Hyperion Essbase™

Version 5

Database Administrator's Guide Volume I

HYPERION®

P/N: 06070-500200-000

US Patent Number 5,359,724

Notice: The information contained in this document is subject to change without notice. Hyperion Solutions Corporation shall not be liable for errors contained herein or consequential damages in connection with the furnishing, performance, or use of this material.

Hyperion Solutions Corporation
1344 Crossman Avenue
Sunnyvale, CA 94089

Publication Number: 40-224-9030

Printed in the USA

10 9 8 7 6 5 4 3 2 **R1**

Table of Contents

Volume I

Introduction

Part I Designing Hyperion Essbase Applications

Part II Building Hyperion Essbase Applications

Part III Designing and Building a Security System

Part IV Loading Data

Part V Calculating Your Data

Volume II

Part VI Reporting on Your Data

Part VII Managing Multidimensional Essbase Data Storage

Part VIII Designing and Building Currency Applications

Introduction

Welcome to the Hyperion Essbase™ *Database Administrator's Guide*. To help you get started using this guide, this introduction provides the following information:

- "Who Should Read This Guide" on page xli
- "What's in This Guide" on page xlii
- "What's Not in This Guide" on page xliii
- "What You Should Know Before You Start" on page xliii
- "Sample Applications" on page xliv
- "Document Conventions" on page xlv
- "Additional Documentation Sources" on page xlvi

Who Should Read This Guide

This guide is primarily for:

- Essbase database administrators. An Essbase database administrator is the person who installs, controls, and maintains the Essbase system. You should have prior experience in networking, system administration, and application software.
- Anyone who needs to use the Hyperion Essbase Application Manager to create and maintain applications, databases, data load rules, calc scripts, and report scripts.

What's in This Guide

This guide provides:

- Database administrators with strategies and techniques to implement, design, and maintain an optimized Essbase multidimensional database. It provides a technical discussion of Essbase concepts to help you think about and manage data multidimensionally. You can then effectively design an Essbase database on your own.
- Step-by-step procedures on how to use the Application Manager.

Use this guide to:

- Learn about Essbase architecture, philosophy, and concepts.
- Design a multidimensional application.
- Learn techniques for partitioning large databases.
- 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 your database.
- Backup and restore data.
- Manage large databases.

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

- Volume I
    - Part I, Designing Hyperion Essbase Applications
    - Part II, Building Hyperion Essbase Applications
    - Part III, Designing and Building a Security System
    - Part IV, Loading Data
    - Part V, Calculating Your Data
- Volume II
    - Part VI, Reporting on Your Data
    - Part VII, Managing Multidimensional Essbase Data Storage
    - Part VIII, Designing and Building Currency Applications
    - Part IX, Maintaining Hyperion Essbase Applications

What's Not in This Guide

This guide does not describe the following:

- Command and function syntax. For this information, see the online *Technical Reference* in your `DOCS` directory.
- Procedures for installing the Hyperion Essbase OLAP Server. For this information, see the *Installation Notes*.
- Basic data retrieval. For this information, you need to use the spreadsheet environment. See the Essbase *Spreadsheet Add-in User's Guide*s in your `DOCS\CLIENT` directory.

What You Should Know Before You Start

To use all the information in this guide, you need the following:

- A working knowledge of the operating system your server uses and the ones your clients use. Essbase supports Windows 95, Windows 98, Windows NT, Solaris, AIX, HP-UX, and DEC Alpha NT.

Note: With the release of Hyperion Essbase 5.0.2, OS/2 versions of Hyperion Essbase OLAP Server, Hyperion Essbase SQL Interface, and Hyperion Essbase API are no longer provided on the product CD. If you want to use OS/2, use the Hyperion Essbase 5.0.1 or earlier product CD.

- An understanding of typical database administration requirements and tasks, including security, setting up user accounts, and maintaining the Essbase database.
- Knowledge of how many users your server can accommodate and how to manage the space on your server.
- Knowledge of where the data resides in your business and who is responsible for updating Essbase with current data.
- Knowledge of your business's data requirements, so you can apply Essbase to your specific application.

However, you might simply want to understand the basics of how Essbase works, or use the Application Manager to do regular tasks, such as running report scripts or calc scripts. In this case, you need only a basic understanding of Windows and basic Windows terminology, such as "dialog box," "list box," and "button." See your Windows documentation for more information on these terms.

Sample Applications

This book provides examples based on applications provided with the Essbase server software, called Sample, Demo, Samppart, and Sampeast. (Samppart and Sampeast are examples of Partitioning. These applications are available only if your company has licensed Hyperion Essbase Partitioning.) The individual who installs the server is responsible for making these 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 don’t have adequate access to the applications.
- You don’t see any data in the databases.

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

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
1, 2, 3	Numbered Items indicate procedures to follow in a specific order.
•	Bulleted items indicate a list of related items.
Product	Dimension names and member names appear in This Font.
`\ESSBASE`	Names of files, directories, and specific text you must type appear in `This Font`.
Essbase System Login dialog box	Names of dialog boxes and their controls, such as buttons, text boxes, and list boxes, appear in **boldface**.
Installation Notes	Titles of books appear in *italics*. Italics also indicate important terms and special emphasis.
File | Open	The vertical bar (|) indicates a menu followed by an individual menu command in that menu.

Additional Documentation Sources

If you don't find what you need in this book, refer to the following:

- The online *Technical Reference* in your `DOCS` directory, which contains a list and description of Essbase functions, macros, calculation commands, report commands, ESSCMD commands, and configuration file settings.
- The Application Manager online help file, which contains descriptions of menu commands and dialog boxes.

Essbase also includes these sources of documentation:

- The *Installation Notes*, which show you how to install and configure the Essbase server, Spreadsheet Add-in, Application Manager, SQL Interface, API, Runtime Client, and sample applications.
- The *SQL Interface Guide*, which contains information on how to set up your systems to load data via the Hyperion Essbase SQL Interface, licensed separately.
- The *Spreadsheet Add-in User's Guide for Excel* explains how to use the Essbase spreadsheet features with Microsoft Excel for Windows. This guide is provided in the `\ESSBASE\DOCS\CLIENT` directory in `.PDF` format for online viewing and printing in Adobe Acrobat (Version 3.0.1 or higher). Adobe Acrobat Reader is provided on the Essbase CD ROM, or you can download Acrobat Reader from `www.adobe.com`.
- The *Spreadsheet Add-in User's Guide for 1-2-3* explains how to use the Essbase spreadsheet features with Lotus 1-2-3 for Windows. This guide is provided in the `\ESSBASE\DOCS\CLIENT` directory in `.PDF` format for online viewing and printing in Adobe Acrobat (Version 3.0.1 or higher). Adobe Acrobat Reader is provided on the Essbase CD ROM, or you can download Acrobat Reader from `www.adobe.com`.
- The Spreadsheet Add-in online help file contains basic information about your spreadsheet environment, tasks for using all aspects of the Spreadsheet Add-in, descriptions of menu commands and dialog boxes, and all the spreadsheet macros and VBA functions.
- The *SQL Drill-Through Guide*, which contains information about how to access detail-level data stored in a remote SQL database via the Hyperion Essbase SQL Drill-Through (licensed separately). This guide is provided in the `\ESSBASE\DOCS\CLIENT` directory in `.PDF` format for online viewing and printing in Adobe Acrobat (Version 3.0.1 or higher). Adobe Acrobat Reader is provided on the Essbase CD ROM, or you can download Acrobat Reader from `www.adobe.com`.
- The online *API Reference* in your `DOCS` directory, which contains a complete listing and description of functions available through the Essbase API.

Part I Designing Hyperion Essbase Applications

Part I introduces you to the Hyperion Essbase OLAP Server by describing general on-line 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, Introducing Essbase OLAP, describes the parts of Essbase, high-level Essbase functionality, key architectural features, and how Essbase works in a client/server environment.
- Chapter 2, Steps for Implementing Essbase, provides a high-level process map for implementing Essbase in your organization with cross references to further information.
- Chapter 3, 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 4, 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 5, 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, Designing Partitioned Applications, introduces you to the various types of database partitions and provides rules for choosing and implementing partitions that span multiple databases, applications, or computers. It contains scenarios that illustrate how to design a partitioned application.

Chapter 1 Introducing Essbase OLAP

This chapter introduces Hyperion Essbase and describes the Essbase environment. Essbase is a multidimensional database server 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 1-1
- "Multidimensional Development Features" on page 1-2
- "Essbase Architectural Features" on page 1-3
- "Getting Started with Essbase" on page 1-5

About the Essbase Product Family

The Essbase product family includes the following feature sets:

- **Hyperion Essbase Application Manager**
  A graphical environment for developing and maintaining Essbase applications. This includes building outlines and dimensions, performing data loading and calculations, and defining security access.
- **Hyperion 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 your analytical data model, data storage, calculations, and data security.
- **Hyperion Essbase Spreadsheet Add-in**
  Desktop software that lets you analyze the data stored in the Essbase server. It is seamlessly integrated with Microsoft Excel or Lotus 1-2-3 spreadsheets.

- **Essbase Application Tools**
  A suite of tools for extending Essbase applications. These tools include Hyperion Essbase Currency Conversion, Hyperion Essbase SQL Interface, Hyperion Essbase Spreadsheet Toolkit, Hyperion Essbase API, and Hyperion Essbase SQL Drill-Through.
- **Hyperion Essbase Partitioning**
  A suite of features that 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 directly to other databases from your own.

