- "Specifying Global Settings for a Database Calculation" on page 826. Some of the settings affect performance
- "Writing Calculation Scripts for Partitions" on page 872
- For the relationship of two-pass calculation and the SET CLEARUPDATESTATUS command, see the *Technical Reference* in the `docs` directory
- When you convert currencies using the CCONV command, the resulting data blocks are marked as *dirty* for the purposes of Intelligent Calculation. This means that Essbase recalculates all the converted blocks when you recalculate your database. For more information on Intelligent Calculation, see Chapter 52, "Optimizing with Intelligent Calculation."

Chapter 52

Optimizing with Intelligent Calculation

This chapter provides information on how to use Intelligent Calculation to optimize the performance of Essbase calculations. This chapter includes the following sections:

For additional information on optimizing overall database calculations, see the following:

- Chapter 13, "Designing Partitioned Applications"
- Chapter 28, "Dynamically Calculating Data Values"
- Chapter 51, "Optimizing Calculations"

Introducing Intelligent Calculation

When you do a full calculation of a database, Essbase tracks which data blocks it has calculated. If you then load a subset of data, on subsequent calculations, you can choose to calculate only those data blocks that Essbase has not yet calculated but need calculation, and those calculated blocks that require recalculation because of the new data. In Essbase, this process is called Intelligent Calculation.

By default, Intelligent Calculation is turned on. You can change this default setting in the configuration file `essbase.cfg`. You can also turn Intelligent Calculation on or off in a calc script. For more information, see "Turning Intelligent Calculation On and Off" on page 1427.

You can check the current Intelligent Calculation setting by selecting Database > Set Default Calc in Application Manager.

Benefits of Intelligent Calculation

Intelligent Calculation is designed to provide significant calculation performance benefits for these types of calculations:

- For a full calculation of a database (CALC ALL)
- For a calc script that calculates all members in one CALC DIM command.
- For database calculations that can't use Intelligent Calculation for the full calculation, you may be able to use Intelligent Calculation for part of the calculation.

  For example, consider a case in which you calculate a database by doing a default consolidation and then an allocation of data. To significantly improve your calculation performance in this case, enable Intelligent Calculation for the default consolidation and then disable Intelligent Calculation for the allocation.

  Assuming that Intelligent Calculation is turned on (the default), create a calculation script to perform these steps for a partial Intelligent Calculation:

  a. Enable Intelligent Calculation if the default has been changed

  b. Use CALC ALL to calculate the database

  c. Use the SET UPDATECALC command to disable Intelligent Calculation for the next step.

  d. Allocate data. Optionally, enable Intelligent Calculation again.

Intelligent Calculation and Data Block Status

To provide Intelligent Calculation, Essbase checks the status of the data blocks in a database. Data blocks have a calculation status of either clean or dirty. Essbase marks a data block as clean after certain calculations.

When Intelligent Calculation is enabled, Essbase calculates only dirty blocks and their dependent parents. Disabling Intelligent Calculation means that Essbase calculates all data blocks, regardless of whether they are marked as clean or dirty.

Use these topics to understand clean and dirty status, and to learn how to manage clean and dirty status for Intelligent Calculation:

- "Marking Blocks As Clean" on page 1425
- "Marking Blocks as Dirty" on page 1426
- "Maintaining Clean and Dirty Status" on page 1426

Marking Blocks As Clean

Essbase marks data blocks as clean in these types of calculations:

- A full calculation (CALC ALL) of a database, the default calculation for a database.
- A calc script that calculates all the dimensions in one CALC DIM statement. For example, the following calc script calculates all members in the Sample Basic database:

  ```
  CALC DIM(Measures, Product, Market, Year, Scenario);
  ```

  Compare this to a calc script that calculates all the members with two CALC DIM statements:

  ```
  CALC DIM(Measures, Product);
  CALC DIM(Market, Year, Scenario);
  ```

  Using two CALC DIM statements causes Essbase to do at least two calculation passes through the database. In this calculation, Essbase does not, by default, mark the data blocks as clean. Because Intelligent Calculation depends on accurate clean and dirty status, you must manage these markers carefully. For more information, see "Maintaining Clean and Dirty Status" on page 1426.

Essbase marks calculated data blocks as clean only in situations described above, unless you use the `SET CLEARUPDATESTATUS` command in a calc script. For more information, see "Using the SET CLEARUPDATESTATUS Command" on page 1429.

Marking Blocks as Dirty

Essbase marks a data block as dirty in the following situations:

- Calculating the data block for a partial calculation of the database only if SET CLEARUPDATESTATUS AFTER is not part of the partial calculation command in the calc script
- Loading data into the data block
- Restructuring the database (for example, by adding a member to a dense dimension)
- Copying data to the data block, for example using DATACOPY

Maintaining Clean and Dirty Status

If you want to use Intelligent Calculation when calculating a subset of a database or when performing multiple calculation passes through a database, consider carefully the implications of how Essbase marks data blocks as clean. When using Intelligent Calculation, you must accurately maintain the clean and dirty status of the data blocks to ensure that Essbase recalculates the database as efficiently as possible.

For example, when you calculate a subset of a database, the newly calculated data blocks are not marked as clean by default. You can ensure that the newly calculated blocks are marked as clean by using the SET CLEARUPDATESTATUS AFTER command in a calc script. To ensure accurate calculation results, review the information in "Using the SET CLEARUPDATESTATUS Command" on page 1429 and the *Technical Reference* in the `docs` directory (Calculation Commands List: CLEARUPDATESTATUS) before creating the calc script.

Limitations of Intelligent Calculation

Consider the following limitations when using Intelligent Calculation:

- Intelligent Calculation works on a data block level and not on a cell level. For example, if you load a data value into one cell of a data block, the whole data block is marked as dirty.
- Changing a formula on the database outline or changing an accounts property on the database outline does not cause Essbase to restructure the database. Therefore, Essbase does not mark the affected blocks as dirty. You must recalculate the appropriate data blocks. For more information, see "Changing a Formula or Accounts Property" on page 1442.
- Whenever possible, Essbase calculates formulas that are tagged as two pass and in the dimension tagged as accounts as part of the main calculation of a database. However, you may need to use a calc script to calculate some formulas twice. When you use a calc script, disable Intelligent Calculation before recalculating formulas.

Using Intelligent Calculation

This section provides information on turning Intelligent Calculation on and off and on using Intelligent Calculation with different types of calculations:

- "Turning Intelligent Calculation On and Off" on page 1427
- "Using Intelligent Calculation for a Default, Full Calculation" on page 1428
- "Using Intelligent Calculation for a Calculation Script, Partial Calculation" on page 1429

Turning Intelligent Calculation On and Off

By default, Intelligent Calculation is turned on. You can change the default by using the UPDATECALC setting in the `ESSBASE.CFG` file.

You can turn Intelligent Calculation on and off for the duration of a calc script by using the SET UPDATECALC command in a calc script. Enabling Intelligent Calculation means that Essbase calculates only dirty blocks and their dependent parents. Disabling Intelligent Calculation means that Essbase calculates all data blocks, regardless of whether they are marked as clean or dirty.

For more information on these commands and on `ESSBASE.CFG`, see the *Technical Reference* in the `docs` directory.

Using Intelligent Calculation for a Default, Full Calculation

Intelligent Calculation provides significant performance benefits when you do a full calculation (CALC ALL) of a database. If you do a full calculation of a database, leave Intelligent Calculation turned on (the default) to take advantage of the performance benefits that it provides.

Unless you have changed the default, a full calculation (CALC ALL) is the default calculation for a database. You can check the default calculation setting by selecting Database > Set Default Calc in Essbase Application Manager.

CAUTION: When using Intelligent Calculation, note the information in "Limitations of Intelligent Calculation" on page 1427.

Calculating for the First Time

When you do a full calculation of a database for the first time, Essbase calculates every existing block. The performance is the same whether you have Intelligent Calculation turned on or off.

Recalculating

When you do a full recalculation of a database with Intelligent Calculation turned on, Essbase checks each block to see if it is marked as clean or dirty. For more information on clean and dirty, see "Intelligent Calculation and Data Block Status" on page 1425.

Checking the data blocks has a 5% to 10% performance overhead. During most recalculations, this small performance overhead is insignificant when compared to the performance gained by enabling Intelligent Calculation.

However, if you recalculate a database in which more than approximately 80% of the values have changed, the overhead of Intelligent Calculation may outweigh the benefits. In this case, disable Intelligent Calculation.

Using Intelligent Calculation for a Calculation Script, Partial Calculation

Essbase marks a data block as clean when it calculates the data block on a full calculation (CALC ALL) or when it calculates all dimensions in one CALC DIM command. For more information, see "Intelligent Calculation and Data Block Status" on page 1425.

In any other calculations, Essbase does not mark calculated data blocks as clean, unless you use the SET CLEARUPDATESTATUS command in a calc script. For example, if you calculate a subset of a database or calculate a database in two calculation passes, Essbase does not mark the calculated blocks as clean, unless you use the SET CLEARUPDATESTATUS command.

The following calc scripts do not cause Essbase to mark the calculated data blocks as clean:

```
FIX("New York")
CALC DIM(Product, Measures);
ENDFIX

CALC DIM(Measures, Product);
CALC DIM(Market, Year, Scenario);
```

Be sure to use SET CLEARUPDATESTATUS to avoid unnecessary recalculations.

Using the SET CLEARUPDATESTATUS Command

In some cases, Essbase does not mark calculated blocks as clean; for example, if you calculate a subset of a database or calculate a database in two calculation passes. To manually mark data blocks as clean for purposes of Intelligent Calculation, use the SET CLEARUPDATESTATUS command in a calc script.

Use these sections to understand the command SET CLEARUPDATESTATUS, choose a setting, and for instructions about how to use the command:

- "Understanding SET CLEARUPDATESTATUS" on page 1430
- "Choosing a SET CLEARUPDATESTATUS Setting" on page 1430
- "Examples Using SET CLEARUPDATESTATUS" on page 1431

For more information about the relationship of Intelligent Calculation and data block status, see "Intelligent Calculation and Data Block Status" on page 1425.

Understanding SET CLEARUPDATESTATUS

The SET CLEARUPDATESTATUS command has three parameters:
AFTER > ONLY > OFF.

- SET CLEARUPDATESTATUS AFTER;

  Essbase marks calculated data blocks as clean, even if it is calculating a subset of a database.

- SET CLEARUPDATESTATUS ONLY;

  Essbase marks the specified data blocks as clean but does not calculate the data blocks. This parameter provides the same result as AFTER, but without calculation.

- SET CLEARUPDATESTATUS OFF;

  Essbase calculates the data blocks but does not mark the calculated data blocks as clean. Data blocks are not marked as clean, even on a full calculation (CALC ALL) of a database. The existing clean or dirty status of the calculated data blocks remains unchanged.

Choosing a SET CLEARUPDATESTATUS Setting

When you use the SET CLEARUPDATESTATUS command to mark calculated data blocks as clean, be aware of these recommendations before selecting the setting (AFTER, ONLY, OFF):

- Only calculated data blocks are marked as clean. For more information, see "Calculating Data Blocks" on page 1433.
- Do not use the SET CLEARUPDATESTATUS AFTER command with concurrent calculations unless you are certain that the concurrent calculations do not need to calculate the same data block or blocks. If concurrent calculations attempt to calculate the same data blocks, with Intelligent Calculation enabled, Essbase does not recalculate the data blocks because the blocks if the data blocks are already marked as clean by the other concurrent calculation. For more information, see "Handling Concurrent Calculations" on page 1435.

- When Essbase calculates data blocks on a first calculation pass through a database, it marks the data blocks as clean. If you try to calculate the same data blocks on a subsequent pass with Intelligent Calculation is enabled, Essbase does not recalculate the data blocks because they are already marked as clean.

Examples Using SET CLEARUPDATESTATUS

Assume a scenario using the Sample Basic database:

- The sparse dimensions are Market and Product.
- New York is a member on the sparse Market dimension.
- Intelligent Calculation is turned on (the default).

These three examples show different ways of using SET CLEARUPDATESTATUS:

- "Example 1: CLEARUPDATESTATUS AFTER" on page 1431
- "Example 2: CLEARUPDATESTATUS ONLY" on page 1432
- "Example 3: CLEARUPDATESTATUS OFF" on page 1432

Example 1: CLEARUPDATESTATUS AFTER

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Product);
ENDFIX
```

In this example, Essbase searches for dirty parent data blocks for New York (for example New York -> Colas, in which Colas is a parent member). It calculates these dirty blocks and marks them as clean. (The calculation is based on the Product dimension.) Essbase does not mark the level 0 data blocks as clean because they are not calculated. For information on level 0 blocks, see Chapter 27, "Defining the Calculation Order."

Example 2: CLEARUPDATESTATUS ONLY

```
SET CLEARUPDATESTATUS ONLY;
FIX("New York")
CALC DIM(Product);
ENDFIX
```

Essbase searches for dirty parent data blocks for New York (for example New York -> Colas, in which Colas is a parent member on the Product dimension). Essbase marks the dirty parent data blocks as clean, but does not calculate the data blocks. Essbase does not mark the level 0 data blocks as clean because they are not calculated. For example, if New York -> 100-10 (a level 0 block) is dirty, it remains dirty.

Example 3: CLEARUPDATESTATUS OFF

```
SET CLEARUPDATESTATUS OFF;
CALC ALL;
CALC TWOPASS;
SET CLEARUPDATESTATUS ONLY;
CALC ALL;
```

In this example, Essbase first calculates all the dirty data blocks in the database. The calculated data blocks remain dirty. Essbase does *not* mark them as clean.

Essbase then calculates the members tagged as two pass that are in the dimension tagged as accounts. Because the data blocks are still marked as dirty, Essbase recalculates them. Again, it does not mark the calculated data blocks as clean.

Essbase then searches for all the dirty blocks in the database and marks them as clean. It does *not* calculate the blocks, even though a CALC ALL command is used.

Calculating Data Blocks

Essbase creates a data block for each unique combination of sparse dimension members, provided that at least one data value exists for the combination. Each data block represents all dense dimension member values for that unique combination of sparse dimension members.

For example, in the Sample Basic database, the Market and Product dimensions are sparse. Therefore, the data block New York -> Colas represents all the member values on the Year, Measures, and Scenario dimensions for the sparse combination New York -> Colas.

These sections provide information about conditions that affect performance with Intelligent Calculation:

- “Calculating a Dense Dimension” on page 1433
- “Calculating a Sparse Dimension” on page 1434
- “Handling Concurrent Calculations” on page 1435
- “Understanding Multiple-Pass Calculations” on page 1436

These sections assumes that you are familiar with the concepts of upper level, level 0, and input data blocks. For information about levels and data blocks, see “Member Relationships, Generations, and Levels” on page 63.

For more information on how Essbase creates data blocks, see Chapter 27, “Defining the Calculation Order.”

Calculating a Dense Dimension

When you calculate a dense dimension and do not use a FIX command, Essbase calculates at least some of the data values in every data block in the database. For example, the following calc script is based on the Sample Basic database:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Year);
```

This script calculates the Year dimension, which is a dense dimension. Because Year is dense, every data block in the database includes members of the Year dimension. Therefore, Essbase calculates data values in every data block. Because the script uses the SET CLEARUPDATESTATUS AFTER command, Essbase marks all the data blocks as clean.

Calculating a Sparse Dimension

When you calculate a sparse dimension, Essbase may not need to calculate every data block in the database. For example, the following calc script is based on Sample Basic:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product);
```

This script calculates the Product dimension, which is a sparse dimension. Because Product is sparse, a data block exists for each member on the Product dimension. For example, one data block exists for New York -> Colas and another for New York -> 100-10.

Level 0 Effects

The data block New York -> 100-10 is a level 0 block, it does not represent a parent member on either sparse dimension (Market or Product). The data values for New York -> 100-10 are input values; they are loaded into the database. Therefore, Essbase does not need to calculate this data block. Nor does Essbase mark the data block for New York -> 100-10 as clean, even though the script uses the SET CLEARUPDATESTATUS AFTER command.

Note: Essbase does calculate level 0 data blocks if a corresponding sparse, level 0 member has a formula applied to it.

If you load data into a database, the level 0 data blocks into which you load data are marked as dirty. If you subsequently calculate only a sparse dimension or dimensions, the level 0 blocks remain dirty, because Essbase does not calculate them. Therefore, when you recalculate only a sparse dimension or dimensions, Essbase recalculates all upper-level data blocks because the upper-level blocks are marked as dirty if their child blocks are dirty, even though the upper level blocks were originally clean.

Upper Level Effects

Colas is a parent level member on the Product dimension. Essbase needs to calculate values for Colas, so Essbase calculates this data block. Because the script uses the SET CLEARUPDATESTATUS AFTER command, Essbase marks the data block as clean.

When Essbase calculates a sparse dimension, it recalculates an upper level data block if the block is dependent on one or more dirty child blocks.

Avoiding Unnecessary Calculation

You can avoid unnecessary calculation by ensuring that you calculate at least one dense dimension. When you calculate a dense dimension and do not use the FIX command, data values are calculated in every data block, including the level 0 blocks. So the level 0 blocks are marked as clean.

Handling Concurrent Calculations

If concurrent calculations attempt to calculate the same data blocks and Intelligent Calculation is turned on, Essbase may not recalculate the data blocks because they are already marked as clean.

Do not use the SET CLEARUPDATESTATUS AFTER command with concurrent calculations unless you are certain that the concurrent calculations do not calculate the same data block or blocks.

Consider the following example, which is based on the Sample Basic database. Actual and Budget are members of the dense Scenario dimension. Because Scenario is dense, each data block in the database contains both Actual and Budget values.