Multidimensional Development Features

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

- Essbase requires no knowledge of query languages, and minimal programming experience. You can design and manage applications using a graphical interface to control most server functions.
- You can add new data dimensions, change calculations, and modify data 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, which operate on very large data sets. You can perform key analytical functions, such as trend analysis, ratios, and allocations, without programming. You can define calculations quickly 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 programming.
- Essbase is easy to deploy and supports standard applications, operating systems, and networking protocols.

Essbase Architectural Features

Essbase incorporates 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 optimizing analytical performance. This method separates data into sparse and dense dimensions. See Chapter 3, Multidimensional Concepts, and Chapter 4, 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.

Multithreaded Design Supports SMP

The Essbase server is a 32-bit, multithreaded software application, and 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 supports multiple threads over SMP servers automatically. This means that performance is not significantly degraded, even with a large number of simultaneous users.

Multi-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.

Client/Server Overview

The Essbase client/server architecture supports enterprise analysis applications. The server runs Essbase software and fields requests from clients. A network connects the server and the clients to each other. The server is typically a PC or UNIX machine. The clients are PCs or UNIX workstations that also run Essbase software.

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 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.

The Essbase Server

All Essbase application components, including database outlines and calc scripts, application control, and multidimensional database information reside on the server. Essbase lets you configure server disk storage to span multiple disk drives, so you can store large databases. Essbase requires your server to run a multithreaded operating system so it 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 UNIX, Windows NT, Windows 95, and Windows 98 workstations and servers. See the *Installation Notes* for specific information on server configuration requirements.

Note: With the release of Hyperion Essbase 5.0.2, OS/2 versions of Hyperion Essbase OLAP Server, Hyperion Essbase SQL Interface, or Hyperion Essbase API are no longer provided on the product CD. If you want to use OS/2 or Windows 3.1 versions of the client, use the Hyperion Essbase 5.0.1 or earlier product CD.

The Essbase Client

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

There are three types of Essbase clients:

- The first client interface is the Spreadsheet Add-in, which provides users with seamless data access to the server. The *Spreadsheet Add-in User's Guide* describes how to use the spreadsheet client interface.
- The second client interface is the Essbase Application Manager, which designs, develops, and maintains Essbase applications. This manual provides task-oriented instructions for using the Application Manager.
- The third client interface is a custom application built with the Essbase Application Programming Interface (API), which enables application developers to create custom interfaces to Essbase quickly, using standard tools. You can use the API with Microsoft Visual Basic, Microsoft Visual C++, and specific C compilers on platforms such as Solaris, HP-UX and AIX. Other programming languages, such as Borland Delphi and PowerBuilder, are sometimes used with the Essbase API, although they are not specifically tested by your software provider. The online *API Reference* in your `DOCS` directory provides a complete listing of functions, platforms, and supported compilers.

See the *Installation Notes* for specific information on client configuration requirements. See the *Start Here* booklet for information about supported platforms for Essbase products.

Getting Started with Essbase

The next chapter provides a table showing what tasks you need to complete to get up and running with Essbase, and where to find information about each task. Read Chapter 2, Steps for Implementing Essbase, as a guide to implementing the product.

Chapter 2 Steps for Implementing Essbase

You've just purchased Hyperion Essbase. You have a lot of data to organize and analyze, and you need to get up and running very quickly. You want to get started using Essbase to integrate your data as efficiently as possible.

The following table tells you the steps in getting up and running with Essbase, and where you can find more information about each step.

Note: This chapter assumes you are a new Essbase user. If you used Version 4, you need to migrate your applications and databases. See the *Start Here* booklet for important migration information.

Table 2-1, A Process Map

Process step	For information, see...
Install Essbase: Decide what components you want to install. Be aware that the license your company purchased might not include all options.	*Installation Notes*
Assess your needs and requirements. Have a clear idea of your data analysis needs, and what types of calculations and reports you want to run.	Your budget, forecasting, and other financial reports with notes on how you would like to improve them.
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 need Hyperion Essbase SQL Interface.	Chapter 3, Multidimensional Concepts

Table 2-1, A Process Map

Process step	For information, see...
Learn the fundamentals of Essbase and distributed OLAP.	Chapter 1, Introducing Essbase OLAP Chapter 3, Multidimensional Concepts Chapter 4, Basic Architectural Elements Chapter 7, Creating Applications and Databases Chapter 6, Designing Partitioned Applications Attend a Fundamentals 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, and which you will designate as Time and Account dimensions.	Chapter 5, Designing a Single-Server Application
Estimate the size of your database.	Chapter 14, Sizing Your Database
Allocate storage and specify Storage Manager settings for your database.	Chapter 40, Specifying Storage Manager Settings
Learn about Hyperion Essbase Partitioning. Think about whether your data would benefit from being decentralized into connected databases.	Chapter 6, Designing Partitioned Applications
Learn about Dynamic Calculations and how they can greatly improve performance.	Chapter 28, Dynamically Calculating Data Values
Create an application and a database.	Chapter 7, 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 III, Designing and Building a Security System
Build an outline for your database.	Chapter 8, Creating and Changing Database Outlines
Build the dimensions. Decide whether your data loads would introduce new members into the outline. If so, set up dynamic dimension building. If not, set up regular data loads.	Chapter 12, Introducing Dynamic Dimension Building Chapter 13, Building Dimensions Using a Rules File
Load your data.	Part IV, Loading Data
Calculate your database.	Part V, Calculating Your Data

Table 2-1, A Process Map

Process step	For information, see...
Run a report.	Part VI, Reporting on Your Data
Use the Spreadsheet Add-in to retrieve data.	The Essbase *Spreadsheet Add-in User’s Guide* for your spreadsheet application.
Maximize the efficiency of your databases using Essbase Partitioning.	Chapter 15, Building and Maintaining Partitions
Link files or cell notes to data cells.	Chapter 11, Linking Objects to Your Essbase Data
Connect to a relational database.	*SQL Interface Guide* *INTERSOLV DataDirect ODBC Drivers Reference*
Assign alias names to your members.	Chapter 10, Creating and Managing Aliases
Copy or export data subsets.	Chapter 38, Copying Data Subsets and Exporting Data to Other Programs
Back up and restore your data.	Chapter 47, Backing Up Data and Recovering Databases
Design a currency application.	Chapter 42, Designing and Building Currency Conversion Applications
Fine-tune your storage settings.	Chapter 39, Introducing the Essbase Storage Manager Chapter 40, Specifying Storage Manager Settings
Automate routine operations using ESSCMD.	Chapter 43, Performing Interactive and Batch Operations Using ESSCMD
Maintain your applications.	Part IX, Maintaining Hyperion Essbase Applications

Chapter 3 Multidimensional Concepts

Hyperion Essbase OLAP Server contains multidimensional databases that support analysis and management reporting applications called Online Analytical Processing (OLAP). This chapter discusses multidimensional concepts and terminology. This chapter contains the following sections:

Introducing OLAP

In 1993, E. F. Codd, who first set the seminal rules describing relational databases, published twelve rules for analytical functions and performance characteristics 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 (MDD) supports multiple views of data sets for users who need to analyze the relationships between data categories. For example, a marketing analyst might want to know:

- 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 it sell by branch, region, and territory?
- Did this product sell better in different regions? Are there regional trends?

- Did customers return Product A last year? Were these 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 this 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, zoom in to find out more detailed information, or zoom out 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). He also covers the kind of database operations a multidimensional database should support, including:

- 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.

Dimensions represent the core components of your business plan, and often relate to your departmental functions. Typical dimensions can include Time, Accounts, Product Line, Market, and Division. Dimensions are static in most databases; once you determine the database dimensions, they rarely change over the life of the application.

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.

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 3-1, 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 zoom and pivot data.

Arranging Dimensions into Hierarchies

All Essbase database development begins with creating a database outline. A database outline:

- Defines the structural relationships between members in an 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.

As an example of consolidation, many businesses summarize their data monthly, then roll up the monthly data to get quarterly figures, and roll up the quarterly data to get 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, for example, the Year dimension consists of five members: the Qtr1, Qtr2, Qt3, 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 each consist of the members that represent individual months plus the member that stores the quarterly totals.

The database outline in Figure 3-1 uses a hierarchical structure to represent the data consolidations and relationships in Qtr1.

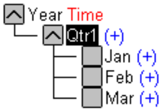


Figure 3-1, Hierarchical Structure

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. You can always divide a member into other members to reflect the proper granularity of your data.

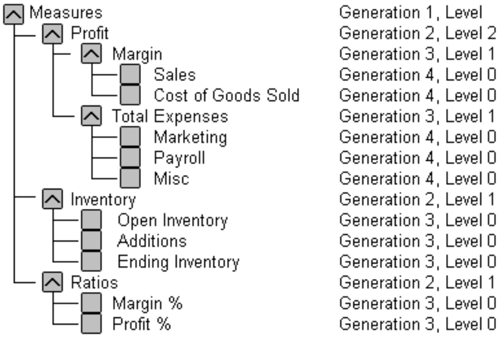
For information on creating a database outline, see Chapter 8, Creating and Changing Database Outlines.

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, family history terms to describe the roles and relationships of the members in an outline. You can describe the position of the members in each branch in Figure 3-2 in several ways.



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

Figure 3-2, Member Generation and Level Numbers

Parents, Children, and Siblings

Figure 3-2 illustrates the following parent, child, and sibling relationships:

Parents: 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.

Children: 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: A sibling is a child member at the same branch level as another member. For example, Sales and Cost of Goods Sold are siblings of the parent Margin, but Marketing (at the same branch level) is not a sibling because its parent is Total Expenses.

Descendants and Ancestors

Figure 3-2 illustrates the following descendant and ancestral relationships:

Descendants: Descendants are members 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: Ancestors are members in a branch above a member. For example, Margin and Profit are ancestors of Sales.

Roots and Leaves

Figure 3-2 illustrates the following root and leaf member relationships:

Root: The root is the first member in a branch. Measures is the root for Profit, Inventory, and Ratios and their children.

Leaves: 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 nodes.

Generations and Levels

Figure 3-2 illustrates the following generations and branch levels:

Generations: Generations refer to consolidation levels within each dimension. The root branch of the tree is Generation 1. The generations count down from the root toward the leaf node. In Figure 3-3, 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 3-3 shows part of the Product dimension with its generations numbered.

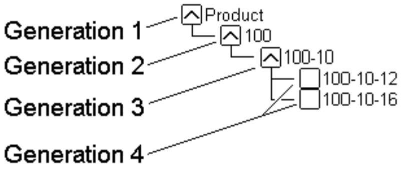


Figure 3-3, Generations

Levels: Levels refer to the branches within each dimension; however, they reverse the numerical ordering Essbase uses for generations. The levels count up from the leaf node toward the root. The root level number varies depending on the depth of the branch. In this example, Sales and Cost of Goods Sold are Level 0. All other leaf nodes 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 3-4 shows part of the Product dimension with its levels numbered.

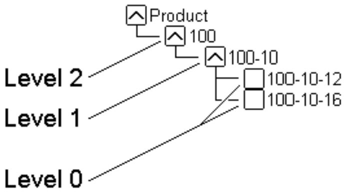
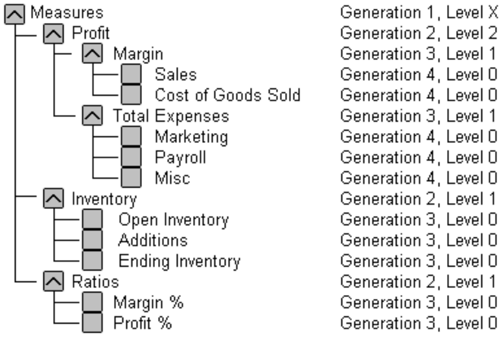


Figure 3-4, Levels

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



Measures	Generation 1, Level X
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

Figure 3-5, Database Outline Section with Generations and Levels

Actions to Perform on Multidimensional Data

The following sections describe some of the most common actions you can perform on multidimensional data.

Consolidate, Aggregate, or Roll-Up

Consolidating, aggregating, or rolling up data means to compute the data relationship for all parent/child combinations within a dimension. A consolidation is typically additive, but can be any type of calculation. For example, the consolidation for the Year dimension is Qtr1 + Qtr2 + Qtr3 + Qtr4 = Year.

Zoom-In or Drill-Down

Zooming in or drilling down is the process of retrieving progressively more detailed data relative to a selected dimension. Zooming in on a database dimension provides you with greater detail on that dimension, while zooming out moves your perspective to a higher consolidation level. For example, you can zoom in on the Year dimension to view the values for each quarter. Then you can zoom in on Qtr1 to see the values for each month in that quarter.

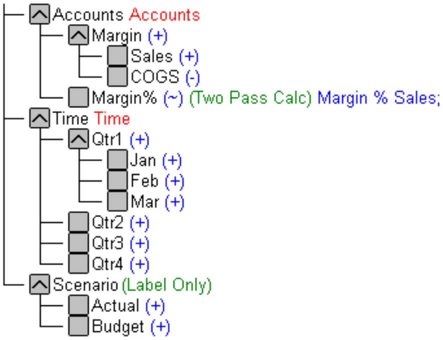
Pivot

Pivoting is the ability to alter the perspective of data retrieved. When Essbase first retrieves a dimension, it expands into rows. A user can pivot or rearrange the data to obtain a different viewpoint.

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* dimension

Consider the simplified database in Figure 3-6:



```
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 (+)
```

Figure 3-6, A Multidimensional Database Outline

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

- The Accounts dimension has four members: Sales, COGS, Margin, Margin%.
- The Time dimension has four quarters. In the example, we consider only the members in Qtr1.
- The Scenario dimension has two child members: Budget for budget values and Actual for actual values.

The intersection of one member from each dimension represents a data value in the database. Our example has three dimensions; we can represent the dimensions and data values in the database as the cube in Figure 3-7:

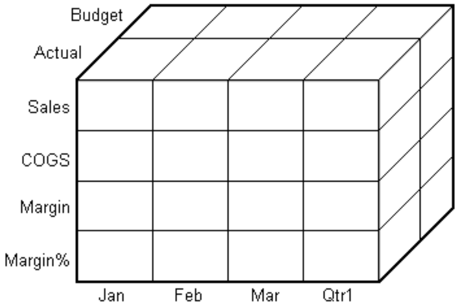


Figure 3-7, Three Dimensional Database

The shaded cells in Figure 3-8 illustrate that when you refer to Sales, you are referring to a slice of the database containing eight Sales values:

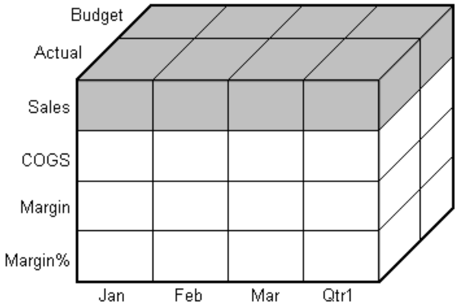


Figure 3-8, Sales Slice of the Database

When you refer to Actual Sales, you are referring to four Sales values:

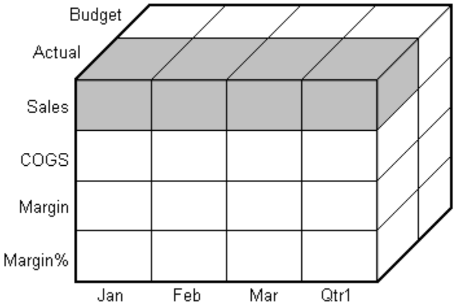


Figure 3-9, Actual, Sales Slice of the Database

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 3-10, the cell containing the data value for Sales, Jan, Actual is shaded. This can also be expressed as, Sales, Actual, Jan or, using the cross-dimensional operator, (->), Sales->Actual->Jan.

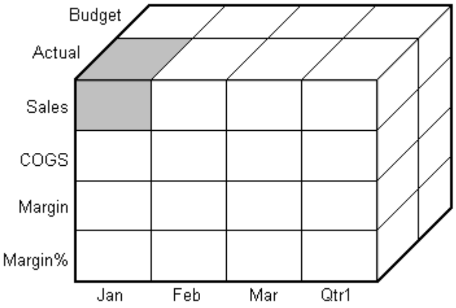


Figure 3-10, Sales, Jan, Actual Slice of the Database

Looking at the Data from Different Perspectives

Slicing the database in different ways gives you different perspectives of the data. The slice in Figure 3-11, for example, shows data about the month of January:

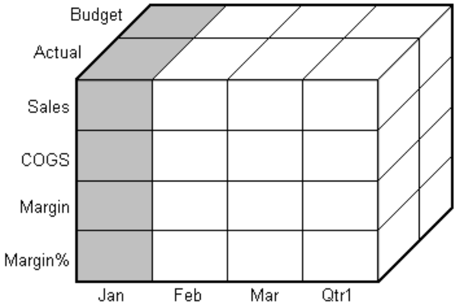


Figure 3-11, Data for January

The slice in Figure 3-12 shows data for the month of February:

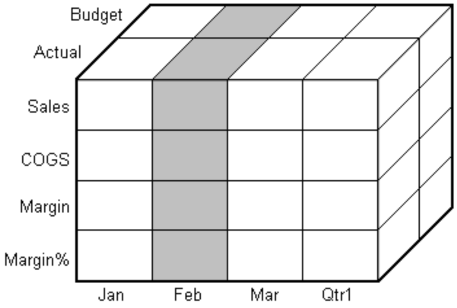


Figure 3-12, Data for February

The slice in Figure 3-13 shows data for profit margin:

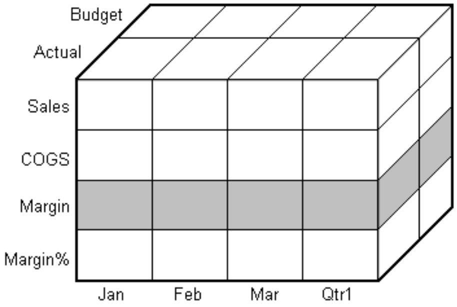


Figure 3-13, Data for Profit Margin

Slicing a database amounts to fixing one or more dimensions at a constant value while allowing the other dimensions to vary. The slice of January, for example, examines all data values for which the Year dimension is fixed at Jan.

Designing and Creating a Simple Application

In this section, you'll learn how to think multidimensionally by building and analyzing a sample application called Simple. This section uses the same process to analyze data as you would when building a typical multidimensional database.

First, let's analyze our typical company called The Car Company (TCC). TCC manufactures, markets, and distributes cars and trucks across the United States. Analysts at TCC prepare budget forecasts and track performance on a monthly basis.

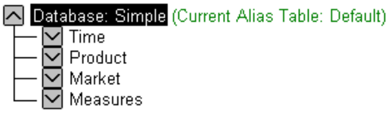
Because TCC plans and tracks a variety of products over several markets, the process of deriving and analyzing data is quite tedious. Last month, analysts spent the majority of their time entering, re-keying, and preparing reports.

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. Since all users have access to the server, they can retrieve data at will, regardless of its origin. To solve their problem, TCC chooses Essbase.

Organizing Multidimensional Data

First, you'll create a database outline for TCC. The outline defines the structure of the database, including the dimensions and members it contains. It's 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 outline to support the change. For more information on how to build an outline, see Chapter 8, Creating and Changing Database Outlines.