```
SET CLEARUPDATESTATUS AFTER;
FIX(“New York”, Actual)
CALC DIM(Product, Year);
ENDFIX
```

If User One runs the above calc script, Essbase calculates the Actual values for all data blocks that represent New York. Essbase marks the calculated data blocks as clean, even though not all the data values in each calculated block have been calculated. For example, the Budget values have not yet been calculated.

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York", Budget)
CALC DIM(Product, Year);
ENDFIX
```

If User Two runs the calc script above to calculate the Budget values for New York, Essbase does not recalculate the specified data blocks, because they are already marked as clean. The calculation results for Budget are not correct.

One way to solve this problem is to make the Scenario dimension sparse; then the Actual and Budget values are in different data blocks, for example, New York -> Colas -> Actual and New York -> Colas -> Budget. In this case, the second calc script correctly calculates Budget data block.

Understanding Multiple-Pass Calculations

Whenever possible, Essbase calculates a database in one calculation pass through the database. For information on calculation passes, see Chapter 27, "Defining the Calculation Order."

When you use a calc script to calculate a database, the number of calculation passes that Essbase performs depends upon the calc script. For more information about the relationship between calculation passes and Intelligent Calculation, see "Intelligent Calculation and Data Block Status" on page 1425. For more information about grouping formulas and calculations, see Chapter 30, "Developing Calculation Scripts."

For example, assume Essbase calculates data blocks on a first calculation pass through a database and then marks them as clean. If you then attempt to calculate the same data blocks on a subsequent pass and Intelligent Calculation enabled, Essbase does not recalculate the data blocks because they are already marked as clean.

Examples and Solutions for Multiple-Pass Calculations

These examples describe situations in which you obtain incorrect calculation results, and provide a solution you can implement to obtain correct results:

- "Example 1: Intelligent Calculation and Two Pass" on page 1437
- "Example 2: SET CLEARUPDATESTATUS and FIX" on page 1438
- "Example 3: SET CLEARUPDATESTATUS and Two CALC DIM Commands" on page 1439

The examples are based on the Sample Basic database and assume that Intelligent Calculation is turned on.

Example 1: Intelligent Calculation and Two Pass

This calc script does a default calculation and then a two-pass calculation:

```
CALC ALL;
CALC TWOPASS;
```

Error

Essbase calculates the dirty data blocks in the database and marks all the data blocks as clean. Essbase then needs to recalculate the members tagged as two pass in the dimension tagged as accounts. However, Essbase does not recalculate the specified data blocks because they are already marked as clean. The calculation results are not correct.

Solution

You can calculate the correct results by disabling Intelligent Calculation for the two pass calculation.

Example 2: SET CLEARUPDATESTATUS and FIX

This calc script calculates data values for New York. The calculation is based on the Product dimension:

```
SET CLEARUPDATESTATUS AFTER;
FIX(“New York”)
CALC DIM(Product);
ENDFIX
CALC TWOPASS;
```

Error

Essbase performs the following processes:

1. Essbase cycles through the database calculating the dirty data blocks that represent New York. The calculation is based on the Product dimension. Thus, Essbase calculates only the blocks that represent a parent member on the Product dimension (for example, New York -> Colas, New York -> Root Beer, and New York -> Fruit Soda), and then only calculates the aggregations and formulas for the Product dimension.
2. Because the SET CLEARUPDATESTATUS AFTER command is used, Essbase marks the calculated data blocks as clean, even though not all data values in each calculated block have been calculated.
3. Essbase should recalculate the members tagged as Two-Pass in the dimension tagged as accounts. However, some of these data blocks are already marked as clean from the calculation in Step 2. Essbase does not recalculate the data blocks that are already marked as clean. The calculation results are not correct.

Solution

You can calculate the correct results by disabling Intelligent Calculation for the two pass calculation. For detailed information on using two pass calculations, see Chapter 27, “Defining the Calculation Order.”

Example 3: SET CLEARUPDATESTATUS and Two CALC DIM Commands

This calc script bases the database calculation on the Product and Year dimensions. Because two CALC DIM commands are used, Essbase does two calculation passes through the database:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product);
CALC DIM(Year);
```

Error

Essbase performs the following processes:

1. Essbase cycles through the database calculating the dirty data blocks. The calculation is based on the Product dimension, as in Example 2.
2. Because the SET CLEARUPDATESTATUS AFTER command is used, Essbase marks the calculated data blocks as clean, even though not all data values in each calculated block have been calculated.
3. Essbase should recalculate the data blocks. The recalculation is based on the Year dimension. However, as a result of the calculation in Step 2, some of the data blocks are already marked as clean. Essbase does not recalculate the data blocks that are already marked as clean. The calculation results are not correct.

Solution

You can calculate the correct results by using one CALC DIM command to calculate both the Product and Year dimensions. Essbase then calculates both dimensions in one calculation pass through the database. The following calc script calculates the correct results:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product, Year);
```

Note: When you calculate several dimensions in one CALC DIM command, Essbase calculates the dimensions in the default calculation order and not in the order in which you list them in the command. For more information, see Chapter 27, “Defining the Calculation Order.”

Example 4: Two Separate Calc Scripts

This example calculates data values for New York but calculates based on two different dimensions using two separate calc scripts. The first calc script calculates the Product dimension:

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Product);
ENDFIX
```

Essbase calculates the data blocks that include New York. The calculation is based on the Product dimension. Thus, Essbase calculates only the dirty blocks that include a parent member on the Product dimension (for example, New York -> Colas, New York -> Root Beer, and New York -> Fruit Soda), and even then only calculates the aggregations and formulas for the Product dimension.

Because of the CLEARUPDATESTATUS AFTER command, Essbase marks the calculated data blocks as clean, even though not all data values in each calculated block have been calculated.

The second calc script calculates the Year dimension:

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Year);
ENDFIX
```

Essbase calculates the data blocks that represent New York. The calculation is based on the Year dimension, which is a dense dimension. Thus, Essbase should calculate all data blocks that include New York, although within each block, Essbase calculates only the aggregations and formulas for the Year dimension.

Error

As a result of the first calculation, some of the data blocks for New York are already marked as clean. Essbase does not recalculate these data blocks with the second calc script because the data blocks are already marked as clean. The calculation results are not correct.

Solution

You can calculate the correct results by telling Essbase *not* to mark the calculated data blocks as clean. The following calc script calculates the correct results:

```
SET CLEARUPDATESTATUS OFF;
FIX("New York")
CALC DIM(Product);
ENDFIX
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Year);
ENDFIX
```

The SET CLEARUPDATESTATUS OFF command. With it, Essbase calculates dirty data blocks, but does not to mark them as clean, unlike the SET CLEARUPDATESTATUS AFTER command.

This solution assumes that the data blocks are not already marked as clean from a previous partial calculation of the database.

You can ensure that all data blocks are calculated, irrespective of their clean or dirty status, by disabling Intelligent Calculation.

The following calc script calculates all specified data blocks, irrespective of their clean or dirty status:

```
SET UPDATECALC OFF;
FIX("New York")
CALC DIM(Year, Product);
ENDFIX
```

Because you have not used the SET CLEARUPDATESTATUS AFTER command, Essbase does not mark calculated data blocks as clean.

Understanding the Effects of Intelligent Calculation

Using Intelligent Calculation may have implications for the way you administer a database. This section discusses the implications of each of the following:

- "Changing a Formula or Accounts Property" on page 1442
- "Using Relationship and Financial Functions" on page 1443
- "Restructuring a Database" on page 1443
- "Copying and Clearing Data" on page 1443
- "Converting Currencies" on page 1443

Changing a Formula or Accounts Property

Neither changing a formula in the database outline nor changing an accounts property in the database outline causes Essbase to restructure the database. Thus, data blocks affected by such a change are not marked as dirty. For example, if you change a time balance tag in the dimension tagged as accounts, Essbase does not restructure the database and does not mark the affected blocks as dirty.

When you subsequently run a default calculation with Intelligent Calculation turned on, your changes are not calculated. Since you must recalculate the appropriate data blocks, you can accomplish this by using a calc script to do any of the following:

- Disable Intelligent Calculation and calculate the member formula that has changed.
- Disable Intelligent Calculation and use the FIX command to calculate the appropriate subset of a database.
- Disable Intelligent Calculation and perform a default CALC ALL on a database.

For more detailed information, see Chapter 30, "Developing Calculation Scripts."

Using Relationship and Financial Functions

If you use relationship functions (for example, @PRIOR or @NEXT) or financial functions (for example, @NPV or @INTEREST) in a formula on a sparse dimension, Essbase always recalculates the data block that contains the formula, even if the data block is marked as clean.

For more information on relationship functions and financial functions, see the *Technical Reference* in the `docs` directory.

Restructuring a Database

When you restructure a database (for example, by adding a member to a dense dimension), all data blocks potentially need recalculating. Therefore, Essbase marks all data blocks as dirty. When you calculate the restructured database, all blocks are calculated.

Note: Changing a formula in the database outline or changing an accounts property in the database outline does not cause Essbase to restructure the database. You must recalculate the appropriate data blocks. For more information, see "Changing a Formula or Accounts Property" on page 1442.

Copying and Clearing Data

When you copy values to a data block by using the DATACOPY command, the resulting data block is marked as dirty. Essbase calculates the block when you recalculate a database.

When you clear data values by using the CLEARDATA and CLEARBLOCK commands, Essbase clears all the blocks regardless of whether they are marked as clean or dirty.

For more information on these commands, see the *Technical Reference* in the `docs` directory.

Converting Currencies

When you convert currencies using the CCONV command, the resulting data blocks are marked as dirty. Essbase calculates all converted blocks when you recalculate a database.

For more information about the CCONV command, see the *Technical Reference* in the `docs` directory.

Chapter

53

Optimizing Reports and Other Types of Retrieval

The time required to generate a report varies depending upon factors such as the size of the database you are reporting from, the number of queries included in the script, and the size of the report buffer.

This chapter describes ways to optimize the time required to generate your reports, and other retrieval optimization issues:

If you are migrating from a previous version of Essbase, see the *Essbase Installation Guide* for information about system changes and enhancements.

Changing Buffer Size

Configurable variables specify the size of the buffers used for storing and sorting data extracted by a report. The buffer should be large enough to prevent unnecessary read and write activities.

These report variables are used in the conditional retrieval and data sorting commands:

- "Setting the Retrieval Buffer Size" on page 1446
- "Setting the Retrieval Sort Buffer Size" on page 1448

For more information about these settings, see the *Technical Reference* in the `docs` directory. For information about the Report Extractor process of retrieving data, see Chapter 34, "Quick Start to Report Scripts."

Setting the Retrieval Buffer Size

The Retrieval Buffer Size setting specifies the size, in kilobytes, of the server buffer that holds extracted row data cells before they are evaluated by the RESTRICT, TOP, or BOTTOM commands. If this buffer is full, the rows are processed, and the buffer is reused. Both the Spreadsheet Retrieval Wizard and the Report Writer use this buffer to process retrievals.

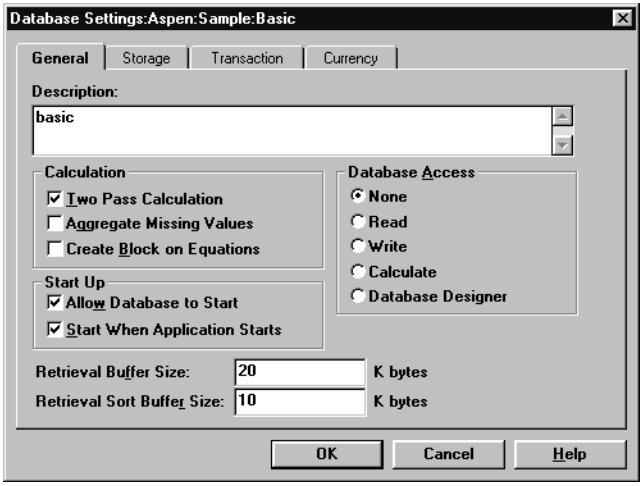
You can adjust the buffer size on a per database basis. The default buffer size is set to 10 kilobytes. If you are increasing the size of the buffer, it is recommended that you do not exceed 100 kilobytes, although the size limit is set at 100,000 kilobytes. Larger buffer sizes can create problems when concurrent users generate reports at the same time.

➤ To set the retrieval buffer size:

1. From the Application Manager menu, choose Database > Settings.
2. Select the General page.

3. Type the number, in kilobytes, of the required buffer size in the Retrieval Buffer Size box, as shown in Figure 547.

Figure 547: Setting the Retrieval Buffer Size



Description: basic

Calculation
- ☑ Two Pass Calculation
- □ Aggregate Missing Values
- □ Create Block on Equations

Start Up
- ☑ Allow Database to Start
- ☑ Start When Application Starts

Database Access
- ● None
- ○ Read
- ○ Write
- ○ Calculate
- ○ Database Designer

Retrieval Buffer Size: 20 K bytes

Retrieval Sort Buffer Size: 10 K bytes

To determine the best buffer size for your report processing needs, test a report script with different settings.

Use any of these alternate methods to set the retrieval buffer size:

Tool	Instruction	Additional Information
Administration Services	Database Properties window > General tab	*Essbase Administration Services Online Help*
MaxL	**alter database**	*Technical Reference* in the `docs` directory
ESSCMD	SETDBSTATEITEM	*Technical Reference* in the `docs` directory

Note: In Essbase Version 4, retrieval size was set with a variable called REPTKBYTE and was set in the `essbase.cfg` configuration file. You can no longer set the variable in the `essbase.cfg` file.

Setting the Retrieval Sort Buffer Size

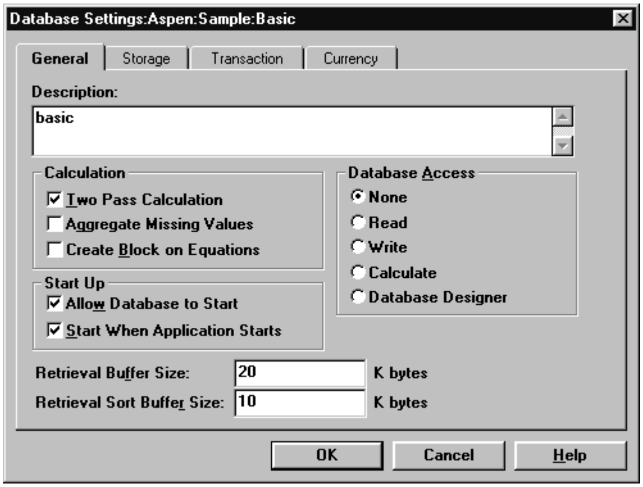
The Retrieval Sort Buffer Size setting specifies the size, in kilobytes, of the server buffer that holds the data to be sorted during an Spreadsheet Add-in or Application Manager Report Writer retrieval. If the sorting buffer is full, Essbase posts an error message.

You can adjust the buffer size on a per-database basis. The default buffer size is set to 10 kilobytes.

➤ To set the retrieval sort buffer size:

1. From the Application Manager menu, choose Database > Settings.
2. Select the General page.
3. Type the number, in kilobytes, of the required sort buffer size in the Retrieval Sort Buffer Size box:

Figure 548: Setting the Retrieval Sort Buffer Size



Description: basic

Calculation
- ☑ Two Pass Calculation
- □ Aggregate Missing Values
- □ Create Block on Equations

Start Up
- ☑ Allow Database to Start
- ☑ Start When Application Starts

Database Access
- ● None
- ○ Read
- ○ Write
- ○ Calculate
- ○ Database Designer

Retrieval Buffer Size: 20 K bytes

Retrieval Sort Buffer Size: 10 K bytes

To determine the best buffer size for your report processing needs, test a report script with different settings.

➤ Use any of these alternate methods to set the retrieval sort buffer size:

Tool	**Instructions**	**Additional Information**
Administration Services	Database Properties window > General tab.	*Essbase Administration Services Online Help*
MaxL	**alter database**	*MaxL Language Reference*
ESSCMD	SETDBSTATEITEM	*Technical Reference* in the `docs` directory

Note: In Essbase Version 4, retrieval sort buffer size was set with a variable called REPTKBYTESORTBUF and was set in the configuration file `essbase.cfg`.

Setting Numeric Precision

The NUMERICPRECISION setting, used by the RESTRICT command, defines the number of precision digits the internal numerical comparison considers in the Report Extractor. If you have a precision setting greater than necessary for your data, your retrieval will be slower than it could be. Identify the correct level of precision and then adjust NUMERICPRECISION accordingly.

➤ To set the NUMERICPRECISION variable:

1. Open the `essbase.cfg` server file.

   Note: The `essbase.cfg` file is created manually. Consult your Database Administrator if the file is not available.