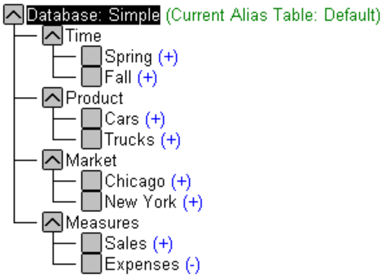
A database outline contains dimensions and members. Your first job is to determine the logical structure for the data. Remember, the dimensions often parallel your company's organization. TCC has four dimensions: Time, Product, Market, and Measures.



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

Figure 3-14, TCC Simple with Four Dimensions

Let's give each dimension two members:



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

Figure 3-15, TCC Simple with Dimensions and Members

To appreciate the power of the multidimensional database, you need to understand how Essbase organizes members and 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 dimensions in the database.

For example, a spreadsheet cell could be the intersection of Row 3 and Column 4. An Essbase data value is defined by the intersection of one member on each of the dimensions. For example, a data value could be at the intersection of Spring, Cars, Chicago, and Sales. See Chapter 4, Basic Architectural Elements, for more information on how Essbase stores the data values.

The database size is defined by its dimensions and members. For example, TCC is a four-dimensional database with three members in each dimension (including the dimension member). To determine the maximum number of values in the database, multiply together 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's easy to see from this example how fast a multidimensional database can grow.

Let's 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?* We'll begin with the member Spring, and list combinations of dimensions and members from the database outline for Spring. Remember, a data value must be defined by one member from each dimension. The following table contains all of the values that make up this list, with the consolidated members in bold.

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
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

Time	Product	Market	Measures	Data Value
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 could 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?

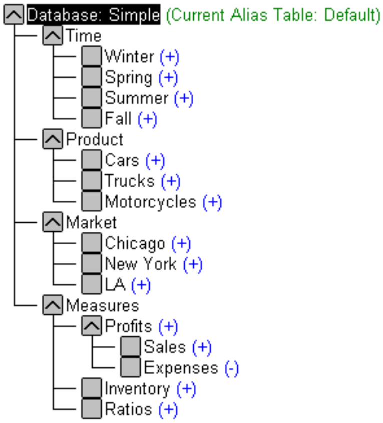
A typical database would contain associated formulas and a calc script to analyze the data. For example, you might want to calculate the variance between budget and actual expenses values. You could do this by defining the appropriate formula on a Variance member. For more information on developing formulas and calc scripts, see Chapter 24, Introduction to Database Calculations.

Adding and Deleting Dimensions and Members

Let's apply a few hypothetical situations to the TCC database. In this section, you'll consider what happens to the multidimensional database when you add and delete members and dimensions.

Adding New Members

In this example, you'll add several new members to the database under each dimension. Let's add two more seasons under Time, one more product called Motorcycles, a new market, LA, and three new members to the Measures dimension: Profits, Inventory, and Ratios. For more information on how to build an outline, see Chapter 8, Creating and Changing Database Outlines.



```
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 (+)
```

Figure 3-16, Adding a Member

Let's create a similar table to the one in the first example.

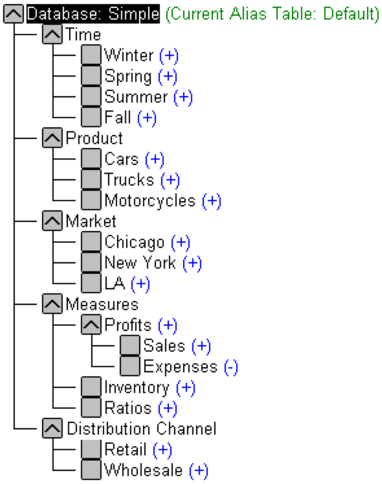
Time	Product	Market	Measures	Data Value
Winter	Cars	Chicago	Sales	1000
Winter	Cars	Chicago	Expenses	600
Winter	Cars	Chicago	Profits	
Winter	Cars	Chicago	Inventory	1600
and so on	...	...	...	...

The new database is potentially 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 together the number of members in each dimension: 5 x 4 x 4 x 6 = 480. So, by adding seven new members, the Simple multidimensional database has now grown to a potential 480 values.

Adding a New Dimension

Let's add a new dimension called Distribution Channel to the database. In addition, you'll give the new dimension two members, Retail and Wholesale. For more information, see "Adding Dimensions and Members to Outlines" on page 8-13.



```
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 (+)
```

Figure 3-17, Adding a New Dimension

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

When you add a new dimension to an outline, you must associate any data in the database with a single member in the new dimension. For example, in the Simple database, you would have to specify what type of channel the existing data represented. You would need to load data for the other members on the new dimension, and calculate the database. For more information, see Chapter 19, Introducing Data Loading, and Chapter 24, Introduction to Database Calculations.

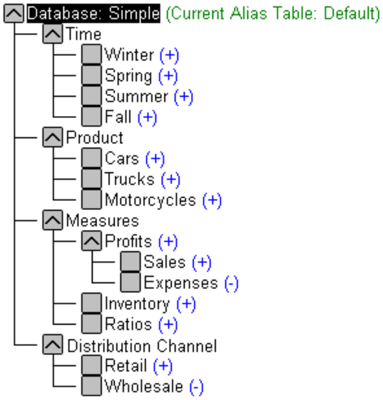
Removing a Dimension

In this example, you'll delete the Market dimension from the database. The TCC company wants the Simple database to represent the LA market only, so there is no need for a Market dimension. For more information, see Chapter 8, Creating and Changing Database Outlines.

What impact does this decision have on the database? One less dimension diminishes the overall size of the database. However, data for all members in the Market dimension still exists. The rule is that, if you delete a dimension from a database outline, the data associated with only one member of the deleted 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 would choose to retain the LA data.

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



```
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 (-)
```

Figure 3-18, Deleting a Dimension and Members

Chapter 4 Basic Architectural Elements

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

Sparse and Dense Dimensions

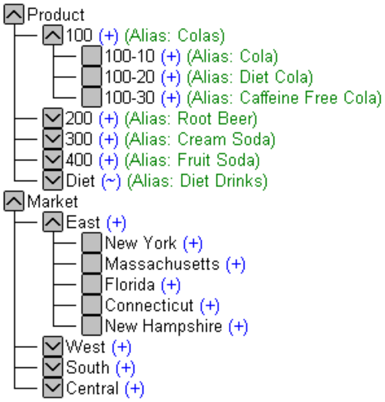
Most multidimensional application data sets have two characteristics:

- Data is *not* smoothly and randomly 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 an application's dimensions into two types: dense dimensions and sparse dimensions. This 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 includes the Product, Market, Measures, Year, 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 that occurs. 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.



```
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)
Market
├─ East (+)
│   ├─ New York (+)
│   ├─ Massachusetts (+)
│   ├─ Florida (+)
│   ├─ Connecticut (+)
│   └─ New Hampshire (+)
├─ West (+)
├─ South (+)
└─ Central (+)
```

Figure 4-1, Sample Basic Database Outline

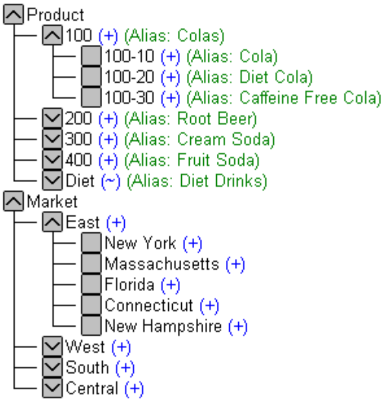
Data Blocks and the Index System

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 dimension members (providing at least one data value exists for the sparse dimension member combination). The data block represents all the dense dimension members for that combination of sparse dimension members.

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

For example, in the Sample Basic database Product and Market are sparse dimensions.



```
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)
Market
  East (+)
    New York (+)
    Massachusetts (+)
    Florida (+)
    Connecticut (+)
    New Hampshire (+)
  West (+)
  South (+)
  Central (+)
```

Figure 4-2, Product and Market Dimensions from the Sample Basic Database

If data exists for Caffeine Free Cola in New York, then Essbase creates a data block and index entry for the sparse member combination 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 index entry, for the sparse member combination 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.

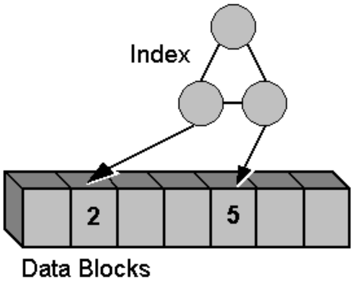


Figure 4-3, Simplified Index and Data Blocks

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 a data block. 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 only includes pointers to existing data blocks.

Figure 4-4 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.

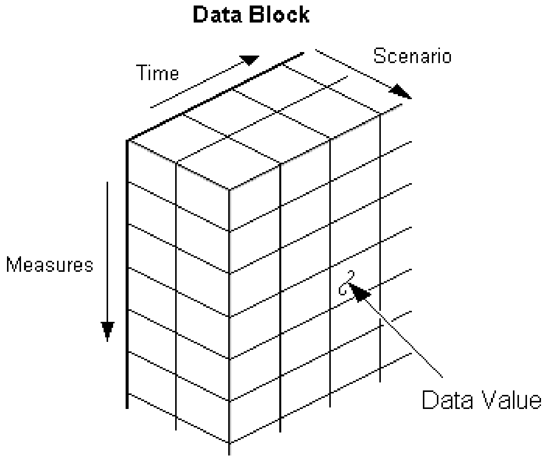


Figure 4-4, Part of a Data Block for the Sample Basic Database

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 in your database outline.

Figure 4-5 shows the data block for the member combination d22,e3 in the following database outline. The block represents the three dense dimensions. 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 (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
```

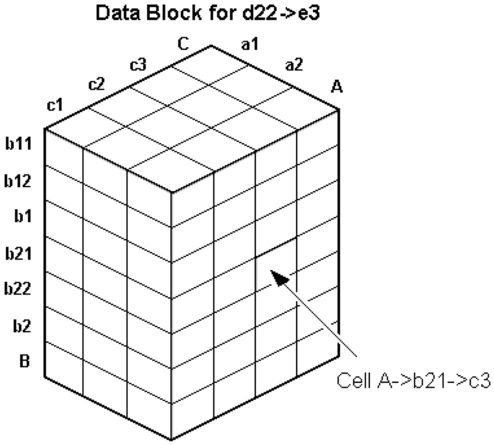


Figure 4-5, Data Block Representing Dense Dimensions for d22->e3

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 4-5, 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 enabled by default. For more information, see "Specifying Data Compression" on page 40-21.

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

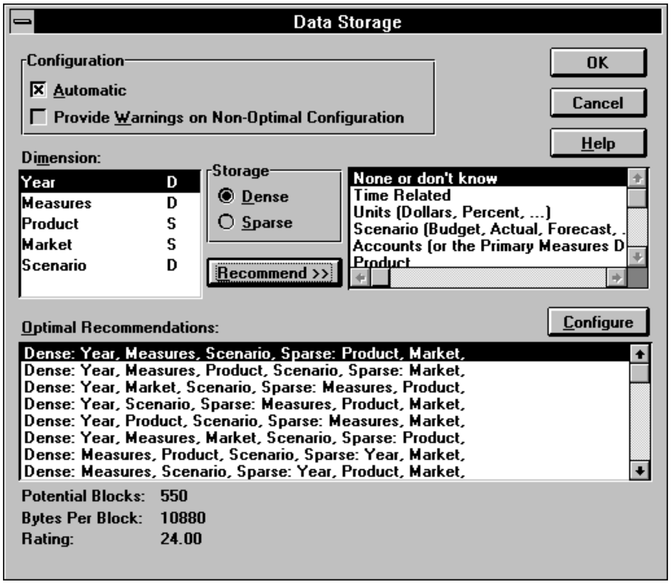
Selecting Sparse and Dense Dimensions

In most data sets the existing data tends to follow predictable patterns of density and sparsity. If you match the 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.

When you create a database outline in the Outline Editor, Essbase automatically suggests which of your dimensions should be sparse and which dense. Essbase does this by, for example, considering the Time and Accounts tags on dimensions and the probable size of the data blocks. For more information on Time and Accounts tags, see Chapter 9, Setting Dimension and Member Attributes. For more information on using the Outline Editor to create database outlines, see Chapter 8, Creating and Changing Database Outlines.

Note: The auto-configuration of dense and sparse dimensions provides an estimate only. It cannot take into account the nature of the data you will be loading into your database or multi-user locking considerations.

You can turn off the auto-configuration and manually choose your sparse and dense dimensions. To help you select sparse and dense dimensions, Essbase provides data storage information in the **Application Manager Data Storage** dialog box. This information includes the data block size and density to help you choose the optimal configuration for your database. For more information, see Chapter 40, Specifying Storage Manager Settings.



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
Product

Optimal Recommendations:

Dense: Year, Measures, Scenario, Sparse: Product, Market,
Dense: Year, Measures, Product, Scenario, Sparse: Market,
Dense: Year, Market, Scenario, Sparse: Measures, Product,
Dense: Year, Scenario, Sparse: Measures, Product, Market,
Dense: Year, Product, Scenario, Sparse: Measures, Market,
Dense: Year, Measures, Market, Scenario, Sparse: Product,
Dense: Measures, Product, Scenario, Sparse: Year, Market,
Dense: Measures, Scenario, Sparse: Year, Product, Market,

Potential Blocks:	550
Bytes Per Block:	10880
Rating:	24.00

Figure 4-6, Data Storage Dialog Box

Consider the Sample Basic database shipped with Essbase. This 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 on the Product and Market dimensions. For example, if Caffeine Free Cola is not sold in Florida, then data values may 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 on these dimensions, Essbase does not create a data block for the combination.

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

In summary:

- Sparse: Product and Market
- Dense: 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:

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

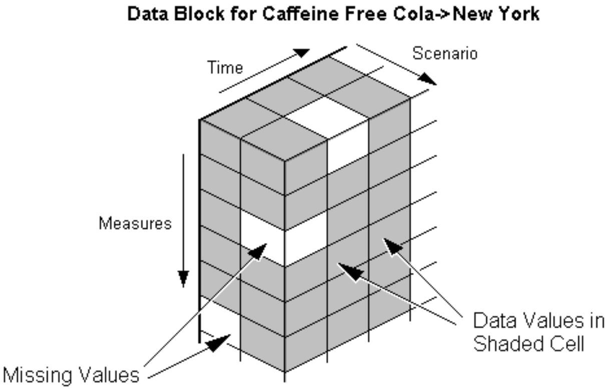

Data Block for Caffeine Free Cola->New York
Time
Scenario
Measures
Data Values in Shaded Cell
Missing Values

Figure 4-7, Dense Data Block for Sample Basic Database

Dense and Sparse Selection Scenarios

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

Scenario 1: A Database Consisting Entirely of Sparse Dimensions

If you make all dimensions sparse, Essbase creates data blocks that consist of a single data cell, which contains a single data value. There is an index entry for each data block and therefore, in this scenario, for each existing data value.

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

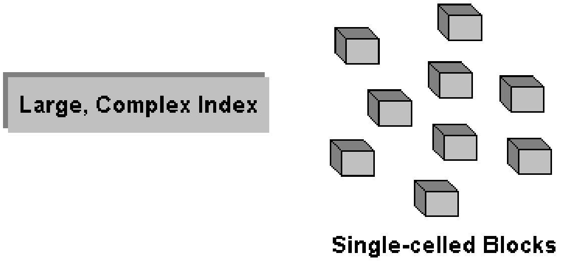


Figure 4-8, Database with All Sparse Dimensions

Scenario 2: A Database Consisting Entirely of Dense Dimensions

If you make all dimensions dense, Essbase creates one index entry and one very large, very sparse block. In most applications, this 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.

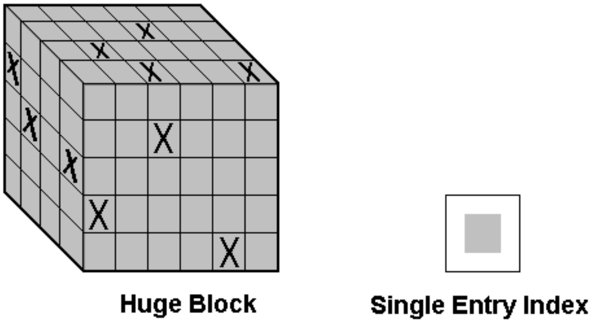


Figure 4-9, Database with All Dense Dimensions

Scenario 3: An Ideal Configuration of Dense and Sparse Dimensions

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

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

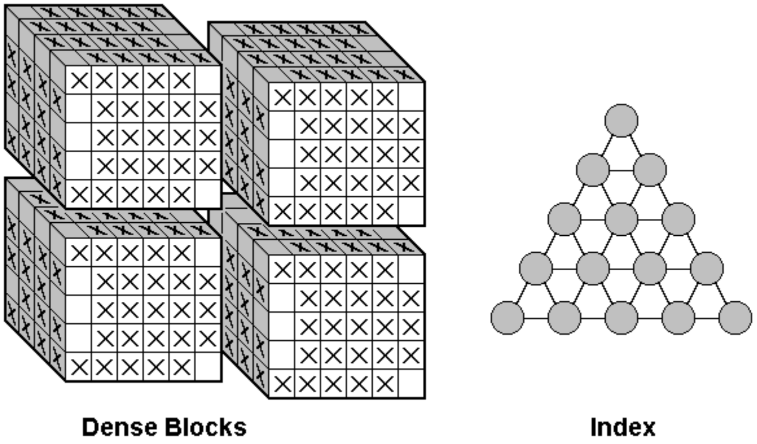


Figure 4-10, An Ideal Configuration

Scenario 4: A Typical Multidimensional Problem

Consider a database with four 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 represent data values from the dense dimensions: Time and Accounts. The members on the Time dimension are J, F, M and Q1. The members on the Accounts dimension are Rev, Exp, and Net.

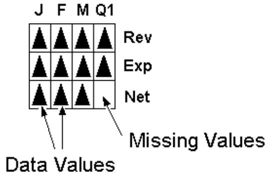


Figure 4-11, Two-dimensional Data Block for Time and Accounts

Essbase creates data blocks for combinations of members on the sparse 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 on the Product dimension are Product A, Product B, Product C, and Total Product.

Figure 4-12 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.

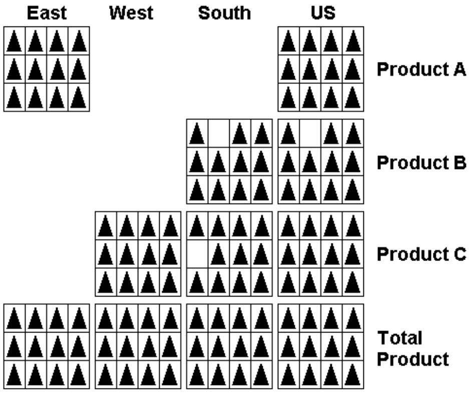


Figure 4-12, Data Blocks Created for Sparse Members on Region and Product

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

Now let's reverse the dense and sparse dimension selections. In the following example, Region and Product are dense dimensions and Time and Accounts are sparse dimensions.

The two-dimensional data blocks represent data values from the dense dimensions: Region and Product.

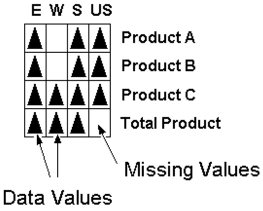


Figure 4-13, Two-Dimensional Data Block for Region and Product

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

Figure 4-14 shows 12 data blocks. Data values exist for all combinations of members on 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.

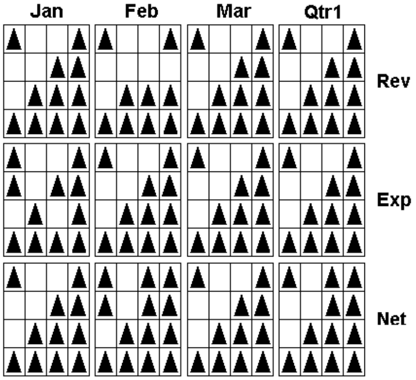


Figure 4-14, Data Blocks Created for Sparse Members on Time and Accounts

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?
- How do you plan to compress the data?
- How do you want to create and order calculations?

For more information on:

- Planning the development of your multidimensional database, see Chapter 5, Designing a Single-Server Application.
- Selecting dense and sparse dimensions, see “Sparse and Dense Dimensions” on page 4-1.
- Loading data, see Chapter 19, Introducing Data Loading.
- Compressing data and optimizing your database, see “Specifying Data Compression” on page 40-21.
- Calculating your database, see Chapter 24, Introduction to Database Calculations.

Chapter 5 Designing a Single-Server Application

To implement a multidimensional database after you have installed the Hyperion Essbase OLAP Server, you must design and create an application. You should analyze your data sources and define your requirements very carefully before deciding 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 your applications, see Chapter 6, Designing Partitioned Applications.

Note: You must license Hyperion Essbase Partitioning separately from the Essbase server for each server to contain a database partition.

This chapter provides an overview of the database planning process and working rules you can follow to design a single-server, multidimensional database solution for your organization. You will also find detailed examples that show how to apply these rules. For detailed information about building applications and databases, see Chapter 7, Creating Applications and Databases.

This chapter includes the following sections:

Process Overview

When designing an application, you need to complete the following basic steps:

1. **Create an application and database.** An Essbase *application* is the container for databases, calc scripts, and other related files. You can create an application on a client or server machine. The application must exist before you can create the database.
2. **Define the database outline.** The *outline* determines the structure of the database—what information will be stored and how different pieces of information relate to one another. The outline can also include formulas for calculating certain data values from other data values.
3. **Check disk space.** Make sure you have enough disk space allocated for your database and that your index and data caches in memory are of adequate size (see Chapter 14, Sizing Your Database).
4. **Implement the security system.** Decide who should have access to the database views and calculations. To define any user and security access, you need to be a Supervisor.
5. **Load data into the database.** After you have the outline and security plan in place, you can fill the database with data.
6. **Perform calculations and consolidations.** Because the ultimate goal of the database is to do analysis, you should ensure that the proper calc scripts and consolidation paths are available for users.
7. **Optimize performance.** When you have completed steps 1 through 6, you should solicit and carefully consider the opinions of your 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.

Design Considerations

When designing a financial or budgeting 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.

Fifty percent or more of customers use Essbase for financial analysis and sales and marketing reports. Sixty percent or more use Essbase for budgeting and forecasting. It is common to use Essbase for analysis and planning and to use data warehouses for reporting.

Before you create a database and build its outline, you must create an Essbase application to house it. Typically, users create a new application for each database, although it is possible to store a database in an existing application. Applications that use the optional Currency Conversion module generally consist of a main database and a separate currency database (see Chapter 42, Designing and Building Currency Conversion Applications).

Planning and Analysis

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

The planning and analysis phase involves:

- Identifying the data you want to include in your database
- Identifying your users’ requirements
- Creating a business model for your data

The Multidimensional Analysis Life Cycle

The Multidimensional Analysis Life Cycle is a method to help you plan and implement an Essbase database. As illustrated in Figure 5-1, the cycle takes you from the planning stage through loading and consolidating your data.

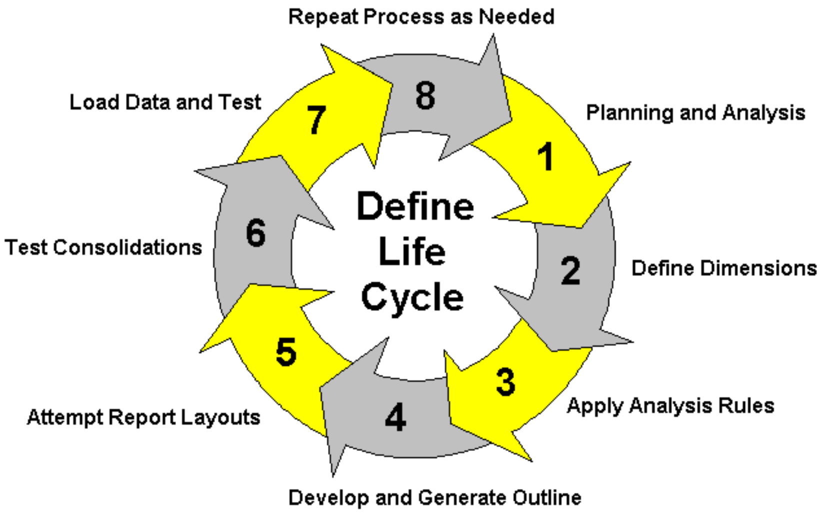


Figure 5-1, Multidimensional Analysis Life Cycle

Here is an overview of the steps illustrated in Figure 5-1:

1. **Plan and Analyze.** This is the very first stage of any application development process and includes creating an Essbase business model using your existing data sets. This process is described in this chapter.

2. **Define Dimensions.** This step involves developing a logical design of the database on paper, including the dimensions of data and the members within each dimension. For background information on dimensions and members, see Chapter 3, Multidimensional Concepts.

3. **Apply Analysis Rules.** Once you define the dimensions of your data, you identify relationships between dimensions and members, describe the current state of data in your organization and how to segment it for use in a multidimensional database, and define the reporting requirements.

4. **Develop and Generate Outline.** This step creates the database outline or structure of an Essbase database and assigns attributes and consolidations to each member in the outline. For information on creating an outline, see Chapter 8, Creating and Changing Database Outlines.

5. **Attempt Report Layouts.** In the database design process, you try different report layouts after you generate an outline. If you need to modify report contents or format, see Chapter 35, Developing Report Scripts.
6. **Test Consolidations.** Calculate data to ensure that consolidations provide the correct totals. If the results are not as you expected, you need to return to the outline and diagnose what went wrong. For information on consolidation and adjusting database calculations, see Chapter 9, Setting Dimension and Member Attributes.
7. **Load Data and Test.** This involves loading the data into the database. For detailed information about loading data, see Chapter 19, Introducing Data Loading, through Chapter 23, Debugging and Optimizing Data Loads.
8. **Repeat the Process.** After you take the primary steps, you might need to repeat one or more of them to arrive at your ideal database solution.

Case Study: The Beverage Company

This section bases the database planning process around the needs of a fictitious company called *The Beverage Company* (TBC) and provides step-by-step instructions on how you could build an Essbase database using TBC as an example. The examples follow the Sample application included with your Essbase installation.

TBC manufactures, markets, and distributes soft drink products internationally. Analysts at TBC prepare budget forecasts and track actual performance versus budgets on a monthly basis. The financial measures they track are profit and loss and inventory data.

TBC uses spreadsheet packages to prepare budget data and perform variance reporting. Since TBC plans and tracks a variety of products over several markets, the process of deriving and analyzing data is quite tedious. Last month, analysts spent most of their time entering, re-keying, 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 accessible to analysts throughout the organization. Users will have access to the server and will be able to load data from different sources and retrieve data when they need it. TBC has a variety of users, so they expect that different users will have different security levels for accessing data.

Identifying Your Source Data

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

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

Carefully define your reporting and analysis needs. Does the data support these needs? If not, what additional data do you need and where will you find this data?

Where is each department currently storing the data? Is it in a form Essbase can use? Do the departments store data in a DB2 database on an IBM mainframe, a relational database on a UNIX-based server, or in a PC-based database or spreadsheet?

Think about how often you need to update your database. Consider who will update the database and how frequently. Do the individuals who need to update the data have access to that data?

Finally, make sure that all the data you want will be ready to load in the necessary format. You can save hours of time by making sure that your 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 22, Performing a Data Load.

Checklist: Identify Your Essbase Data

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

Creating a Business Model

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

Dimensions that apply to most businesses include:

- Geography
- Business units
- Time
- Accounts
- Measures
- Distribution channels
- Products
- Scenarios

Now, choose the elements or items within a perspective. These elements will translate into the members within the dimensions in your model. For example, Scenario members might include Actual, Budget, and Variance.

Identifying the Objectives of Your Analysis

A critical step in designing an Essbase application is deciding how you want to analyze your data—by time, by geographical region, by product line, by general ledger account, by division, and so forth. Each type of analysis gives you a different view of the data. You can represent each of these views as a separate dimension in your database.

The dimensions you choose determine what types of analysis you will be able to perform on the data. Essbase lets you use as many dimensions as you need for your analysis. A typical Essbase database contains at least seven dimensions.

The structure of an Essbase database makes it easy for users to analyze information from many different perspectives. A marketing analyst, for example, might want to know:

- How were 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 were actual values to budgeted values?

In other words, the analyst might want to examine the information from three different perspectives: by time, by account, and by scenario. Because the sample database shown in Figure 5-2 has only three dimensions, you can easily picture it as a cube, with one dimension represented along each of the three axes:

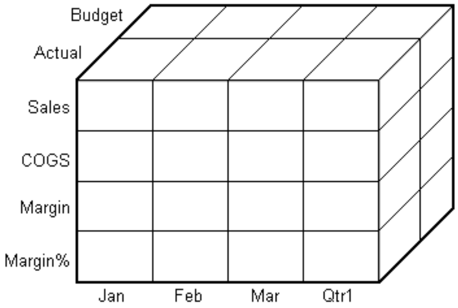


Figure 5-2, Three-Dimensional Cube Representing Three Database Dimensions

- The Accounts dimension, which consists of accounting figures such as Sales, COGS, Margin, Margin%, is displayed along the X-axis.
- The Time dimension, which consists of the individual months Jan, Feb, and Mar and the total for Qtr1, is displayed along the Y-axis.
- The Scenario dimension, which consists of Budget for budget values and Actual for actual values, is displayed along the Z-axis.

Essbase provides several special dimension types with built-in functionality. Primary among these are Time and Accounts. Users with the optional Currency Conversion module can also use the Currency and Country dimension types.

Table 5-1 shows a summary of TBC's business dimensions. 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 members in column 2.

Table 5-1, TBC Sample Dimensions

Dimensions	Members	Sub Members
Year	Qtr1	Jan, Feb, Mar
	Qtr2	Apr, May, Jun
	Qtr3	Jul, Aug, Sep
	Qtr4	Oct, Nov, Dec
Measures	Profit	Margin, Sales, COGS
	Total Expenses	Marketing, Payroll, Miscellaneous
	Inventory	Opening Inventory, Additions, Ending Inventory
	Ratios	Margin %, Profit %
Product	Colas (100)	Cola (100-10), Diet Cola (100-20), Caffeine Free Cola (100-300)
	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 %	

Security and Multi-User Planning Issues

This is a good time to think about the type of security privileges you plan to issue for the Essbase database. As an Essbase administrator, you will frequently load external data into the database. Determine who else should have privileges for these operations. End your plan with a list of users and privileges. See Chapter 16, Managing Security at Global and User Levels, to learn more about assigning user privileges.

Checklist: Create a Business Model

- What are your candidates for possible dimensions?
- What are candidates for possible members?
- How many levels does your data require?
- How will your data consolidate?
- Who will have load data privileges?
- Who are your users and what privileges will they have?

Drafting an Outline

At this point, you will find it useful to take all your work papers and create a first draft outline. The draft defines all dimensions, members, consolidations, and calculations. Use the outline to design consolidation requirements and identify exceptions to default calculations before you proceed to creating calculation scripts. Determine what items should be dimensions and whether they are best represented by one or two dimensions.

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

- **By Year**—TBC needed 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 quarterly results. Quarterly data, stored in members such as Qtr1 and Qtr2, consolidates into total Year.

- **By Measures**—Sales, Cost of Goods Sold, Marketing, Payroll, Miscellaneous, Opening Inventory, Additions, and Ending Inventory data are standard measures. Essbase calculates Margin, Total Expenses, Profit, Total Inventory, Profit %, and Margin % from these measures. Specifically, TBC needs to calculate Measures on a monthly, quarterly, and yearly basis.
- **By 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.

  TBC grouped each of the soda products into a product family. Families consolidate into total Product. Some products must consolidate in multiple directions. For example, all diet soft drink products consolidate into their flavor or Diet family.
- **By 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 total Market.
- **By Scenario**—TBC derives and tracks budget data versus actuals. Managers must monitor and track budgets and actuals, plus the variance and variance percentage between them.

Building the Outline

In this section, you will learn how TBC built its database outline. TBC first grouped elements within dimensions and narrowed down the elements within the dimensions. Then, as described in this section, they defined all of the outline elements.

When you define the database outline, each dimension and member play a role in the design of the multidimensional structure. Essbase provides qualifiers that you can assign to members to access the built-in functionality of Essbase including:

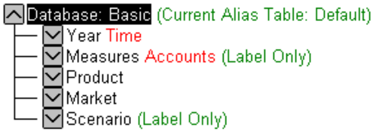
- Generation and level names
- Aliases
- Dimension types
- Attributes
- Field types
- Consolidation
- Formulas

We'll introduce each of these terms and how to use them as we build each section of the TBC outline.

Defining Dimension Attributes

Let’s take a close look at the TBC database dimensions. Notice that some of the dimensions have special tags. Essbase calls these tags *attributes*.

As shown in Figure 5-3, the Year dimension has a Time tag, the Measures dimension has an Accounts tag and a Label Only tag, and Scenario has a Label Only tag.



Database: Basic (Current Alias Table: Default)
Year Time
Measures Accounts (Label Only)
Product
Market
Scenario (Label Only)

Figure 5-3, TBC Dimensions and Related Attributes

Essbase provides dimension types that add functionality to the dimension. The most commonly used dimension types are Time and Accounts. Table 5-2 defines each of the Essbase dimension types.

Table 5-2, 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 only tag one dimension as Time, although you do not need to have a Time dimension. The Time dimension enables several Accounts dimension functions, such as First and Last Time Balances.
Accounts	Contains items you want to measure, such as profit and inventory, and makes the Essbase built-in accounting functionality available. (For example, see “Accounts Dimension Calculation” on page 5-19.) Only one dimension can be defined as Accounts, although you do not need to have an Accounts dimension.

Table 5-2, Dimension Types

Dimension Types	Description
Country	Contains data about where your 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 your application. This dimension type is used only in the main database and is used for currency conversion applications. If your base currency for analysis is US dollars, the local currency members would contain values based on the currency type of the region.

Essbase lets you specify data storage attributes for dimensions and members. By default, Essbase adds members and stores the result at the parent level. You can change this default logic by using attribute tags that alter the way data is consolidated. Members with the Label Only tag, for example, do not have data associated with them. Some member names exist only for purposes of data grouping or navigation. Members of this type contain no data, and cannot be consolidated.

For example, TBC planners did not combine the two financial dimensions, Measures and Scenario, because the members provide discrete information that does not logically consolidate. In the Measures dimension, for example, the member Ratios has two children, Margin% and Profit%. When consolidated, Margin% and Profit% do not add up to Ratios.

Table 5-3 shows Essbase data storage attributes.

Table 5-3, Essbase Data Storage Attributes

Attributes	Effects on Members
Store Data	This member stores data. It is the default value.
Never Share	The data associated with this member will be duplicated with its parent or child if an implied shared relationship exists.
Label Only	This member displays the value of its first child. A Label Only member has no data associated with it, although it can still display a value. This tag groups members or eases navigation and reporting from the Spreadsheet Client. Typically, Label Only members have a No Consolidation attribute. (See “Ordering Consolidations” on page 5-16.)

Checklist: Define Dimension and Member Attributes

- Can you identify a Time dimension or an Accounts dimension?
- Does your data include foreign currencies? If so, did you identify a Currency Partition dimension?
- Which dimensions require special data storage attributes?
- What members require special attributes?

For a complete discussion of this topic, see Chapter 9, Setting Dimension and Member Attributes.

Defining 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 found in many Essbase applications. You will learn about:

- Consolidation paths
- Consolidation order
- Time and Accounts dimension calculations
- Formulas

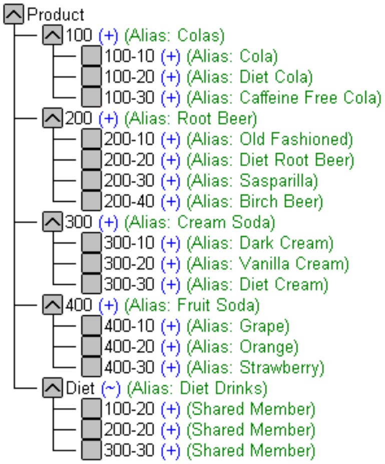
Defining Consolidation Paths

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

Unary 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 results of that member to other members in the branch. For example, it will add Cola, Diet Cola, and Caffeine Cola and store the result in their parent, Colas, as shown in Figure 5-4.