2. Find the following line in the file:

   ```
   NUMERICPRECISION X
   ```

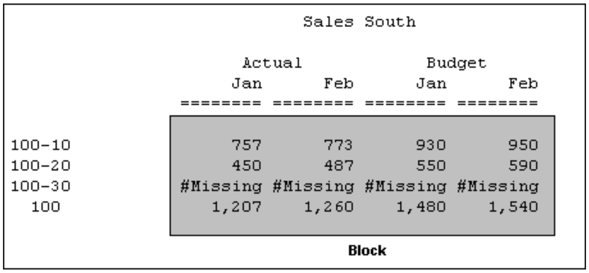
   where *X* is a numeric value representing the number of precision digits that are used by the Report Writer for numerical comparison.

   Note: If the line does not exist, type in the variable information using the example above. See the *Technical Reference* in the `docs` directory for a syntax description and details about the NUMERICPRECISION setting, as well as more information about adding information to the `essbase.cfg` file.

Generating Symmetric Reports

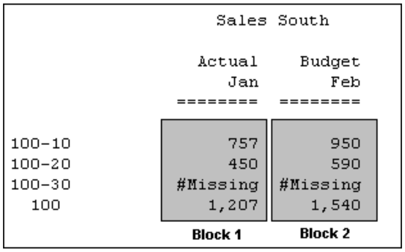
If report processing time is of primary importance, and you are using Report Writer, consider making all reports symmetric. Symmetric reports provide better processing performance than asymmetric reports, as the Report Extractor composes the member list based on all possible member combinations. A list of this nature allows Report Extractor to create the list in one pass. With asymmetric reports, the Extractor must retrieve and process each block of possible member combinations separately

Figure 549: 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
```

Figure 550: Asymmetric Report Member Combinations Requiring Multiple Passes



```
                           Sales South

                      Actual    Budget
                         Jan       Feb
                    ========  ========

100-10                   757       950
100-20                   450       590
100-30              #Missing  #Missing
  100                  1,207     1,540
```

For more information about how the Report Extractor retrieves data, see Chapter 34, “Quick Start to Report Scripts.”

Organizing Members to Optimize Data Extraction

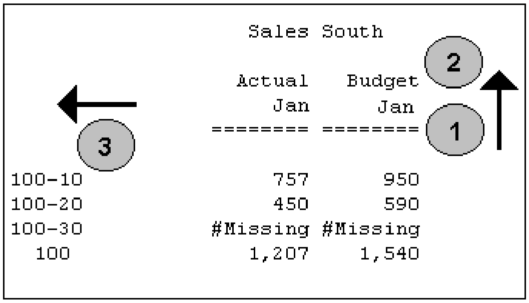
Report Extractor extracts data in a certain order for the Report Writer. If you do not require a formatted report and you are using Report Writer, you can reduce the time required to generate the report by using any of these strategies:

- Creating the report script in the same order as Report Extractor extracts data
- Grouping dense dimensions in columns and grouping sparse dimensions in rows

These strategies save the most time if used to create large production reports.

Report Extractor looks at data from bottom to top and right to left, starting from the bottom column member to the top column member and then proceeding from the innermost row member (right) to the outermost row member (left). Figure 551 illustrates the sequence in which the report is read.

Figure 551: How Report Extractor Examines Data



```
                 Sales South

               Actual   Budget
                  Jan      Jan
             ======== ========

100-10            757      950
100-20            450      590
100-30       #Missing #Missing
  100           1,207    1,540
```

The column members come from dense dimensions, and the row members come from sparse dimensions. To reduce the time needed to extract data, group dense dimensions first, then group sparse dimensions in the same sequence as they are displayed in your outline.

When dense dimensions are nested in the report columns, Report Extractor examines each data block only once, thus improving performance time.

Attributes are sparse dimensions and are dynamically calculated. Hence, Essbase cannot use the sparse data extraction method when a report contains attribute dimensions. For more information on the SPARSE command, see the *Technical Reference* in the `docs` directory.

Understanding Reports for Outlines that Contain Dynamic or Transparent Members

If you generate a report that accesses a database outline that contains Dynamic Calc And Store members, the first time that you generate the report takes longer than subsequent retrievals that access the same data block.

If you generate a report that accesses a database outline that contains Dynamic Calc or Dynamic Time Series members, Essbase calculates the member every time a report is generated, which increases the reporting time.

For more information about dynamic members, see Chapter 28, “Dynamically Calculating Data Values.”

If you run a report that contains transparent members, the report takes longer to generate, as it must access more than one server to retrieve the required data.

Limiting LRO File Sizes for Storage Conservation

Because Essbase stores linked files in a repository on the server, you might want to limit the size of files that users can link. This would prevent a user from taking up too much of the server’s resources by storing extremely large objects. You can set the maximum linked file size for each application. If a user attempts to link a file that is larger than the limit, an error message displays.

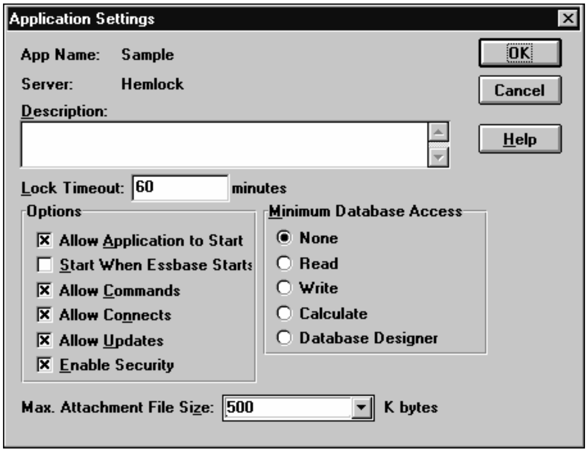
Note: The maximum file size setting applies only to linked files and does not affect cell notes or URLs. The maximum cell note length is fixed at 599 characters. The maximum URL string length is fixed at 512 characters.

➤ To specify a maximum file size for an application (by default the size is unlimited), use this procedure:

1. In Application Manager, connect to the appropriate server and select the name of the application.

2. Select Application > Settings. The Application dialog box is displayed:

Figure 552: Application Settings Dialog Box



App Name: Sample
Server: Hemlock
Description:
Lock Timeout: 60 minutes

Options
- ☒ Allow Application to Start
- □ Start When Essbase Starts
- ☒ Allow Commands
- ☒ Allow Connects
- ☒ Allow Updates
- ☒ Enable Security

Minimum Database Access
- ● None
- ○ Read
- ○ Write
- ○ Calculate
- ○ Database Designer

Max. Attachment File Size: 500 K bytes

3. Enter the maximum file size (in kilobytes) in the Max. Attachment File Size text box.

   To prevent users from attaching anything except very small files, enter 1. This lets users link only cell notes, URLs, and files less than 1 kilobyte in size.

4. Click OK to save your setting.

Tip: You can set a limit on the size of a linked object without using Application Manager.

Tool	Instructions	For more information
Administration Services	Application Properties window > General tab	*Essbase Administration Services Online Help*
MaxL	**alter application**	*MaxL Language Reference*
ESSCMD	SETAPPSTATE	*Technical Reference* in the `docs` directory Chapter 45, “Automating the Production Environment” for information about ESSCMD.

Appendix A Limits

This appendix contains a list of limits you may encounter when creating or manipulating Essbase objects:

- Object name length and related object value limits
- Size limits or quantity limits
- Data load and dimension build limits

Table 92: Objects and Limits

Object	**Description of Limit**
Names and Related Fields	
Alias name	80 bytes
Alias table	30 bytes
Application name	8 bytes
Application description	79 bytes
Custom-defined function name Custom-defined macro name	127 bytes. After 127 bytes, characters truncated by MaxL and API. No truncation on server. No error occurs.
Custom-defined function and macro specification	127 bytes. After 127 bytes, characters truncated by MaxL and API. No truncation on server. No error occurs.
Custom-defined function and macro comment	255 bytes. After 127 bytes, characters truncated. by MaxL and API. No truncation on server. No error occurs.
Database name	8 bytes
Database description	79 bytes
Directory path	255 bytes. For example: `/essbase/bin`
Filter name	30 bytes

Table 92: Objects and Limits (Continued)

Object	**Description of Limit**
Group name	30 bytes
Linked reporting object cell note	599 bytes
Linked reporting object URL	512 characters (always single-byte characters)
Member comment field	255 bytes
Member comment field (extended)	8192 bytes
Member name	30 characters
OLAP Server name	30 bytes
Password	101 bytes. After 100 single-byte characters or 50 double-byte characters, Essbase will not let the user log in and reports an error.
Substitution variable name	80 bytes
Substitution variable value	255 bytes
User names	30 bytes
Variable names	32 bytes
Data Load and Dimension Building Limits	
Selection and rejection criteria	Number of characters that describe selection and rejection criteria: combination of all criteria limited to 32 KB
`essbase.cfg`-Releated Limits	
`AUTHENTICATIONMODULE`	256 byte limit: parameter *default_connection_parameter*
Other Limits	
Caches: data, data file, index	2 GB
DATAERRORLIMIT, number of error messages written to a log.	Default 1000, minimum 1, maximum 65000
Formula size	• Limited to 64 KB if created with Formula Editor. Formulas in calculation scripts are not subject to this limit. • Formulas created in MaxL, using multi-byte characters, are limited to 40 KB.

Table 92: Objects and Limits (Continued)

Object	Description of Limit
Number of security filters	Per OLAP Server, 32767 Per Essbase database, 32290
Number of users	30,000. Errors may occur if you create more than 30,000 users.
Number of members in an outline	1,000,000 explicitly defined in an Essbase outline. Because Hybrid Analysis and some uses of partitions can involve many more members than are explicitly listed in an outline, the actual number of members accessible through the database is much higher.

Appendix

B

Error Handling and Troubleshooting for Essbase

This chapter describes tools that you can use to diagnose errors, and suggests methods for correcting some errors:

- "Understanding Fatal Error Handling" on page 1460
- "Recovering from Full Restructure Failure" on page 1461
- "Recovering from Sparse Restructure Failure" on page 1461
- "Synchronizing Member Names in a Report Script and Database Outline" on page 1461
- "Multiple Reports and Server Problem" on page 1462

Note: Chapters related to partitions, currency conversion, and data load have basic troubleshooting information in the relevant chapter. For information about restoring from backups, see "Restoring Data from Backups" on page 1255. For information about specific error messages, see the Error Messages Guide in `/docs/errmsgs` in your Essbase installation. For information about error and exception logs, see Chapter 43, "Using Essbase Logs."

Understanding Fatal Error Handling

The Essbase Kernel considers the following errors fatal:

- One or more control fields have unexpected or inconsistent values.
- The kernel detects data corruption.
- The kernel cannot perform an operation that is necessary to ensure data integrity (for example, disk space is insufficient).
- The kernel encounters a condition that can lead to data corruption.

When the kernel encounters a fatal error, it shuts down and restarts, attempting to reinitialize itself and proceed with database recovery. When recovery begins, Essbase displays an error message similar to this one:

```
1080022 Reinitializing the Essbase Kernel for database
database_name due to a fatal error ...
```

This message is followed by other informational messages related to database recovery, such as this one:

```
1080028 Performing transaction recovery for database
database_name during fatal error processing.
```

When you see such messages, you know that the kernel shut itself down and is attempting to start up again. Check the OLAP Server log and determine whether Essbase issued a fatal error message just before it generated the reinitialization messages. For information on viewing the OLAP Server log, see "Viewing the Server and Application Logs" on page 1225.

If the kernel did encounter a fatal error, in most cases you need to reinitiate any operation that was active at the time of the fatal error. If the operation was a calculation or a data load, you may be able to continue where the operation left off; check the OLAP Server log to see how far Essbase processed the operation. When in doubt, reinitiate the operation. For more information, see "What to Expect If a Server Interruption Occurs" on page 1148.

If the kernel did not encounter a fatal error, contact your software provider's technical support to determine what caused the kernel to shut down and restart.

See "Understanding the Contents of the OLAP Server Log" on page 1209 for information about the OLAP Server log. See "Server and Application Log Message Categories" on page 1219 for information about identifying the component where the error occurred.

Recovering from Full Restructure Failure

If an error or system failure occurs while Essbase is restructuring, it is most likely to occur during Step 2 in the procedure "Understanding a Full Restructure" on page 1346.

➤ To recover from a failure during Step 2 of "Understanding a Full Restructure" on page 1346:

1. Delete the temporary files, both to free up disk space and to avoid conflicts the next time you restructure the database.
2. Restart the database.

➤ To recover from a failure during Step 1, Step 3, or Step 4 of "Understanding a Full Restructure" on page 1346:

1. Review the disk directory and determine how far the restructuring has progressed.
2. If all but Step 4 is complete, rename the temporary files to the correct file names

Recovering from Sparse Restructure Failure

If a system failure occurs during any of step of a sparse restructure, you can recover by restarting the database.

Synchronizing Member Names in a Report Script and Database Outline

When you run a report, it is important to ensure that the member names in the report script match the member names in the database outline. An error displays every time the Report Extractor cannot find a matching member name, and you must correct the name in the report script before the report continues processing.

Multiple Reports and Server Problem

Your server machine may freeze if you try to run more reports in parallel than you have assigned server thread resources.

If you are running multiple report scripts and your server freezes, check the value of the configuration file setting SERVERTHREADS in the server `essbase.cfg` file. There should be at least one thread for each report running. For example, if you are running 22 reports, the value for SERVERTHREADS should be at least 22.

Appendix C

Estimating Disk and Memory Requirements

This chapter helps you estimate disk and memory requirements. This chapter contains the following sections:

- "Understanding How Essbase Stores Data" on page 1463
- "Determining Disk Space Requirements" on page 1465
- "Estimating Memory Requirements" on page 1482

Note: If you are migrating from an earlier version of Essbase, see the *Essbase Installation Guide* for additional information about estimating space requirements.

This chapter uses a worksheet approach to help you keep track of the many components that you calculate. If you are using the printed version of this book, you can photocopy the worksheets. Otherwise, you can simulate the worksheets on your own paper. Labels, such as DA and MA help you keep track of the various calculated disk and memory component values,

Understanding How Essbase Stores Data

You need to understand the units of storage that Essbase uses in order to size a database. This discussion assumes that you are familiar with the following basic concepts before you continue:

- How Essbase stores data, as described in Chapter 3, "Basic Architectural Elements"
- How Essbase Kernel components manage Essbase data, as described in Chapter 38, "Managing Essbase Kernel Settings"

An Essbase database consists of many different components. In addition to an outline file and a data file, Essbase uses several types of files and memory structures to manage data storage, calculation, and retrieval operations.

Table 93 describes the major components that you must consider when you estimate the disk and memory requirements of a database. "Yes" means the type of storage indicated is relevant, "No" means the type of storage is not relevant.

Table 93: Storage Units Relevant to Calculation of Disk and Memory Requirements

Storage Unit	Description	Disk	Memory
Outline	A structure that defines all elements of a database. The number of members in an outline determines the size of the outline.	Yes	Yes
Data files	Files in which Essbase stores data values in data blocks in data files. Named `ess`*`xxxxx`*`.pag,` where *xxxxx* is a number. Essbase increments the number, starting with `ess00001.pag`, on each disk volume. Memory is also affected because Essbase copies the files into memory.	Yes	Yes
Data blocks	Subdivisions of a data file. Each block is a multidimensional array that represents all cells of all dense dimensions relative to a particular intersection of sparse dimensions.	Yes	Yes
Index files	Files that Essbase uses to retrieve data blocks from data files. Named `ess`*`xxxxx`*`.ind`, where *xxxxx* is a number. Essbase increments the number, starting with `ess00001.ind`, on each disk volume	Yes	Yes
Index pages	Subdivisions of an index file. Contain index entries that point to data blocks. The size of index pages is fixed at 8 KB.	Yes	Yes
Index cache	A buffer in memory that holds index pages. Essbase allocates memory to the index cache at startup of the database.	No	Yes

Table 93: Storage Units Relevant to Calculation of Disk and Memory Requirements (Continued)

Storage Unit	Description	Disk	Memory
Data file cache	A buffer in memory that holds data files. When direct I/O is used, Essbase allocates memory to the data file cache during data load, calculation, and retrieval operations, as needed. Not used with buffered I/O.	No	Yes
Data cache	A buffer in memory that holds data blocks. Essbase allocates memory to the data cache during data load, calculation, and retrieval operations, as needed.	No	Yes
Calculator cache	A buffer in memory that Essbase uses to create and track data blocks during calculation operations.	No	Yes

Determining Disk Space Requirements

Essbase uses disk space for its server software and for each database. Before estimating disk storage requirements for a database, you must know how many dimensions the database includes, the sparsity and density of the dimensions, the number of members in each dimension, and how many of the members are stored members.

➤ To calculate the disk space required for a database:

1. Calculate the "Factors To Be Used in Sizing Disk Requirements" on page 1466.
2. Use the "Estimating Disk Space Requirements for a Single Database" on page 1472 to calculate the space required for each component of a single database. If your server contains more than one database, you must perform calculations for each database.
3. Use the "Estimating the Total Server Disk Space Requirement" on page 1481 to calculate the final estimate for the server.

Note: The database sizing calculations in this chapter assume an ideal scenario with an optimum database design and unlimited disk space. The amount of space required is difficult to determine precisely because most multidimensional applications are sparse.

Factors To Be Used in Sizing Disk Requirements

Before estimating disk space requirements for a database, you must calculate the factors to be used in calculating the estimate. Later in the chapter you will use these values to calculate the components of a database. For each database, you will then add together the sizes of its components.

Table 94 lists the sections that provide instructions to calculate these factors. Go to the section indicated, perform the calculation, then write the calculated value in the Value column.

Table 94: Factors Affecting Disk Space Requirements of a Database

Database Sizing Factor	**Label**	**Value**
"Potential Number of Data Blocks" on page 1466	DA	
"Number of Existing Data Blocks" on page 1468	DB	
"Size of Expanded Data Block" on page 1469	DC	
"Size of Compressed Data Block" on page 1471	DD	

Potential Number of Data Blocks

The potential number of data blocks is the maximum number of data blocks possible in the database.

If the database is already loaded, you can see the potential number of blocks on the Statistics tab of the Database Information dialog box of Application Manager or on the Statistics tab of the Database Properties dialog box of Essbase Administration Services.

If the database is not already loaded, you must calculate the value.

➤ To determine the potential number of data blocks, assume that data values exist for all combinations of stored members.

1. Using Table 95 on page 1467 as a worksheet, list each sparse dimension and its number of stored members. If there are more than seven sparse dimensions, list the dimensions elsewhere and include all sparse dimensions in the calculation.

The following types of members are not stored members:

- Members from attribute dimensions
- Shared members
- Label Only members
- Dynamic Calc members (Dynamic Calc And Store members are stored members)

2. Multiply the number of stored members of the first sparse dimension (line a.) by the number of stored members of the second sparse dimension (line b.) by the number of stored members of the third sparse dimension (line c.), and so on. Write the resulting value to the cell labeled DA in Table 94 on page 1466.