```
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)
```

Figure 5-4, TBC Product Dimension

The Product dimension contains mostly + operators, which indicate that each group of members are added together and rolled up to the parent. Diet has a tilde (~), which indicates that Essbase won't use it to roll up to the parent, Product. The Diet alias 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 was able to create a separate Diet member without impacting the overall consolidation.

For more information on consolidation and calculations, see Chapter 9, Setting Dimension and Member Attributes.

Ordering Consolidations

Essbase calculates the data in a branch in top-down order. For example, if you have two members tagged with a + and a third member tagged with a *, in this order, Essbase adds the first two and multiplies the result by the third. Be aware that Essbase always begins with the topmost member when it consolidates, so the order of your members and how you label them is very important. Table 5-4 shows the Essbase unary operators. See Chapter 9, Setting Dimension and Member Attributes, for more information.

Table 5-4, Unary Operations

Operator	Description
+	The default operator. When a member has a + operator, Essbase adds that member to the result of previous calculations performed on other members.
–	When a member has a - operator, Essbase multiplies the member by -1 and then adds it to the result of previous calculations performed on other members.
*	When a member has a * operator, Essbase multiplies the member by the result of previous calculations performed on other members.
/	When a member has a / operator, Essbase divides the member into the result of previous calculations performed on other members.
%	When a member has a % operator, the member is divided into the sum of previous calculations performed on other members. The result is multiplied by 100.
~	When a member has a ~ 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 share the same data. The shared member stores a pointer to data contained in another 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.

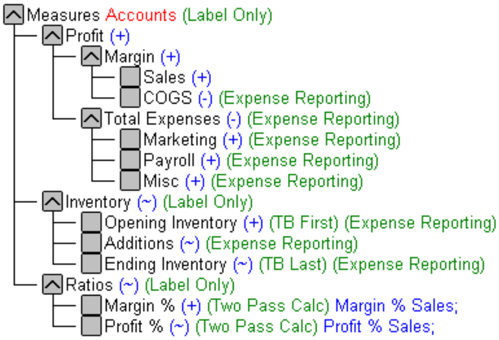
Checklist: Define Consolidations

- Have you identified the consolidations in your outline?
- Did you tag each member with the proper unary operator?
- Did you specify a shared member tag for designated members?

Time and Accounts Calculations

The Measures dimension is the most complex one in the TBC outline because it uses both Time and Accounts data. It might also contain several formulas and special tags to help Essbase calculate the outline. This section discusses the formulas and tags TBC included in the Measures dimension.

Take a moment to look closely at the Measures dimension tags defined by TBC in Figure 5-5. You will see that you already know about many of the attributes in the Measures dimension. You've learned about positive (+), negative (–), and tilde (~) unary operators, as well as Accounts and Label Only tags. 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.