```
a * b * c * d * e * f * g (and so on) = potential number
of blocks
```

Table 95: List of Sparse Dimensions with Numbers of Stored Members

Enter Sparse Dimension Name	Enter Number of Stored Members	
	a.	
	b.	
	c.	
	d.	
	e.	
	f.	
	g.	

Example

The Sample Basic database contains the following sparse dimensions:

- Product (19 stored members)
- Market (25 stored members)

Therefore, there are 19 * 25 = 475 potential data blocks.

Number of Existing Data Blocks

As compared with the potential number of blocks, the term existing blocks refers to those data blocks that Essbase actually creates. For Essbase to create a block, at least one value must exist for a combination of stored members from sparse dimensions. Because many combinations can be missing, the number of existing data blocks is usually much less than the potential number of data blocks.

If the database is already loaded, you can see the number of existing blocks on the Statistics tab of the Database Information dialog box of Essbase Application Manager or on the Statistics tab of the Database Properties dialog box of Essbase Administration Services. Write the value in the cell labeled DB in Table 94 on page 1466.

If the database is not already loaded, you must estimate a value.

➤ To estimate the number of existing data blocks:

1. Estimate a database density factor that represents the percentage of sparse dimension stored-member combinations that have values.
2. Multiply this percentage against the potential number of data blocks and write the number of actual blocks to the cell labeled DB in Table 94 on page 1466.

```
number of existing blocks = estimated density * potential
number of blocks
```

Example

The following three examples show different levels of sparsity and assume 100,000,000 potential data blocks:

- Extremely sparse: Only 5 percent of potential data cells exist.

```
.05 (estimated density) * 100,000,000 (potential blocks) =
5,000,000 existing blocks
```

- Sparse: 15 percent of potential data cells exist.

```
.15 (estimated density) * 100,000,000 (potential blocks) =
15,000,000 existing blocks
```

- Dense: 50 percent of potential data cells exist.

```
.50 (estimated density) * 100,000,000 (potential blocks) =
50,000,000 existing blocks
```

Size of Expanded Data Block

The potential, expanded (uncompressed) size of each data block is based on the number of cells in a block and the number of bytes used for each cell. The number of cells in a block is based on the number of stored members in the dense dimensions. Essbase uses eight bytes to store each intersecting value in a block.

If the database is already loaded, you can see the size of an expanded data block on the Statistics tab of the Database Information dialog box of Application Manager or on the Statistics tab of the Database Properties dialog box of Essbase Administration Services.

If the database is not already loaded, you must estimate the value.

➤ To determine the size of an expanded data block:

1. Using Table 96 on page 1470 as a worksheet, enter each dense dimension and its number of stored members. If there are more than seven dense dimensions, list the dimensions elsewhere and include all dense dimensions in the calculation.

   The following types of members are not stored members:

   - Members from attribute dimensions
   - Shared members
   - Label Only members
   - Dynamic Calc members (Dynamic Calc And Store members are stored members.)

2. Multiply the number of stored members of the first dense dimension (line a) by the number of stored members of the second dense dimension (line b) by the number of stored members of the third dense dimension (line c), and so on, to determine the total number of cells in a block.

   ```
   a * b * c * d * e * f * g (and so on) = the total number of cells
   ```

3. Multiply the resulting number of cells by 8 bytes to determine the expanded block size. Write the resulting value to the cell labeled DC in Table 94 on page 1466.

```
(Total number of cells) * 8 bytes per cell = expanded
block size
```

Table 96: Determining the Size of a Data Block

Enter Dense Dimension Name	Number of Stored Members	
	a.	
	b.	
	c.	
	d.	
	e.	
	f.	
	g.	

Example

The Sample Basic database contains the following dense dimensions:

- Year (12 stored members)
- Measures (8 stored members)
- Scenario (2 stored members)

Perform the following calculations to determine the potential size of a data block in Sample Basic:

```
12 * 8 * 2 = 192 data cells
192 data cells * 8 bytes = 1,536 bytes (potential data block size)
```

Size of Compressed Data Block

Compression affects the actual disk space used by a data file. The two types of compression, bitmap and run-length encoding (RLE), affect disk space differently. For information about data compression unrelated to estimating size requirements, see "Data Compression" on page 1124.

If you are not using compression or if you have enabled RLE compression, skip this calculation and proceed to "Compressed Data Files" on page 1473.

Note: Due to sparsity also existing in the block, actual (compressed) block density varies widely from block to block. The calculations in this discussion are only for estimation purposes.

➤ To calculate an average compressed block size when bitmap compression is enabled:

1. Determine an average block density value.
   - If the database is already loaded, you can see the size of an expanded data block on the **Statistics** tab of the **Database Information** dialog box of Application Manager or on the **Statistics** tab of the **Database Properties** dialog box of Essbase Administration Services. Use the value that is displayed for **Block Density**.
   - If you want to estimate block density prior to loading data, estimate the ratio of existing data values to potential data values.
2. To determine the compressed block size, perform the following calculation and write the resulting block size to the cell labeled DD in Table 94 on page 1466.

   ```
   expanded block size * block density = Compressed block size
   ```

Example

Assume an expanded block size of 1,536 bytes and a block density of 25%:

```
1,536 bytes * .25 = 384 bytes (compressed block size)
```

Estimating Disk Space Requirements for a Single Database

To estimate the disk-space requirement for a database, make a copy of Table 97 or use a separate sheet of paper as a worksheet for a single database. If multiple databases are on a server, repeat this process for each database. Write the name of the database on the worksheet.

Each row of this worksheet refers to a section that describes how to size that component. Perform each calculation and write the results in the appropriate cell in the Size column. The calculations use the factors that you wrote in Table 94 on page 1466.

Table 97: Worksheet for Estimating Disk Requirements for a Database

Database Name:	
Database Component	**Size**
"Compressed Data Files" on page 1473	DE
"Fixed-Size Overhead" on page 1473	DF
"Index Files" on page 1476	DG
"Fragmentation Allowance" on page 1476	DH
"Outline" on page 1477	DI
"Work Areas" on page 1479 (sum of DE through DI)	DJ
"Linked Reporting Objects Considerations" on page 1480, if needed	DK
Total disk space required for the database. Total the size values from DE through DK and write the result to Table 99 on page 1482.	

After writing all the sizes in the Size column, add them together to determine the disk space requirement for the database. Add the database name and size to the list in Table 99 on page 1482. Table 99 is a worksheet for determining the disk space requirement for all databases on the server.

Repeat this exercise for each database on the server. After estimating disk space for all databases on the server, proceed to "Estimating the Total Server Disk Space Requirement" on page 1481.

The following sections describe the calculations to use to estimate components that affect the disk-space requirements of a database.

Compressed Data Files

The calculation for the space required to store the compressed data files (`ess`*`xxxxx`*`.pag`) uses the following factors from Table 94 on page 1466:

- Number of existing blocks (value DB)
- Compressed block size (value DD)

To estimate the space required for the compressed data files, multiply the compressed block size by the number of data blocks and write the size of the compressed data files cell labeled DE in Table 97 on page 1472.

Note: If compression is not used, substitute the expanded block size for the compressed block size in this formula.

Example

```
384 (compressed block size) * 15,000,000 (number of data blocks)
= 5,760,000,000 bytes (size of compressed data files)
```

Fixed-Size Overhead

The following subtopics show how to calculate fixed-size overhead. Use one of two methods of calculation, depending whether the database uses bitmap compression, run-length encoding (RLE), or no compression.

Fixed Size Overhead Using Bitmap Compression

Calculations for fixed-size overhead using bitmap compression use the following factors and constants:

- Number of existing blocks (value DB from Table 94 on page 1466)
- Expanded block size (value DC from Table 94 on page 1466)
- 72 bytes—block header size
- 64 bytes

The compression bitmap uses one bit for each cell in a block. Dividing the expanded block size by 8 provides the number of cells, which equals the number of bits in the bitmap. Dividing this value again by 8 determines the number of bytes; therefore the following procedure divides the expanded block size by 64 to obtain the fixed-size overhead for each block.

➤ To calculate the fixed-size overhead when bitmap compression is enabled:

1. Determine the fixed-size overhead per block. Perform the following calculation.

   ```
   ((expanded block size in bytes/64)+72) = temporary value
   ```

2. Round up the temporary value from step 1 to the nearest multiple of eight.

   a. Divide the number by 8.

   b. Use the whole number only.

   c. If anything is left over add 1.

   d. Multiply by 8.

   The result of this calculation is the fixed-size overhead per block.

3. Determine the fixed-size overhead for the database. Perform the following calculation and write the resulting value to the cell labeled DF in Table 97 on page 1472.

   ```
   fixed-size overhead per block * number of existing blocks
   = fixed-size overhead for the database
   ```

Example

Assume bitmap compression and an expanded block size of 4,802 bytes with a total of 15,000,000 existing blocks.

1. Calculate the formula, then round the result to the nearest multiple of eight.

   ```
   (4,802 / 64) + 72 = 147.03 bytes
   ```

2. Round to the next multiple of eight, as shown in Table 98 on page 1475.

3. Multiply the overhead per block by the number of blocks:

```
152 bytes * 15,000,000 blocks = 2,280,000,000 bytes
(database overhead)
```

Table 98: Fixed Size Overhead Calculation

Calculation for Rounding Up	Result
1. Divide the number of bytes by 8	147.03 / 8 = 18.3
2. Use the whole number only	18
3. If anything is left over add 1	18 + 1 = 19
4. Multiply by 8	19 * 8 = 152 bytes in overhead per block

Fixed Size Overhead Using Run-Length Encoding (RLE) or No Compression

Calculations for fixed-size overhead for databases not using bitmap compression use the following factors and constants:

- Number of existing blocks (value DB from Table 94 on page 1466)
- Expanded block size (value DC from Table 94 on page 1466)
- 72 bytes—block header size

➤ To calculate the fixed-size overhead for a database that uses RLE or no compression:

1. Using the 72-byte header as the fixed-size overhead per block.
2. Determine the fixed-size overhead for the database. Perform the following calculation and write the resulting value to the cell labeled DF in Table 97 on page 1472.

```
72 * Number of existing blocks
= Fixed-size overhead for the database
```

Example

Assume a total of 15,000,000 existing blocks.

```
72 bytes * 15,000,000 blocks = 1,080,000,000 bytes
```

Index Files

The calculation for the space required to store the index files (`ess`*`xxxxx`*`.ind`) uses the following factors:

- Number of existing blocks (value DB from Table 94 on page 1466)
- 112 bytes—the size of an index entry

To calculate the total size of a database index, including all index files. perform the following calculation. Write the size of the compressed data files to the cell labeled DG in Table 97 on page 1472.

```
number of existing blocks * 112 bytes = the size of database index
```

Example

Assume a database with 15,000,000 blocks.

```
15,000,000 blocks * 112 = 1,680,000,000 bytes
```

Note: If the database is already loaded, select Database > Information in Application Manager and look at the Files tab for the size of the index file. If you are using Essbase Administration Services, click the Storage tab on the Database Properties window.

Fragmentation Allowance

If you are using bitmap or RLE compression, a certain amount of fragmentation occurs. The amount of fragmentation is based on individual database and operating system configurations and cannot be precisely predicted.

As a rough estimate, calculate 20% of the compressed database size (value DE from Table 97 on page 1472) and write the result to the cell labeled DH in the same table.

Example

Assume a compressed database size of 5,769,000,000 bytes.

```
5,769,000,000 bytes * .2 = 1,153,800,000 bytes
```

Outline

The space required by an outline can have two components.

- The main area of the outline is a component of both disk and memory space requirements and is calculated using the following factors:
  - The number of members in the outline
  - The length, in characters, of member names and aliases
- The attribute association area of an outline is a component only of disk space and is calculated using the following factors:
  - The number of members in each base dimension
  - The number of members in each attribute dimension

➤ To estimate the size of the outline:

1. Estimate the main area of the outline by multiplying the number of members by a name-length factor between 350 and 450 bytes.

   If the database includes few aliases or very short aliases and short member names, use a smaller number within this range. If you know that the names or aliases are very long, use a larger number within this range.

   Because the name-length factor is an estimated average, the following formula provides only a rough estimate of the main area of the outline.

   ```
   number of members * name-length factor = size of main area
   of outline
   ```

   Note: See Appendix A, “Limits,” for the maximum sizes for member names and aliases.

   For memory space requirements calculated later in this chapter, use the size of the main area of the outline.

2. For disk space requirements, if the outline includes attribute dimensions, calculate the size of the attribute association area for the database. Calculate the size of this area for each base dimension. Multiply the number of members of the base dimension by the sum of the count of members of all attribute dimensions associated with the base dimension, and then divide by 8.

   Note: Within the count of members, do not include Label Only members and shared members.

```
(number of base-dimension members * sum of count of
attribute-dimension members)/8 = size of attribute
association area for a base dimension
```

3. Sum the attribute association areas of each dimension to determine the total attribute association area for the outline.

4. For the total disk space required for the outline, add together the main outline area and the attribute association area, and write the result of this calculation to the cell labeled DI in Table 97 on page 1472.

   ```
   main area of outline + total attribute association area =
   total disk space required for the outline
   ```

Example

Assume the outline has the following characteristics:

- 26,000 members
- A name-length factor of 400 bytes
- A base dimension Product (23,000 members—excluding Label Only members and shared members) with two attribute dimensions associated with it: Ounces (20 members) and Pkg Type (50 members)
- A base dimension Market (2,500 members—excluding Label Only members and shared members) with one associated attribute dimension, Population (12 members)

1. Calculate the main area of the outline:

   ```
   400 bytes (name-length factor) x 26,000 members = 10,400,000
   bytes
   ```

2. Calculate the attribute association areas:

   - For the base dimension Product:

     ```
     (23,000 * (20 + 50)) bits / 8 bits per byte = 201,250 bytes
     ```

   - For the base dimension Market:

     ```
     (2,500 * 12) bits / 8 bits per byte = 3,750 bytes
     ```

3. Sum these areas for the total attribute association area for the database:

   ```
   201,250 bytes + 3,750 bytes = 205,000 bytes
   ```

4. For a total estimate of outline disk space, add the main area of the outline and the total attribute association area:

   ```
   10,400,000 bytes + 205,000 bytes = 10,605,000 bytes (outline
   disk space requirement)
   ```

Note: Do not use this procedure to calculate outline memory space requirements. See "The Outline Size Used in Memory" on page 1486.

Work Areas

Three different processes create temporary work areas on the disk:

- For recovery purposes, Essbase maintains a data recovery area on the disk. The size of this area increases until the database is restructured.
- During restructuring, Essbase uses a restructuring work area on the disk.
- During migration from prior releases of Essbase, for recovery purposes, Essbase creates a copy of the database in a migration work area.

To create these temporary work areas, Essbase may require disk space equal to the size of the entire database. Restructuring and migration need additional work space the size of the outline. Because none of these activities occur at the same time, a single allocation can represent all three requirements.

To calculate the size of a work area used for restructuring, migration, and recovery, calculate the sum of the sizes of the following database components from Table 97 on page 1472:

- Compressed data files (value DE)
- Fixed-size overhead (value DF)
- Index files (value DG)
- Fragmentation allowance (value DH)
- Outline (value DI)

```
work area = size of compressed data files + fixed-size overhead
+ size of index files + fragmentation allowance + outline size
```

Write the result of this calculation to the cell labeled DJ in Table 97 on page 1472.

Linked Reporting Objects Considerations

You can use the Linked Reporting Objects (LROs) feature to associate objects with data cells. The objects can be flat files, HTML files, graphics files, and cell notes. For information about linked reporting objects, see Chapter 11, "Linking Objects to Essbase Data."

Two aspects of LROs affect disk space:

- The size of the object. Because Essbase copies files used as LROs to the server, you must know the combined size of all files you are using as LROs.

  Note: You can set a limit on the size of a linked object, if the linked object is a file (as opposed to a cell note). For information, see "Limiting LRO File Sizes for Storage Conservation" on page 1452.

- The size of the LRO catalog, where the Essbase Kernel stores information about LROs. Cell notes and URLs are also stored in the catalog. A catalog entry is stored as an index page. For every catalog entry, Essbase uses 8 KB.

➤ To estimate the disk space requirements for linked reporting objects:

1. Estimate the size of the objects. If a limit is set, multiply the number of LROs by that limit. Otherwise, sum the size of all anticipated LROs.
2. Size the LRO catalog. Multiply the total number of LROs by 8192 bytes.
3. Add together the two areas and write the result of this calculation to the cell labeled DK in Table 97 on page 1472.

```
sum of LRO sizes + size of LRO catalog = LRO disk space
requirement
```

Example

Assume the database uses 1500 LROs which are composed composed of:

- 1000 URLs, at a maximum of 512 bytes each
- 500 cell notes

1. Multiply 1000 * 512 bytes for 512,000 bytes maximum required for the stored URLs.
2. Calculate the size of the LRO catalog. Multiply 1500 total LROs * 8192 bytes = 12,288,000 bytes.

3. Add together the two areas:

```
512,000 bytes + 12,288,000 bytes = 12,800,000 bytes total LRO
disk space requirement
```

Estimating the Total Server Disk Space Requirement

The earlier calculations in this chapter estimate the data storage requirement for a single database. Often, more than one database resides on the server.

In addition to the data storage required for each database, the total Essbase data storage requirement on a server includes Essbase software. Allow approximately 80 to 202 MB (84,451,328 to 211,953,664 bytes) for the base installation of Essbase software and sample applications. The allowance varies by platform and file management system. For details, see the *Essbase Installation Guide*.

➤ To estimate the total server disk space requirement:

1. In the worksheet in Table 99 on page 1482, list the names and disk space requirements that you calculated for each database.
2. Sum the database requirements and write the total in the cell labeled DL.
3. In the cell labeled DM, write the appropriate disk space requirement for the software installed on the server.
   - If server software only, write 84,451,328 bytes.
   - If server, client, and API software, write 211,953,664 bytes.
4. For the total server disk space requirement in bytes, sum the values in cells DL and DM. Write this value in the cell labeled DN.
5. To convert to megabytes (MB), divide the value in cell DN by 1,048,576 bytes. Write this value in the cell labeled DO.

Table 99: Worksheet for Total Server Disk Space Requirement

List of Databases **(From** Table 97 on page 1472**)**		**Size**
a.		
b.		
c.		
d.		
e.		
f.		
g.		
Sum of database disk sizes a + b + c + d + e + f + g		DL:
84,451,328 to 211,953,664 bytes for Essbase server software		DM:
Total Essbase server disk requirement in bytes: DL + DM		DN:
Total Essbase server disk requirement in megabytes (MB): DN divided by 1,048,576 bytes		DO:

Estimating Memory Requirements

The minimum memory requirement for running Essbase is 64 MB. On UNIX systems, the minimum requirement is 128 MB. Based on the number of applications and databases and the database operations on the server, the amount of memory you require may be more.

To estimate the memory required on the server:

1. Calculate the startup memory requirement for each application.
2. Using the worksheet in Table 97 on page 1472 to estimating memory requirements for each database, calculate the following components:
   - Startup memory used for each database
   - The memory used for database operations such as data retrievals and calculations

3. Add these requirements together for all databases and applications, as shown in the Worksheet for Total Memory Requirement in Table 103 on page 1496.

Startup Memory Requirement for an Application

Each open application has the following memory requirement at startup:

- Essbase code and static data (10 MB). This number may be more or less, depending on the operating system used.
- (Optional) Java Virtual Machine (2 to 4 MB), which is used for custom-defined functions. This value depends on the operating system used.
- (Optional) Essbase Administration Services (15 MB).

Multiply the number of applications that will be running simultaneously on the server by the appropriate startup requirement and write the resulting value to the cell labeled ML in Table 103 on page 1496.

Startup Memory Requirement for a Single Database

To estimate the memory requirement for a database, make a copy of Table 100 or use a separate sheet of paper as a worksheet for a single database. If multiple databases are on a server, repeat this process for each database. Write the name of the database on the worksheet.

Each row links to information that describes how to size that component. Perform each calculation and note the results in the appropriate cell in the Size column. Some calculations use the factors that you wrote in Table 101 on page 1485. After filling in all the sizes in the Size column, add them together to determine the memory requirement for that database.

After estimating disk space for all databases on the server, proceed to "Estimating the Total Server Disk Space Requirement" on page 1481.

Table 100: Worksheet for Estimating Memory Requirements for a Database

Database Name:	
Memory Requirement	**Size**
Startup requirements per database:	
Database outline. See "The Outline Size Used in Memory" on page 1486.	MA:
Index cache. See "Sizing the Index Cache" on page 1320.	MB:
Cache-related overhead. See "Cache-Related Overhead" on page 1487.	MC:
Area for data structures. See "Memory Area for Data Structures" on page 1489.	MD:
Operational requirements:	
Memory used for data retrievals. See "Estimating Additional Memory Requirements for Data Retrievals" on page 1490	ME:
Memory used for calculations. See "Estimating Additional Memory Requirements for Calculations" on page 1495	MF:
Summarize the size values from MB through MF for an estimate of the total memory required for a database.	MG
Divide the value from MG. by 1,048,576 bytes for the total database memory requirement in megabytes (MB).	MH

In Table 103 on page 1496, enter the name of the database and the total memory requirement in megabytes, MH.

Factors To Be Used in Sizing Memory Requirements

Before you start the estimate, calculate factors to be used in calculating the estimate.

Table 101 on page 1485 lists sizing factors with references to sections in this and other chapters that provide information to determine these sizes. Go to the section indicated, perform the calculation, then return to Table 101 and write the size, in bytes, in the Value column of this table.

Later in this chapter, you can refer to Table 101 for values to use in various calculations.

Table 101: Factors Used to Calculate Database Memory Requirements

Database Sizing Factor	**Value**
The number of cells in a logical block. See “The Number of Cells in a Logical Block” on page 1488.	MI:
The number of threads allocated through the ESSCMD, SERVERTHREADS. See the *Technical Reference* in the `docs` directory.	MJ:
Potential stored-block size. See “Size of Expanded Data Block” on page 1469.	MK:

The calculations in this chapter do not account for other factors that affect how much memory is used. The following factors have complex implications and their affects on memory size cannot be calculated:

- Cache memory locking. Whether or not cache memory locking is enabled affects how the operating system and Essbase manage memory. See “Before You Size: Using Cache Memory Locking” on page 1317.
- Different operation types and their associated cache allocations. Data load, data retrieval, and calculation operations set aside memory for the data file, data, and calculation caches, plus some overhead associated with the caches.
- The sizes of the retrieval buffer and the retrieval sort buffer. See Chapter 53, “Optimizing Reports and Other Types of Retrieval.”
- Database workload; for example, the complexity of a calculation script or the number and complexity of data queries.

- The number of data blocks defined using the `CALCLOCKBLOCK` setting in the `ESSBASE.CFG` file in combination with the `SET LOCKBLOCK` setting, which specifies which `CALCLOCKBLOCK` setting to use. See "Locking Blocks During Calculation" on page 1404.
- The number of Dynamic Calc members in the outline, including members of attribute dimensions.
- The isolation level settings. See "Isolation Levels" on page 1132.
- Synchronization points. See "Uncommitted Access" on page 1137.

The Outline Size Used in Memory

The attribute association area included in disk space calculations is not a sizing factor for memory. Calculate only the main area of the outline.

For memory size requirements, outline size is calculated using the following factors:

- The number of members in the outline
- The length, in characters, of member names and aliases

➤ To calculate the outline memory requirement, multiply the number of members by a name-length factor between 350 and 450 bytes and write the result to the cell labeled MA in Table 100 on page 1484.

If the database includes few aliases or very short aliases and short member names, use a smaller number within the 350–450 byte range. If you know that the names or aliases are very long, use a larger number within this range.

Because the name-length factor is an estimated average, the following formula provides only a rough estimate of the main area of the outline:

```
memory size of outline = number of members * name-length factor
```

Note: See Appendix A, "Limits," for the maximum sizes for member names and aliases.

Example

Assuming the outline has 26,000 members and a median name-length, use the following calculation to estimate the outline size used in memory:

```
26,000 members * 400 bytes per member = 10,400,000 bytes
```

Index Cache

At startup, Essbase sets aside memory for the index cache, the size of which can be user-specified. To determine the size of the index cache, see "Sizing the Index Cache" on page 1320 and write the size in the cell labeled MB in Table 100 on page 1484.

Cache-Related Overhead

Essbase uses additional memory while it works with the caches.

The calculation for this cache-related overhead uses the following factors:

- Index cache (value MB from Table 100 on page 1484)
- The number of server threads (value MJ from Table 101 on page 1485)

➤ To calculate the cache-related overhead at startup:

1. Calculate half the index cache size, in bytes.

   ```
   index cache size * .5 = index cache-related overhead
   ```

2. Calculate additional cache overhead in bytes using the following formula:

   ```
   ((# of server threads allocated to the Essbase server process
   * 3) * 256) + 5242880 bytes = additional cache overhead
   ```

3. Sum the index cache overhead plus the additional cache overhead. Write the result to the cell labeled MC in Table 100 on page 1484.

   ```
   cache-related overhead = index cache-related overhead +
   additional cache overhead
   ```

The Number of Cells in a Logical Block

The term logical block applies to an expanded block in memory.

➤ To determine the cell count of a logical block, multiply together all members of each dense dimension (including Dynamic Calc and Dynamic Calc And Store members but excluding Label Only and shared members).

1. Using Table 102 on page 1488 as a worksheet, enter each dense dimension and its number of members excluding Label Only and shared members. If there are more than seven dense dimensions, list the dimensions elsewhere and include all dense dimensions in the calculation.
2. Multiply the number of members of the first dense dimension (line a.) by the number of members of the second dense dimension (line b.) by the number of members of the third dense dimension (line c.), and so on, to determine the total number of cells in a logical block. Write the result to the cell labeled ME in Table 101 on page 1485.

```
a * b * c * d * e * f * g = the total number of cells
```

Table 102: Determining the Number of Cells in a Logical Block

Enter Dense Dimension Name	Number of Members	
	a.	
	b.	
	c.	
	d.	
	e.	
	f.	
	g.	

Example

Excluding Label Only and shared members, the dense dimensions in Sample Basic contain 17 (Year), 14 (Measures), and 4 (Scenario) members. The calculation for the cell count of a logical block in Sample Basic is:

```
17 * 14 * 4 = 952 cells
```

Memory Area for Data Structures

At application startup time, Essbase sets aside an area of memory based on the following factors:

- The number of members in the outline
- The number of cells in a logical block (value MH in Table 101 on page 1485)
- The number of threads on the server (value MJ in Table 101 on page 1485)

➤ To calculate the data structure area in memory:

1. Use the following formula to calculate the size in bytes:

```
Number of threads * ((Number of members in the outline * 26
bytes) + (Logical block cell count * 36 bytes))
```

2. Write the result to the cell labeled MD in Table 100 on page 1484.

Example

Assuming 20 threads for the Sample Basic database, the startup area in memory required for data structures is calculated as follows:

```
20 threads * ((79 members * 26 bytes) + (952 cells * 36 bytes))
= 726,520 bytes
```

```
726,520 bytes / 1,048,576 bytes = .7 MB
```

Estimating Additional Memory Requirements for Database Operations

In addition to startup memory requirements, operations such as queries and calculations require additional memory. Because of many variables, the only way to estimate memory requirements of operations is to run sample operations and monitor the amount of memory used during these operations.

Estimating Additional Memory Requirements for Data Retrievals

Essbase processes requests for database information (queries) from a variety of sources. For example, Essbase processes queries from the Spreadsheet Add-in and from Report Writer. Essbase uses additional memory when it retrieves the data for these queries, especially when Essbase must perform dynamic calculations to retrieve the data. This section describes Essbase memory requirements for query processing.

Essbase is a multithreaded application in which queries get assigned to threads. Threads are automatically created when Essbase is started. In general, a thread exists until you shut down OLAP Server (for more information, see Chapter 41, "Running Essbase Servers, Applications, and Databases").

As Essbase processes queries, it cycles through the available threads. For example, assume 20 threads are available at startup. As each query is processed, Essbase assigns each succeeding query to the next sequential thread. After it has assigned the 20th thread, Essbase cycles back to the beginning, assigning the 21st query to the first thread.

While processing a query, a thread allocates some memory, and then releases most of it when the query is completed. Some of the memory is released to the operating system and some of it is released to the dynamic calculator cache for the database being used. However, the thread holds on to a portion of the memory for possible use in processing subsequent queries. As a result, after a thread has processed its first query, the memory held by the thread is greater than it was when Essbase first started.

Essbase uses the maximum amount of memory for query processing when both of these conditions are true:

- The maximum number of simultaneous queries that will occur are being processed.
- All threads have been assigned to at least one query by Essbase.

In the example where 20 threads are available at startup, the maximum amount of memory is used for queries when at least 20 queries have been processed and the maximum number of simultaneous queries are in process.

Calculating the Maximum Amount of Additional Memory Required

➤ To estimate query memory requirements by observing actual queries:

1. Observe the memory used during queries.
2. Calculate the maximum possible use of memory for query processing by adding together the memory used by queries that will be run simultaneously, then add the extra memory that had been acquired by threads that are now waiting for queries.

Use the following variables when you calculate the formula in "Estimating the Maximum Memory Usage for A Query Before and After Processing" on page 1492:

- Total number of threads (*Total#Threads*)
- Maximum memory usage for any query *during* processing (*MAXAdditionalMemDuringP*)
- Maximum number of possible concurrent queries (*Max#ConcQueries*)
- Maximum memory usage for any query *after* processing (*MAXAdditionalMemAfterP*)

Determining the Total Number of Threads

The potential number of threads available is based on the number of licensed ports that are purchased. The actual number of threads available depends on settings you define for the Agent or the server. Use the number of threads on the system as the value for *Total#Threads* in later calculations.

Estimating the Maximum Number of Concurrent Queries

Determine the maximum number of concurrent queries and use this value for *Max#ConcQueries* in later calculations. This value cannot exceed the value for *Total#Threads.*

Estimating the Maximum Memory Usage for A Query Before and After Processing

The memory usage of individual queries depends on the size of each query and the number of data blocks that Essbase needs to access to process each query. To estimate the memory usage, calculate the additional memory Essbase uses during processing and after processing each query.

Decide on several queries that you expect to use the most memory. Consider queries that must process large numbers of members; for example, queries that perform range or rank processing.

➤ To estimate the memory usage of a query:

1. Turn the dynamic calculator cache off by setting the ESSBASE.CFG setting DYNCALCACHEMAXSIZE to 0 (zero). Turning off the dynamic calculator cache enables measurement of memory still held by a thread by ensuring that after the query is complete, the memory used for blocks during dynamic calculations is released by the ESSSVR process to the operating system. For more information, see the *Technical Reference* in the `docs` directory.
2. Start the Essbase application.
3. Using memory monitoring tools for the operating system, note the memory used by OLAP Server *before* processing the query. Use the value associated with the ESSSVR process.

   Use this value for *MemBeforeP*.
4. Run the query.
5. Using memory monitoring tools for the operating system, note the peak memory usage of OLAP Server *while* the query is processed. This value is associated with the ESSSVR process.

   Use this value for *MemDuringP*.
6. Using memory monitoring tools for the operating system, *after* the query is completed, note the memory usage of Essbase. This value is associated with the ESSSVR process.

   Use this value for *MemAfterP*.

7. Calculate the following two values:

   - Additional memory used while Essbase processes the query (*AdditionalMemDuringP*):

     AdditionalMemDuringP = MemDuringP - MemBeforeP

   - Additional memory used after Essbase has finished processing the query (*AdditionalMemAfterP*):

     AdditionalMemAfterP = MemAfterP - MemBeforeP

8. When you have completed the above calculations for all the relevant queries, compare all results to determine the following two values:

   - The maximum *AdditionalMemDuringP* used by a query: (*MAXAdditionalMemDuringP*)

   - The maximum *AdditionalMemAfterP* used by a query: (*MAXAdditionalMemAfterP*)

9. Insert the two values from step 7 into the formula in the following statement.

   The amount of additional memory required for data retrievals will not exceed:

   Max#ConcQueries * *MAXAdditionalMemDuringP* + (*Total#Threads* - *Max#ConcQueries*) * *MAXAdditionalMemAfterP*

   Write the result of this calculation, in bytes, to the cell labeled ME in Table 100 on page 1484.

Because this calculation method assumes that all of the concurrent queries are maximum-sized queries, the result may exceed your actual requirement. It is difficult to estimate the actual types of queries that will be run concurrently.

To adjust the memory used during queries, you can set values for the retrieval buffer and the retrieval sort buffer. For information, see “Setting the Retrieval Buffer Size” on page 1446 and “Setting the Retrieval Sort Buffer Size” on page 1448.

Estimating Additional Memory Requirements Without Monitoring Actual Queries

If you cannot perform this test with actual queries, you can calculate a very rough estimate for the operational requirement of a query by summarizing the following values and multiplying the sum by the maximum number of possible concurrent queries:

- The size of the retrieval buffer (see “Setting the Retrieval Buffer Size” on page 1446)
- (If sorting is involved in the query) The size of the retrieval sort buffer (see “Setting the Retrieval Sort Buffer Size” on page 1448)
- (If the query uses dynamic calculations) The amount of memory used for dynamically calculated values, which is calculated by multiplying together two factors:
    - The number of blocks specified by the SET LOCKBLOCK command
    - The number of cells in a logical block, which includes Dynamic Calc members. See value MH. in Table 101 on page 1485.

Example

To estimate the maximum memory needed for concurrent queries, assume the following values:

- The maximum number of concurrent queries is 20
- The size of the retrieval buffer is 10,240 bytes
- The size of the retrieval sort buffer is 10,240 bytes
- The size of the memory area set aside by the SET LOCKBLOCK command. If SET LOCKBLOCK specifies 100 blocks, the value for the memory area set aside by the SET LOCKBLOCK command equals:

  ```
  100 blocks * 7616 bytes = 761,600 bytes
  ```

Estimated memory for retrievals:

```
20 * (10,240 bytes + 10,240 bytes + 761,600 bytes)
= 15,641,600 bytes
```

Estimating Additional Memory Requirements for Calculations

For existing calculation scripts, you can use the memory monitoring tools provided for the operating system on the server to observe memory usage. Run the most complex calculation and take note of the memory usage both before and while running the calculation. Calculate the difference and use that figure as the additional memory requirement for the calculation script.

To understand calculation performance, see Chapter 51, "Optimizing Calculations."

If you cannot perform a test with a calculation script, you can calculate a very rough estimate for the operational requirement of a calculation by adding together the following values:

- The size of the calculator cache (see "Managing Caches to Improve Performance" on page 1403)
- The size of the outline. For calculations, Essbase uses approximately 30 additional bytes of memory per member of the database outline (see "Managing Caches to Improve Performance" on page 1403).
- The size of the memory area used by the blocks set aside by the SET LOCKBLOCK command. To calculate the memory requirement in bytes, multiply the specified number of data blocks by the logical size of the data blocks. For the logical block size, use value ME in Table 101 on page 1485.

For the total calculation requirement, summarize the amount of memory needed for all calculations that will be run simultaneously and write that total to the cell labeled MF in Table 100 on page 1484.

Note: The size and complexity of the calculation scripts affect the amount of memory required. The effects are difficult to estimate.

Estimating Total Essbase Memory Requirements

You can use Table 103 as a worksheet on which to calculate an estimate of the total memory required on the server.

Table 103: Worksheet for Total Server Memory Requirement

Component		**Memory Required, in Megabytes (MB)**
Sum of application startup memory requirements (see "Startup Memory Requirement for an Application" on page 1483)		ML:
In rows a through g below, list concurrent databases (from copies of Table 100 on page 1484) and enter their respective memory requirements (MH) in the column to the right.		
a.		MH:
b.		MH:
c.		MH:
d.		MH:
e.		MH:
f.		MH:
g.		MH:
Operating system memory requirement		MM:
Total estimated memory requirement for the server		MN:

➤ To estimate the total Essbase memory requirement on a server:

1. Make sure the total startup memory requirement for applications is recorded in the cell labeled ML, as described in section "Startup Memory Requirement for an Application" on page 1483.

1. List the largest set of databases that will run concurrently on the server. In the Memory Required column, for each database note the memory requirement estimated in the database requirements worksheet, Table 100 on page 1484.

2. Determine the operating system memory requirement and write the value in megabytes to the cell labeled MM in Table 103.

3. Total all values and write the result in the cell labeled MN.

4. Compare the value in MN with the total available random-access memory (RAM) on the server.

   If cache memory locking is enabled, the total memory requirement should not exceed two-thirds of available RAM; otherwise, system performance can be severely degraded. If cache memory locking is disabled, the total memory requirement should not exceed available RAM.

   If there is insufficient memory available, you can redefine your cache settings and recalculate the memory requirements. This can be an iterative process. For guidelines, see "Fine Tuning Cache Settings" on page 1339. In some cases, you may need to purchase additional RAM.

Appendix

D

Accessing Relational Data with Hybrid Analysis

Because relational databases can store several terabytes of data, they offer nearly unlimited scalability. Multidimensional databases are generally smaller than relational databases but offer sophisticated analytical capabilities. With Hybrid Analysis, you can integrate a relational database with an Essbase database and thereby leverage the scalability of the relational database with the conceptual power of the multidimensional database.

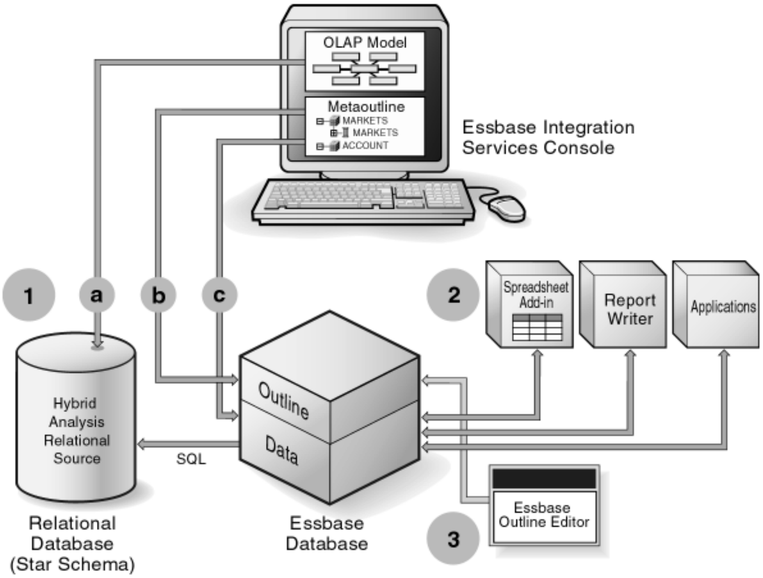
Hybrid Analysis eliminates the need to load and store lower-level members and their data within the Essbase database. This feature gives Essbase the ability to operate with almost no limitation on outline sizes and provides for rapid transfer of data between Essbase databases and relational databases.

This chapter helps you understand Hybrid Analysis and explains how you can take advantage of its capabilities. The chapter includes the following sections:

Overview of Hybrid Analysis

Hybrid Analysis integrates a relational database with an Essbase multidimensional database so that applications and reporting tools can directly retrieve data from both databases. Figure 553, below, illustrates the Hybrid Analysis architecture:

Figure 553: Hybrid Analysis Architecture



Hybrid Analysis Relational Source

The initial step in setting up Hybrid Analysis is to define the relational database as a Hybrid Analysis relational source (**1** in Figure 553).

You define the Hybrid Analysis relational source in Essbase Integration Services Console. (The individual tasks are discussed in "Defining the Hybrid Analysis Relational Source" on page 1502.) Through Integration Services Console, you first specify the relational data source for the *OLAP model*. The OLAP model is a schema that you create from tables and columns in the relational database. To build the model, Integration Services accesses the star schema of the relational database (**a** in Figure 553).

Using the model, you define hierarchies and tag level members to be Hybrid Analysis-enabled. You then build the *metaoutline*, a template containing the structure and rules for creating the outline, down to the desired Hybrid Analysis level. The information enabling Hybrid Analysis is stored in the *OLAP Metadata Catalog* which describes the nature, source, location, and type of data in the Hybrid Analysis relational source.

Next, you perform a *member load* which adds dimensions and members to the outline (**b** in Figure 553). When the member load is complete, you run a data load to populate the Essbase database with data (**c** in Figure 553). At this point, the Hybrid Analysis architecture is in place:

- The lower level members and their associated data remain in the relational database.
- The data in the relational database is mapped to the Hybrid Analysis-defined outline.
- The outline resides in the Essbase database.

  Metadata is data that describes the values within the database. The metadata that defines the Hybrid Analysis data resides in both the Essbase outline and in the Integration Services metaoutline on which the outline is based. Any changes that are made to Hybrid Analysis data in an OLAP model or metaoutline associated with an Essbase outline must be updated to the outline to ensure accuracy of the data reported in Essbase. See "Managing Data Consistency" on page 1506 for information on keeping your data and metadata in sync.
- The upper level members and their associated data reside in the Essbase database.

Data Retrieval

Applications and reporting tools, such as spreadsheets and Report Writer interfaces, can directly retrieve data from both databases (**2** in Figure 553). Using the dimension and member structure defined in the outline, Essbase determines the location of a member and then retrieves data from either the Essbase database or the Hybrid Analysis relational source. If the data resides in the Hybrid Analysis relational source, Essbase retrieves the data through SQL commands. Data retrieval is discussed in "Retrieving Hybrid Analysis Data" on page 1503.

If you want to modify the outline, you can use the Essbase Outline Editor to enable or disable dimensions for Hybrid Analysis on an as-needed basis. (**3** in Figure 553). For information on using the Outline Editor, see "Using the Outline Editor with Hybrid Analysis" on page 1505.

Hybrid Analysis Guidelines

Hybrid Analysis has some guidelines with which you should be familiar:

- Essbase requires the OLAP Metadata Catalog created in Integration Services in order to drill down in a Hybrid Analysis relational source.
- A single Essbase database can be associated with only one Hybrid Analysis relational source.
- A Hybrid Analysis relational source can consist of only one relational database.
- Only the lowest level members of a dimension can be Hybrid Analysis-enabled.
- Hybrid Analysis supports only parent-child prefixing on member names.
- Essbase does not support aliases for Hybrid Analysis-enabled members.
- Essbase does not support attributes for Hybrid Analysis-enabled members.
- Hybrid Analysis does not support the Dynamic Times Series function.
- Hybrid Analysis does not support transparent, replicated, or linked partitions.
- Hybrid Analysis does not support a metaoutline that contains an accounts dimension created from a user-defined dimension.

Defining the Hybrid Analysis Relational Source

The Hybrid Analysis relational source is defined in Essbase Integration Services Console. Detailed information and the specific procedures for performing the following steps is available in Integration Services online help.

➤ To define the Hybrid Analysis relational source, perform the following steps:

1. Specify the relational data source.

   The OLAP model is created from tables and columns in the relational database. To build the model, Integration Services accesses the star schema of the relational database.

2. In the OLAP model, define the hierarchies that you will use for Hybrid Analysis.

   Note: You can define any member of any dimension as Hybrid Analysis-enabled except members in an accounts dimension. This restriction is necessary because all members of an accounts dimension, including lower-level members, must remain in the Essbase database.

3. In the metaoutline, use the Build Multidimensional Down to Here command to select the level members that will reside in the Essbase multidimensional database.
4. Use the Enable Hybrid Analysis Down to Here command to select the member levels that will remain in the relational database.
5. Run a member load to add dimensions and members to the outline.
6. Run a data load to populate the Essbase database with data.

Retrieving Hybrid Analysis Data

In Hybrid Analysis, applications and reporting tools can directly retrieve data from both the relational and Essbase databases by using the following tools:

- Essbase Spreadsheet Add-in
- Report Writer
- Hyperion Analyzer
- Third-party applications

Note: The Essbase database and the relational database must be registered to the same ODBC, and Integration Services must use the same source name for both databases.

Because data is being accessed from both the Hybrid Analysis relational source and the Essbase database when you perform calculations or generate reports, data retrieval time may increase with Hybrid Analysis; however, all capabilities of Essbase data retrieval operations are available with Hybrid Analysis, including pivot, drill-through, and other metadata-based methods.

Retrieving Hybrid Analysis Data with Spreadsheet Add-in

Use the Span Hybrid Analysis option in the Essbase Options dialog box in the Essbase Spreadsheet Add-in to drill down to members in the Hybrid Analysis relational source. The following grid Essbase API functions can be used to enable or disable access to the Hybrid Analysis relational storage:

- ESSSETSHEETOPTION
- ESSGETSHEETOPTION

Refer to the *Essbase Spreadsheet Add-in User's Guide* and to Spreadsheet Add-in online help for more information.

Supported Drill-Down Options in Hybrid Analysis

Hybrid Analysis supports the following drill-down options in Spreadsheet Add-in:

- Next Level (children)
- All Levels (all descendants)
- Bottom Level (level 0)
- All Siblings (all members with common parent)

Supported Drill-Up Option in Hybrid Analysis

Hybrid Analysis supports the following drill-up option in Spreadsheet Add-in: Parent

Retrieving Hybrid Analysis Data with Report Writer

In Report Writer, two commands enable/disable Hybrid Analysis:

- <HYBRIDANALYSISON enables a report script to retrieve the members of a dimension that is Hybrid Analysis-enabled.
- <HYBRIDANALYSISOFF prevents a report script from retrieving the members of a dimension that is Hybrid Analysis-enabled.

The <ASYM and <SYM commands are not supported with Hybrid Analysis. If these commands are present in a report, errors may result. The <SPARSE command is ignored in reports retrieving data from a Hybrid Analysis relational source and does not generate errors. Refer to the *Technical Reference* in the `docs` directory for more information.

The following is a sample Report Writer script which uses the IDESCENDANTS command to return Hybrid Analysis data:

```
<PAGE (Accounts, Scenario, Market)
Sales
Actual

<Column (Time)
<CHILDREN Time

<Row (Product)
<IDESCENDANTS 100-10

!
```

Retrieving Hybrid Analysis Data with Hyperion Analyzer

When you use Hyperion Analyzer, the procedures for retrieving Hybrid Analysis data are the same as the procedures for retrieving data that is not defined for Hybrid Analysis. See the Hyperion Analyzer documentation for detailed information.

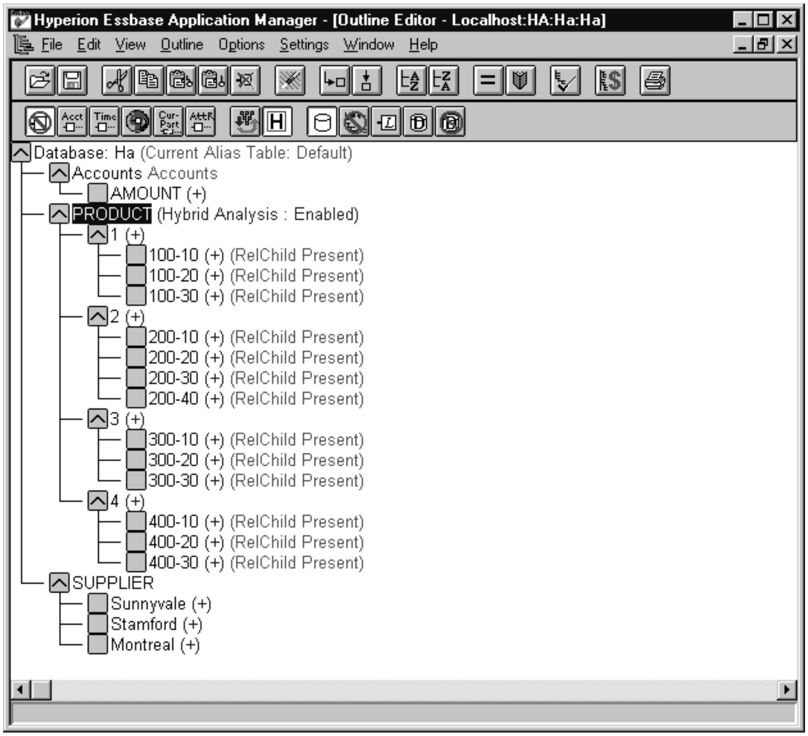
Using the Outline Editor with Hybrid Analysis

In the Outline Editor, you can toggle the Hybrid Analysis option H button to enable or disable Hybrid Analysis for each dimension that has been Hybrid Analysis-defined in Integration Services Console. If you open an outline that is not Hybrid Analysis-defined, the Hybrid Analysis option button is not displayed on the toolbar.

Note: When Hybrid Analysis is disabled for a dimension, the end user is unable to see and drill-through to the Hybrid Analysis data associated with the dimension; however, the members are still visible in the Outline Editor.

Figure 554, below, is an example of how a Hybrid Analysis-defined outline appears in the Outline Editor. Note that Hybrid Analysis-enabled dimensions are identified to distinguish them from dimensions that are not Hybrid Analysis enabled.

Figure 554: Example of Hybrid Analysis in Outline Editor


Hyperion Essbase Application Manager - [Outline Editor - Localhost:HA:Ha:Ha]
File
Edit
View
Outline
Options
Settings
Window
Help

```
Database: Ha (Current Alias Table: Default)
  Accounts Accounts
    AMOUNT (+)
  PRODUCT (Hybrid Analysis : Enabled)
    1 (+)
      100-10 (+) (RelChild Present)
      100-20 (+) (RelChild Present)
      100-30 (+) (RelChild Present)
    2 (+)
      200-10 (+) (RelChild Present)
      200-20 (+) (RelChild Present)
      200-30 (+) (RelChild Present)
      200-40 (+) (RelChild Present)
    3 (+)
      300-10 (+) (RelChild Present)
      300-20 (+) (RelChild Present)
      300-30 (+) (RelChild Present)
    4 (+)
      400-10 (+) (RelChild Present)
      400-20 (+) (RelChild Present)
      400-30 (+) (RelChild Present)
  SUPPLIER
    Sunnyvale (+)
    Stamford (+)
    Montreal (+)
```

Managing Data Consistency

When you create a Hybrid Analysis relational source, your data and metadata are stored and managed in the relational database and the Essbase database:

- The lower-level members and their associated data remain in the relational database.
- The data in the relational database is mapped to the outline that is Hybrid Analysis defined.

- The outline resides in the Essbase database.
- The upper-level members and their associated data reside in the Essbase database.

Because the data and metadata exist in different locations, information may become out of sync.

Essbase depends upon the OLAP Metadata Catalog in Integration Services to access the Hybrid Analysis relational source. At Essbase database startup time, Essbase OLAP Server checks the number of dimensions and members of the Essbase outline with the related metaoutline. Any changes made to the associated OLAP model or metaoutline during an Integration Services session are not detected by the OLAP server until the Essbase database is started again. Undetected changes can cause data inconsistency between the Essbase database and the Hybrid Analysis relational source.

If changes are made in the Hybrid Analysis relational source and members are added or deleted in an OLAP model or metaoutline, such changes can cause the Essbase outline to be out of sync with the metaoutline on which it it is based. These types of changes and their effect on the hierarchical structure of a dimension are not reflected in the Essbase database until the outline build and data load process is completed through Integration Services Console.

In Application Manager, the Restructure Database dialog box has a check box that enables a warning whenever a restructuring affects an outline containing a Hybrid Analysis relational source. Such a problem occurs, for example, if members with relational children are moved or deleted. Warnings are listed in the application log file. You should decide if these warnings reflect a threat to your data consistency.

The Essbase administrator has the responsibility to ensure that the Essbase multidimensional database, the relational database, and the Integration Services OLAP model and metaoutline remain in sync. Both Application Manager and Integration Services Console provide commands that enable the administrator to perform consistency checks and make the appropriate updates.

For additional information on maintaining data consistency, see Chapter 40, “Ensuring Data Integrity.”

For additional information on restructuring your database, see Chapter 49, “Optimizing Database Restructuring.”

Managing Security in Hybrid Analysis

The Essbase administrator determines access to the Hybrid Analysis relational source on an individual Essbase user level. Access for Hybrid Analysis is governed by the same factors that affect overall Essbase security:

- Privileges and access levels for individual users and groups of users
- Privileges and access levels for the server, application, or database
- Specific database access levels for particular database members

If a security filter allows you to view only the relational children of the level 0 members that you have access to in Essbase, then you cannot view the relational children of the level 0 members that you do not have access to in Essbase.

Assume that you have the following outline, where San Francisco and San Jose are relational children of California, and Miami and Orlando are relational children of Florida:

```
Market
  West
    California
      San Francisco
      San Jose
  East
    Florida
      Miami
      Orlando
```

In this example, if a filter allows you to view only level 0 member California and its descendants, you can view California and its relational children, San Francisco and San Jose; however, you cannot view the children of level 0 member Florida.

For detailed information on Essbase security, see the following chapters:

- Chapter 15, “Managing Security for Users and Applications”
- Chapter 16, “Controlling Access to Database Cells”
- Chapter 17, “Security Examples”

Using Formulas with Hybrid Analysis

Formulas used with Hybrid Analysis-enabled members are subject to the following limitations:

- Formulas are supported only on a measures dimension.
- Formulas cannot be attached to relational members.
- Formulas cannot reference a relational member by name.
- Member set functions (such as @CHILDREN and @DESCENDANTS), which generate member lists in a formula, execute only in the Essbase portion of the outline.

Unsupported Functions in Hybrid Analysis

Hybrid Analysis does not support all Essbase functions.

Generating Member Lists

Hybrid Analysis does not support the following functions used to generate member lists:

- @ANCEST
- @CURRMBR
- @PARENT

Specifying Member Conditions

Hybrid Analysis does not support the following functions used to specify member conditions:

- @ISIANCEST
- @ISIPARENT
- @ISISIBLING
- @ISLEV
- @ISSAMEGEN
- @ISUDA

Looking up Values

Hybrid Analysis does not support the following functions used to set a range for members:

- @ATTRIBUTESVAL
- @XREF

Range Function

Hybrid Analysis does not support the following function used to look up specific values of members: @MDSHIFT.

Current Member

Hybrid Analysis does not support the following function used to determine whether the current member is the member being specified: @ISMBR.

Glossary

accounts dimension. A dimension type that makes accounting intelligence available. You can tag only one dimension as Accounts; you do not have to have an accounts dimension.

administrator. An individual who installs and maintains the Essbase system, including setting up user accounts and security. *See also* database administrator (DBA) and system administrator.

Advanced Interpretation mode. An option in Essbase Spreadsheet Add-in that you use to define a layout through drill through or Essbase Query Designer or by typing data into the sheet. When you construct a free-form report in Advanced Interpretation mode, Essbase interprets the member names and creates a default view that is based on the location of the labels.

agent. A process on the server that starts and stops applications and databases, manages connections from users, and handles user-access security. The agent is referred to as ESSBASE.EXE.

aggregate. *See* consolidate.

alias. An alternative name for a dimension, member, or description.

alias table. A database table that stores aliases for the dimensions or members.

ancestor. A branch member that has members below it. For example, in a dimension that includes years, quarters, and months, the members Qtr2 and 2001 are ancestors of the member April.

application. A management structure containing one or more Essbase databases and the related files that control many system variables, such as memory allocation and autoload parameters.

application designer. An individual who designs, creates, and maintains Essbase applications and databases.

application log. A record of user actions performed on an application.

Application Manager. Essbase software that you use to create and maintain Essbase applications.

Application Programming Interface (API). A library of functions that you can use in a custom program. Provides programmatic access to an application's data or services. Hyperion Application Builder provides a Java API that you can use to develop client programs.

area. A predefined set of members and values that makes up a partition.

arithmetic data load. A data load that performs operations on values in the database, such as adding 10 to each value.

asymmetric report. A report characterized by groups of members that differ by at least one member across the groups. There can be a difference in the number of members or the names of members under each heading in the report. For example, a report based on Sample Basic can have three members grouped under "East" and two members grouped under "West."

attribute. A classification of a member in a dimension. You can select and group members based on their associated attributes. You can also specify an attribute when you perform calculations and use calculation functions. For example, a Product dimension can have several attributes, such as Size and Flavor. A specific member of the Product dimension can have the Size attribute, 8, and the Flavor attribute, Cola.

attribute association. A relationship in a database outline whereby a member in an attribute dimension describes a characteristic of a member of its base dimension. For example, if product 100-10 has a grape flavor, the product 100-10 has the Flavor attribute association of grape. Thus, the 100-10 member of the Product dimension is associated with the Grape member of the Flavor attribute dimension.

Attribute Calculations dimension. A system-defined dimension that performs the following calculation operations on groups of members: Sum, Count, Avg, Min, and Max. The calculation is based on the attributes associated with the members. This dimension is calculated dynamically and is not visible in the database outline. For example, by using the Avg member, you can calculate the average sales value for Red products in New York in January.

attribute dimension. A type of dimension that enables analysis based on the attributes or qualities of the members of its base dimension.

attribute reporting. A process of defining reports that is based on the attributes of the base members in the database outline.

attribute type. A text, numeric, Boolean, or date type that enables different functions for grouping, selecting, or calculating data. Although assigned at the dimension level, the attribute type applies only to level 0 members of the attribute dimension. For example, because the Ounces attribute dimension has the type numeric, you can use the number of ounces that is specified as the attribute of each product to calculate the profit per ounce for that product.

bang character (!). A character that terminates a series of report commands and requests information from the database. A report script must be terminated with a bang character; several bang characters may be used within a report script.

base currency. The currency in which daily business transactions are performed.

base dimension. A standard dimension that is associated with one or more attribute dimensions. To classify a member of a base dimension, you associate it with a member of one or more attribute dimensions that describes the classification, such as a specific flavor. For example, assuming products have flavors, the Product dimension is the base dimension for the Flavors attribute dimension.

batch calculation. Any calculation on a database that is done in batch; for example, a calculation script or a full database calculation. Dynamic calculations are not considered to be batch calculations.

batch file. An operating system file that can call multiple ESSCMD scripts and run multiple sessions of ESSCMD. Batch files handle batch data loads and complex calculations and can include commands that run report scripts. You can run a batch file on the server from the operating system prompt. On Windows-based systems, batch files have .BAT file extensions. On UNIX, a batch file is written as a shell script.

batch processing mode. A method of using ESSCMD to write a batch or script file that can be used to automate routine server maintenance and diagnostic tasks. ESSCMD script files can execute multiple commands and can be run from the operating system command line or from within operating system batch files. Batch files can be used to call multiple ESSCMD scripts or run multiple instances of ESSCMD.

block. The primary storage unit within Essbase. A block is a multidimensional array representing the cells of all dense dimensions.

build method. A method used to modify database outlines. You choose a build method based on the format of data in data source files.

cache. A buffer in memory that holds data temporarily.

cache memory locking. An Essbase database setting that, when enabled, locks the memory used for the index cache, data file cache, and data cache into physical memory, potentially improving database performance. This setting is disabled by default.

calculation. The process of aggregating or of running a calculation script on a database.

calculation script. A set of commands that define how a database is consolidated or aggregated. A calculation script may also contain commands that specify allocation and other calculation rules separate from the consolidation process.

cascade. The process of creating multiple reports for a subset of member values.

cell. A unit of data representing the intersection of dimensions in a multidimensional database; the intersection of a row and a column in a worksheet.

cell note. A text annotation of up to 599 characters for a cell in an Essbase database. Cell notes are a type of linked reporting object.

change log. *See* outline change log.

child. A member that has a parent above it in the database outline.

clean block. A data block is marked as clean if the database is fully calculated, if a calculation script calculates all dimensions at once, or if the SET CLEARUPDATESTATUS command is used in a calculation script.

client. A client interface, such as the Essbase Spreadsheet Add-in software, a custom API program, or Essbase Application Manager. A client is also a workstation that is connected to a server through a local area network.

client log. A record of all messages, actions, and errors that are generated by a client.

column. A vertical display of information in a grid or table. A column can contain data from a single field, derived data from a calculation, or textual information. The terms column and field are sometimes used interchangeably. *Contrast with* row.

column heading. A part of a report that lists members across a page. When you define columns that report on data from more than one dimension, you produce nested column headings. A member that is listed in a column heading is an attribute of all data values in its column.

committed access. A Essbase Kernel Isolation Level setting that affects how Essbase handles transactions. Under committed access, concurrent transactions hold long-term write locks and yield predictable results.

consolidate. The process of gathering data from dependent entities and aggregating the data up to parent entities. After you enter or load data into dependent child entities, you perform a consolidation to aggregate the data through the organization. As data consolidates, intercompany processing, conversion methods, equity adjustments, and minority ownerships perform calculations on the data. For example, if the dimension Year consists of the members Qtr1, Qtr2, Qtr3, and Qtr4, its consolidation is Year. The terms aggregate and roll-up also describe the consolidation process.

crosstab reporting. A type of reporting that categorizes and summarizes data in a table format. The cells within the table contain summaries of the data that fit within the intersecting categories. For example, a crosstab report of product sales information could show size attributes, such as Small and Large, as column headings and color attributes, such as Blue and Yellow, as row headings. The cell in the table where Large and Blue intersect could contain the total sales of all Blue products that are sized Large.

currency. The monetary unit of measure associated with a balance or transaction.

currency conversion. A process that converts currency values in a database from one currency into another currency. For example, to convert one US dollar into the euro, the exchange rate of 0.923702 is multiplied with the dollar (1* 0.923702). After conversion, the euro amount is .92.

currency partition. A dimension type that separates local currency members for a base currency, as defined in an application. A currency partition identifies currency types, such as Actual, Budget, and Forecast.

currency symbol. A character that represents a currency. For example, the currency symbol for the US dollar is $ and the currency symbol for the British pound is £.

custom-defined function (CDF). Essbase calculation functions that you develop in the Java programming language and add to the standard Essbase calculation scripting language by means of MaxL. *See also* custom-defined macro (CDM).

custom-defined macro (CDM). Essbase macros that you write with Essbase calculator functions and special macro functions. Custom-defined macros use an internal Essbase macro language that enables you to combine calculation functions and operate on multiple input parameters. *See also* custom-defined function (CDF).

data block. *See* block.

data cache. A buffer in memory that holds uncompressed data blocks.

data cell. *See* cell.

data file. A file containing data blocks; Essbase generates the data file during a data load and stores it on disk.

data file cache. A buffer in memory that holds compressed data (.PAG) files.

data load. The process of populating an Essbase database with data. Loading data establishes actual values for the cells defined by the structural outline of the database.

data value. *See* cell.

database. A repository of data within Essbase that contains a multidimensional data storage array. Each database consists of a storage structure definition (a database outline), data, security definitions, and optional calculation scripts, report scripts, and data loading scripts. An application contains one or more databases.

database administrator (DBA). An individual who administers database servers, such as Essbase, and who may also design, maintain, and create databases.

database designer. In Essbase, the highest type of access that can be assigned globally (per database). This type of access allows complete calculate and update access and the ability to run report and calculation scripts.

database filter layer. A layer in the Essbase security plan that defines specific settings for database members down to the cell level.

dense dimension. A dimension with a high probability that data exists for every combination of dimension members.

descendant. Any member below a parent in the database outline. For example, in a dimension that includes years, quarters, and months, the members Qtr2 and April are descendants of the member Year.

dimension. A data category that is used to organize business data for retrieval and preservation of values. Each dimension usually contains a hierarchy of related members grouped within it. For example, a Year dimension often includes members for each time period, such as quarters and months. Other common business dimensions may be measures, natural accounts, products, and markets.

dimension build rules. Specifications, similar to data load rules, that Essbase uses to modify an outline. The modification is based on data in an external data source file.

dimension type. A dimension property that enables the use of predefined functionality. Dimensions that are tagged as Time have a predefined calendar functionality.

dirty block. A data block containing cells that have been changed since the last calculation. Upper level blocks are marked as dirty if their child blocks are dirty (that is, have been updated).

disabled user name. A user name that has become inactive, meaning that the user is not able to log on to the server. Users with supervisor privilege can disable a user name for any reason. User names are disabled automatically if they exceed server-specific limitations on login attempts or the number of inactive days. Only users with supervisor privilege can enable disabled user names.

drill down. The process of retrieving progressively detailed data relative to a selected dimension by expanding a parent member to reveal its children. The expansion can reveal hierarchical relationships, such as those between a parent entity and its child entity, a parent account and a child account, and a summary time period and a base time period. For example, drilling down can reveal the hierarchical relationships between year and quarters or between quarter and months.

dynamic reference. A pointer in the rules file to header records in a data source. Header records define data load or dimension build criteria for the fields in a data source.

Dynamic Time Series. A process that is used to perform dynamic period-to-date reporting for all values associated with a query.

Essbase kernel. A layer of the Essbase server that provides the foundation for a variety of functionality, including data loading, calculations, spreadsheet lock&send, partitioning, and restructuring. The Essbase kernel reads, caches, and writes data; it manages transactions; and it enforces transaction semantics to ensure data consistency and data integrity.

essbase.cfg. The name of an optional configuration file for Essbase. There is a file for the server and one for the client. Administrators may enter parameters and values in this file to customize Essbase server or client settings.

EssCell. The Essbase cell retrieve function. An EssCell function is entered into a cell in Essbase Spreadsheet Add-in to retrieve a single database value that represents an intersection of specific database members.

ESSCMD. A command-line interface that is used to perform server operations interactively or through a batch file.

ESSCMD script file. A text file that contains ESSCMD commands, which Essbase executes in order to the end of the file. You can run a script file from the operating system command line or from within an operating system batch file. The default extension is .SCR.

essmsh. *See* MaxL Shell.

extraction command. A type of reporting command that handles the selection, orientation, grouping, and ordering of raw data extracted from a database. These commands begin with the less than (<) character.

field. A value or item in a data source file that will be loaded into an Essbase database.

file delimiter. One or more characters, such as commas or tabs, that separate fields in a data source.

filter. A method for controlling access to database cells in Essbase. A filter is the most detailed level of security, allowing you to define varying access levels different users can have to individual database values.

FlashBack. An Essbase Spreadsheet Add-in command that restores the previous database view. This command is similar to a typical Undo command.

formula. A combination of operators and functions as well as dimension names, member names, and numeric constants. Formulas are used to perform specific calculations on members of a database.

Free-Form mode. An option in Essbase Spreadsheet Add-in that you use to type report script commands in the worksheet to create reports.

free-form reporting. A method of creating reports in which you type members of dimensions or report script commands in a worksheet. Free-form reporting is available in both Advanced Interpretation mode and Free-Form mode.

function. A predefined routine that returns a value, a range of values, a Boolean value, or a list of database members. The system provides the following categories of functions: mathematical, relationship, financial, member set, Boolean, statistical, forecasting, allocation, and date-time.

generation. A layer in a hierarchical tree structure that defines member relationships in a database. For example, Essbase orders generations incrementally from the dimension (generation 1) down to the child members.

generation name. A unique name that describes a generation.

global access layer. A layer in the Essbase security system that is used to define common access settings for applications and databases.

global report command. A command that is executed when it occurs in the report script file and that stays in effect until the end of the report file or until another global command replaces it.

grouping. A set of members that is selected by a filtering process and that may be treated as a separate aggregate group. This group behaves very much like a parent to all of its specific members, and it supports full calculation logic, including additive and non-additive calculations. For example, you can use the attribute Small to view and work with all members with the attribute Small.

header record. One or more records at the top of a data source. Header records describe the contents of the data source.

hierarchy. A set of multidimensional relationships in an outline, often created in a tree formation. For example, parents, children, and generations represent a hierarchy.

Hybrid Analysis. The integration of a relational database with an Essbase multidimensional database so that lower-level data remains in the relational database and is mapped to higher-level data residing in the Essbase database. By eliminating the necessity of loading and storing lower-level data in the Essbase database, Hybrid Analysis allows Essbase to take advantage of the mass scalability of the relational database.

index. A method that Essbase uses to retrieve data. The retrieval is based on the combinations of sparse dimensions. The term index also refers to the index file.

index cache. A buffer in memory that holds index pages.

index entry. A pointer to an intersection of sparse dimensions. Each index entry points to a data block on disk and locates a particular cell within the block by means of an offset.

index file. A file that Essbase uses to store data retrieval information. It resides on disk and contains index pages.

index page. A subdivision of an index file containing entries that point to data blocks.

input block. A type of data block that has at least one loaded data value.

input data. Any data that is loaded from a data source and is not generated by calculating the database.

intelligent calculation. A calculation method that tracks which data blocks have been updated since the last calculation.

interactive mode. A method of using ESSCMD by entering commands in the ESSCMD window and responding to prompts if necessary. For routine server administration tasks, or for complex tasks that require many commands, consider using batch processing mode.

interdimensional irrelevance. A situation in which a specific dimension does not intersect with other dimensions. The data is not irrelevant, but because the data in the specific dimension cannot be accessed from the other dimensions, those other dimensions are not relevant to the specific dimension.

isolation level. An Essbase kernel setting that determines the lock and commit behavior of database operations. Choices are committed access and uncommitted access.

latest. A key word that is used within Essbase Spreadsheet Add-in or within Report Writer to extract data values based on the member defined as the latest period of time.

leaf member. A member that has no children.

level. A branch within a dimension. The levels are numbered incrementally from the leaf member (level 0) towards the root.

level 0 block. A data block that is created for sparse member combinations when all of the members of the sparse combination are level 0 members.

level 0 member. *See* leaf member.

level name. A unique name that describes a level.

linked object. A term that encompasses linked partitions and linked reporting objects.

linked partition. A form of shared partition that provides the ability to use a data cell to link together two different databases. When a user clicks on a linked cell in a worksheet, for example, Essbase opens a new sheet displaying the dimensions in the second database. The user can then drill down into the available dimensions in the second database.

linked reporting object (LRO). An external file that is linked to a data cell in an Essbase database. Linked reporting objects (LROs) can be cell notes, URLs, or files that contain text, audio, video, or pictures.

location alias. A location alias is a descriptor that identifies a data source. The location alias specifies a server, application, database, username, and password. Location aliases are set by the database administrator at the database level using Application Manager, ESSCMD, or the API.

log. A system-maintained record of transactional data resulting from actions and commands.

log delimiter. A character inserted between fields of a log file to allow a program to parse and manipulate log file information.

mathematical operator. A symbol that defines how data is calculated. A mathematical operator can be any of the standard mathematical or Boolean operators; for example, +, -, *, /, and %. Mathematical operators are used in formulas, abd outlines.

MaxL. The multi-dimensional database access language for Essbase. Using statements, the MaxL language enables you to perform batch or interactive system-administrative tasks on the Essbase system. *See also* MaxL Shell.

MaxL Perl Module. A Perl module (essbase.pm) that is part of the MaxL component of Essbase. You can add essbase.pm to your Perl package to provide access to Essbase databases from Perl programs. Communication from Perl to MaxL to Essbase combines the system-administrative functionality of the Essbase MaxL language with the rich programmatic control of Perl.

MaxL Script Editor. A script-development environment that is part of the Essbase Administration Console interface. The MaxL Script Editor is an integrated alternative to using a text editor and the MaxL Shell for creating, opening, editing, and running MaxL scripts for Essbase system administration.

MaxL Shell. An interface for passing MaxL statements to the Essbase OLAP server. The MaxLShell executable file, located in the bin directory for Essbase, is named essmsh (UNIX) or essmsh.exe (Windows).

member. A discrete component within a dimension. For example, a time dimension might include such members as Jan, Feb, and Qtr1.

member filtering (member selection). The process of selecting specific members that will be used in a query. You can apply selection criteria, such as generation names, level names, pattern match, attributes, and UDAs.

member load. In Essbase Integration Services, the process of adding new dimensions and members (without data) to a Hyperion Essbase outline. *Contrast with* data load.

member select. A feature within Essbase Spreadsheet Add-in that you use to specify members for a report.

member selection report command. A type of Report Writer command that selects ranges of members based on database outline relationships, such as sibling, generation, and level.

member-specific report command. A type of Report Writer formatting command that is executed as it is encountered in a report script. The command affects only the member to which it is associated and executes the format command before it processes the member.

metaoutline. In Essbase Integration Services, a template containing the structure and rules for creating a Hyperion Essbase outline from an OLAP model.

Minimum Database Access. An option group that controls the default security to all of the databases of an application, using access settings (such as Read or None) that are applied globally to the application. All users connecting to databases within the application have the access level defined as minimum database access; however, individual user privileges may be higher.

missing data (#MISSING). A marker indicating that data in the labeled location does not exist, contains no meaningful value, or was never entered or loaded. For example, missing data exists when an account contains data for a previous or a future period but not for the current period.

multidimensional database (MDDB). A method of organizing, storing, and referencing data through three or more dimensions. An individual value is the intersection of a point for a set of dimensions.

multithreading. Within a single program, concurrent handling of multiple, separately executable sequences of program instructions.

Named Pipes. A network protocol stack that enables Essbase clients on Windows 95/98 to communicate with Essbase servers on Windows NT (when both operating systems use NetBEUI instead of TCP/IP).

Navigate Without Data. An Essbase Spreadsheet Add-in option that you use to turn off data retrieval. This feature is most useful when a database has Dynamic Calc and Dynamic Calc And Store members.

nested column headings. A column heading format for report columns that displays data from more than one dimension. For example, in the Sample Basic database, a column heading that contains both Year and Scenario members is a nested column. This is scripted as follows: <COLUMN (Year, Scenario). The nested column heading shows Q1 (from the Year dimension) in the top line of the heading, qualified by Actual and Budget (from the Scenario dimension) in the bottom line of the heading.

numeric attribute range. A feature that you can use to associate a base dimension member that has a discrete numeric value with an attribute that represents a range of values. For example, to classify your customers by age, you can define an Age Group attribute dimension that contains members for the following age ranges: 0-20, 21-40, 41-60, and 61-80. You can associate each member of the Customer dimension with a particular Age Group range. You can then retrieve data based on the age ranges rather than based on individual age values.

object. A program component that is related to an application or database. Objects can be outlines, rules files, calculation scripts, report scripts, or data sources. They are stored within the application or database subdirectory on the server or client machine.

OLAP Metadata Catalog. In Essbase Integration Services, a relational database containing metadata describing the nature, source, location, and type of data that you pull from the relational data source. Essbase Integration Server accesses the OLAP Metadata Catalog to generate the SQL statements and the information required to generate a Hyperion Essbase database outline.

OLAP model. In Essbase Integration Services, a logical model (star schema) that you create from tables and columns in a relational database. You can then use the OLAP model to generate the structure of a multidimensional database.

OLAP Server log. A record of actions performed by the OLAP Server (agent).

online analytical processing (OLAP). A multidimensional, multi-user, client-server computing environment for users who need to analyze consolidated enterprise data. OLAP systems feature functionality such as drilling down, data pivoting, complex calculations, trend analyses, and modeling.

outline. The database structure of a multidimensional database, including all dimensions, members, tags, types, consolidations, and mathematical relationships. Data is stored in the database according to the structure defined in the outline.

outline change log. A record of changes made to an Essbase database outline.

page file. *See* data file.

page heading. A type of report heading that lists members that are represented on the current page of the report. All data values on the page have the members in the page heading as a common attribute. Note: when printed, a report page may occupy more than one piece of paper.

paging. A storage scheme that makes use of spare disk space to increase the available memory.

parallel calculation. An optional calculation setting. Essbase divides a calculation into tasks and calculates some of the tasks at the same time.

parallel data load. In Essbase, the concurrent execution of different stages of a single data load by multiple process threads.

parallel export. The ability to export Essbase data to multiple files. This may be faster than exporting to a single file, and it may resolve problems caused by a single data file becoming too large for the operating system to handle.

parent. A member that has an aggregated branch of children below it.

partition area. A subcube within a database. A partition is composed of one or more areas. These areas are composed of cells from a particular portion of the database. For replicated and transparent partitions, the number of cells within an area must be the same for both the data source and the data target to ensure that the two partitions have the same shape. If the data source area contains 18 cells, the data target area must also contain 18 cells to accommodate the number of values.

Partition Manager. An Essbase tool that you use to create and maintain a replicated, linked, or transparent database partition. Partition Manager includes Partition Wizard, a component that contains a series of pages that step you through the partition creation process.

partitioning. The process of defining areas of data that are shared or linked between data models. Partitioning can affect the performance and scalability of Essbase applications.

Password Management. A group of options in the server settings that you use to limit a user's allowed number of login attempts, number of days of inactivity, and number of days using the same password.

pattern matching. The ability to match a value with any or all characters of an item that is entered as a criterion. A missing character may be represented by a wild card value such as a question mark (?) or an asterisk (*). For example, “Find all instances of apple” returns apple, but “Find all instances of apple*” returns apple, applesauce, applecranberry, and so on.

period. An interval within the time dimension.

permission. A level of access users and groups can have for managing data or other users and groups. Permissions can be granted to users and groups explicitly or by means of filters. Administrators can also set global minimum permissions as settings for Essbase applications and databases, so that all users can have at least access to data specified by the minimum permission setting for a particular scope. The scope of a permission refers to the area of data it encompasses. The scope of a permission can be the system, an application, or a database.

persistence. The continuance or longevity of effect for any Essbase operation or setting. For example, an Essbase administrator may limit the persistence of user-name and password validity.

Personal Essbase. A version of the Essbase OLAP Server that is designed to run on one computer.

pivot. The ability to alter the perspective of retrieved data. When Essbase first retrieves a dimension, it expands data into rows. You can then pivot or rearrange the data to obtain a different viewpoint.

precalculation. The process of calculating the database prior to user retrieval.

preserve formulas. The process of keeping user-created formulas within a worksheet while retrieving new data.

property. A characteristic of a member, such as two-pass calculation or shared member. Properties affect how Essbase works with the data.

query governor. An Essbase Integration Server parameter or Essbase OLAP Server configuration setting that you set to control the duration and size of the queries made to the data source.

record. In a database, a group of fields that make up one complete entry. For example, a record about a customer might contain fields for name, address, telephone number, and sales data.

redundant data. Duplicate data blocks that Essbase retains during transactions until Essbase commits the updated blocks.

Remove Only. An Essbase Spreadsheet Add-in command that you use to remove only the highlighted cells within a worksheet.

replicated partition. A portion of a database, defined through Partition Manager, that you use to propagate an update to data that is mastered at one site to a copy of data that is stored at another site. Users are able to access the data as though it were part of their local database.

report. The formatted summary information that is returned from a database after a report script is run. One or more reports can be generated from a report script.

Report Extractor. An Essbase component that retrieves report data from the Essbase database when you run a report script.

report script. An ASCII file containing Essbase Report Writer commands that generate one or more production reports. Report scripts can be run in batch mode, through the ESSCMD command-line interface, or through Essbase Application Manager. The report script is a text file that contains data retrieval, formatting, and output instructions.

Report Viewer. An Essbase component that displays the complete report after a report script is run. Saved reports typically have the file extension .RPT.

request. A query sent to Essbase by a user or by another process; for example, starting an application, or restructuring a database outline. Requests happen in the context of sessions. Only one request at a time can be processed in each session. A request can be terminated by another user with the appropriate permissions (for example, by an administrator). *See also* session.

restore. An operation to reload data and structural information after a database has been damaged or destroyed. The restore operation is typically performed after you shut down and restart the database.

restructure. An operation to regenerate or rebuild the database index and, in some cases, the data files.

root member. The highest member in a dimension branch.

row. A horizontal display of information in a grid or table. A row can contain data from a single field, derived data from a calculation, or textual information. The words row and record are sometimes used interchangeably. *Contrast with* column.

row heading. A report heading that lists members down a report page. The members are listed under their respective row names.

scope. The area of data encompassed by any Essbase operation or setting; for example, the area of data affected by a security setting. Most commonly, scope refers to three levels of granularity, where higher levels encompass lower levels. From highest to lowest, these levels are as follows: the entire system (an Essbase OLAP server), applications on an OLAP server, or databases within Essbase applications. *See also* persistence.

serial calculation. The default calculation setting. Essbase divides a calculation pass into tasks and calculates one task at a time.

server. A multi-user computer that accesses data values based on the intersection of dimension members.

server application. *See* server.

server console. For Essbase, the computer on which you can enter Agent commands and see messages from the Agent. If you run Essbase in the foreground on either Windows or UNIX, you can enter Agent commands. If you run Essbase in the background, you must stop Essbase and restart as a foreground process before having access to Agent commands. On UNIX, you can use a telnet session to access Essbase remotely. On Windows, you must access Essbase only from the server console.

server interruption. Any occurrence that stops the server, including a crash, a power outage, or a user pressing the Ctrl+C keys.

session. The time between login and logout for a user connected to Essbase. A session can be terminated by another user with the appropriate permissions (for example, an administrator). If a session is processing a request at the time that an administrator attempts to terminate it, the administrator must either terminate the request first, or use “force” to terminate the session and the request simultaneously. *See also* request.

shared member. A member that shares storage space with another member of the same name. The shared member has a property that designates it as shared. The use of shared members prevents duplicate calculation of members that appear more than once in an Essbase outline.

sibling. A child member at the same generation as another child member and having the same immediate parent. For example, the members Florida and New York are both children of the member, East, and siblings of each other.

SMP. *See* symmetric multiprocessing (SMP).

sparse dimension. A dimension with a low probability that data exists for every combination of dimension members.

Spreadsheet Add-in. Essbase software that works with your spreadsheet. Essbase Spreadsheet Add-in is an add-in module to your spreadsheet software.

standard dimension. A dimension that is not an attribute dimension.

subset. A cross-section of data. Subsetting further defines members that meet specific conditions.

substitution variable. A variable that acts as a global placeholder for information that changes regularly. You set the variable and a corresponding string value; the value can then be changed at any time. Substitution variables can be used in calculation scripts, report scripts, Essbase Spreadsheet Add-in, and Essbase API.

suppress rows. The option to exclude rows that contain missing values and to underscore characters from spreadsheet reports.

swapping. *See* paging.

symmetric multiprocessing (SMP). A server architecture that enables multiprocessing and multithreading. Essbase supports multiple threads over SMP servers automatically. Thus, performance is not significantly degraded when a large number of users connect to an Essbase server simultaneously.

system administrator. A person who maintains the hardware, software, disk space distribution, and configurations for running software applications such as Essbase.

TCP/IP. *See* Transmission Control Protocol/Internet Protocol (TCP/IP).

template. A predefined format that is designed to retrieve particular data on a regular basis and in a consistent format.

time series reporting. A process of reporting data based on a calendar date (for example, year, quarter, month, or week).

toolbar. A series of shortcut buttons providing quick access to commands. The toolbar is usually located directly below the menu bar. Not all windows have a toolbar.

Transmission Control Protocol/Internet Protocol (TCP/IP). A standard set of communications protocols that is adapted by many companies and institutions around the world and that links computers with different operating systems and internal architectures. You use TCP/IP utilities to exchange files, send mail, and store data to various computers that are connected to local and wide area networks.

transparent partition. A form of shared partition that provides the ability to access and manipulate remote data transparently as though it is part of your local database. The remote data is retrieved from the data source each time you request it. Any updates made to the data are written back to the data source and become immediately accessible to both local data target users and transparent data source users.

UDA. A user-defined attribute. A UDA is a term associated with members of an outline to describe a particular characteristic of the members. Users can specify UDAs within calculation scripts and reports so that they return lists of members that have the specified UDA associated with them. UDAs can be applied to dense as well as sparse dimensions.

unary operator. A group of mathematical indicators (+, -, *, /, %) that define how roll-ups take place on the database outline.

uncommitted access. An Essbase kernel setting that affects how Essbase handles transactions. Under uncommitted access, concurrent transactions hold short-term write locks and can yield unpredictable results.

Uniform Resource Locator (URL). An address for a resource located on the World Wide Web, such as a document, image, downloadable file, service, or electronic mailbox. URLs use a variety of naming schemes and access methods, such as HTTP, FTP, and Internet mail. An example of a URL is http://www.hyperion.com. A URL can also point to a file located on a local or network drive, such as file:///D:/essbase/docs/essdocs.htm.

URL. *See* Uniform Resource Locator (URL).

validation. A process of checking a rules file, report script, or partition definition against the outline to make sure the object being checked is valid.

visual cue. A formatted style, such as a font or a color, that highlights specific types of data values. Data values may be dimension members; parent, child, or shared members; dynamic calculations; members containing a formula; read only data cells; read/write data cells; or linked objects.

workbook. An entire spreadsheet file with many worksheets.

Index