```
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)
      Opening Inventory (+) (TB First) (Expense Reporting)
      Additions (~) (Expense Reporting)
      Ending Inventory (~) (TB Last) (Expense Reporting)
   Ratios (~) (Label Only)
      Margin % (+) (Two Pass Calc) Margin % Sales;
      Profit % (~) (Two Pass Calc) Profit % Sales;
```

Figure 5-5, TBC Measures Dimension

For details on Essbase calculations, see Chapter 24, Introduction to Database Calculations, through Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

Formulas

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

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

The operators include:

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

The Measures dimension uses the following equations:

Margin = Sales - COGS
Total Expenses = Marketing + Payroll + Miscellaneous
Profit = Margin - Total Expenses
Profit % = Profit % Sales
Margin % = Margin % Sales

Essbase calculates the Margin, Total Expenses, and Profit dimensions using unary operators. The Margin% formula uses a % operator, which means "express Margin as a percentage of Sales." The Profit% formula uses the same % operator.

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

- Mathematical functions, which perform specialized calculations
- Index functions, which look up data values within a database during a calculation
- Financial functions, which perform specialized financial calculations
- Macro functions, which generate lists of members based on a specified member
- Boolean functions, which provide a conditional test by returning a True or False value

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

Accounts Dimension Calculation

In this section, we'll discuss two forms of accounts dimension calculations: time balance and variance reporting.

Time Balance Tags

Note the three tags in the Measures dimension: TB First, TB Last, and TB Average. These tags, called time balance tags, provide instructions to Essbase about how to calculate your 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 exclusively for use with Accounts dimension members.

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

Ending Inventory data represents the inventory that TBC carries at the end of each month. The quarterly value is equal to the ending value for the quarter. Ending Inventory requires a time balance of TB Last. Table 5-5 shows Accounts dimension time balance attributes.

Table 5-5, Accounts Member Tags

Tags	Description
Time Balance Last	The value for the last child member is carried to the next level. For example, March is consolidated up to Qtr1.
Time Balance First	The value for the first child is carried to the parent. For example, the value for Jan is consolidated up to Qtr1.
Average	The Average value of the children is carried to the parent. For example, the average value of Jan, Feb, and Mar are consolidated up to Qtr1.

Table 5-6 shows how consolidation in the Time dimension is affected by Time Balance attributes in the Accounts dimension.

Table 5-6, TBC Consolidations Affected by Time Balance Attributes

Account Member	Jan	Feb	Mar	Qtr1	...	Year
Member1	11	12	13	**36**		**Qtr1 + Qtr2 + Qtr3 + Qtr4**
Member2 (**First**)	20	25	21	**20**		**20**
Member3 (**Last**)	25	21	30	**30**		**Value of Qtr4**
Member4 (**Avg**)	20	30	28	**26**		**Avg of Qtr1, Qtr2, Qtr3, Qtr4**

Normally, a parent in the Time dimension is calculated based on the consolidation and formulas of its children. However, if a member in an Accounts branch is marked as First, then any parent in the Time dimension will match the member marked as First.

Variance Reporting

One of TBC's Essbase requirements was the ability to perform variance reporting on Actual versus Budget data. The variance performance 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 attributes: Expense and Non Expense. The default is Non Expense. Variance reporting attributes define how Essbase calculates the difference between actual and budget data in members with the @VAR or @VARPER function in the member formula.

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

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

Two Pass Calculations

Both Margin% and Profit% contain the label “Two Pass Calc.” This is a default label indicating that some member formulas need to be calculated twice to produce the desired value. You can only tag Accounts members as Two Pass. The following examples illustrate why Profit% has a Two Pass Calc.

Essbase loads data into the system as follows:

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

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

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

Next, Essbase calculates the Time 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% (Two Pass Calc) Profit%Sales	10%	10%	10%	30%

The result in Profit% ->Qtr1 of 30% is not correct. Because TBC tagged Profit% as Two Pass Calc, Essbase recalculates profit percent at each occurrence of the member Profit%. The data is then correct and appears as follows:

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

Checklist: Define Your Calculations

- Which dimensions require formulas or calculations?
- Does the default calculation logic achieve accurate results?
- What type of calculations do they require?
- Which members require variance reporting?
- Do members that require variance reporting have a Time or Accounts tag?

Analyzing Your Database Scope

This section includes a set of rules you can use to analyze your Essbase database. Use these rules to fine tune your database to leverage the multidimensional technology. These rules will help you to achieve an efficient design and meet your consolidation and calculation goals.

The number of members that you need to describe a potential data point should determine the number of dimensions. If you are not sure whether to delete a dimension, keep it and apply more analysis rules until you feel confident about deleting or keeping it.

Multidimensional Database Analysis Example

A company sells products to multiple customers over multiple markets. The customers are unique to each market:

```
                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. Therefore, the company could fold Customers into the Market dimension, unless reporting needs dictate a two dimensional layout. See Chapter 34, Quick Start to Report Scripts and Chapter 35, Developing Report Scripts, to learn about reports and creating report scripts. See your spreadsheet *User's Guide* for information about spreadsheet retrievals.

Except for reporting requirements, the company would typically define the data in one dimension:

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

Rule 1: Examine dimension combinations.

Break each combination of two dimensions into a two-dimensional matrix. Once you have more experience, you can examine only those dimensional combinations where additional complexity exists. These combinations vary, depending on the complexity of your business model. Typical combinations you will examine include:

- Total Year vs. Year
- Total Year vs. Accounts
- Total Year vs. Employees
- Total Year vs. Product
- Total Year vs. Accounts
- Year vs. Employees
- Year vs. Region

Analyzing combinations will help you to understand the most frequently used intersections and eliminate dimensions that don't intersect with other dimensions. You will be able to see which dimensions you should keep and those you should delete.

Rule 2: 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, ask these questions to determine if it is a valid combination for the database. Ideally, the combination should add analytic value, add utility for reporting, and should have few unused combinations. From this exercise, you can determine if you should eliminate or add dimensions.

Rule 3: Avoid repetition in the outline.

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

Repetition

```
Budget Profit
  Budget Margin
     Budget Sales
     Budget COGS
  Budget Expenses
Actual Profit
  Actual Margin
     Actual Sales
     Actual COGS
  Actual Expenses
```

No Repetition

```
Accounts
     Profit
         Margin
            Sales
            COGS
         Expenses
Scenario
     Budget
     Actual
```

Rule 4: 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 will store only at the summary (dimension) level. In this situation, either combine the dimension with another or split the model into separate databases.

Salary databases often prove to be irrelevant in the context of a corporate database. Most salaries are confidential and apply to only one individual. The individual and salary typically represent one cell, with no reason to intersect with any other dimension. By adding an employee dimension, the database would risk the problem of naming a supposedly sparse dimension dense. The dimensions would take unnecessary storage space. Table 5-7 shows an example of interdimensional irrelevance.

Table 5-7, Interdimensional Irrelevance Example

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

Rule 5: Split databases if necessary.

TBC agreed that the salary database was irrelevant, and also placed a restriction on who had the authority to see the salary data. Rather than keep the salary data in a database where the surrounding dimensions are irrelevant, TBC decided to split the database to create a Human Resources (HR) database. The new HR database will contain a group of related dimensions including salaries, benefits, insurance, and 401(k) plans.

There are many other reasons for splitting a database in addition to interdimensional irrelevance. For example, your company maintains an organizational database containing several international subsidiaries in different time zones. Each subsidiary relies on time-sensitive financial calculations. You might want to split the database for groups of subsidiaries in the same time zone to ensure that their financial calculations are timely.

Checklist: Analyze Your Database

- Have you minimized your 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 if necessary?
- Did you examine your consolidations and adjust as necessary?

Testing the Database

Once you have analyzed your data, you need to load the data to see if the database satisfies your analysis and reporting needs. Are your reports in the correct format? Are the calculations correct? Did all the data load? If the database fails in any area, return to the Design Life Cycle steps to identify where you might have gone wrong. Essbase provides many utilities and screens to help you isolate problems. See Chapter 7, Creating Applications and Databases, for step-by-step instructions on creating the database.

Chapter 6 Designing Partitioned Applications

A Hyperion Essbase OLAP Server 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:

Warning: You must design partitions carefully. We strongly recommend that you read this chapter before creating partitions.

Introducing Hyperion Essbase Partitioning

Hyperion Essbase Partitioning is a collection of features that makes it easy to design and administer databases that span Essbase applications or servers. You must license Hyperion Essbase Partitioning separately from Hyperion Essbase OLAP Server. You must license the Hyperion Essbase 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 your 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 your 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.

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.

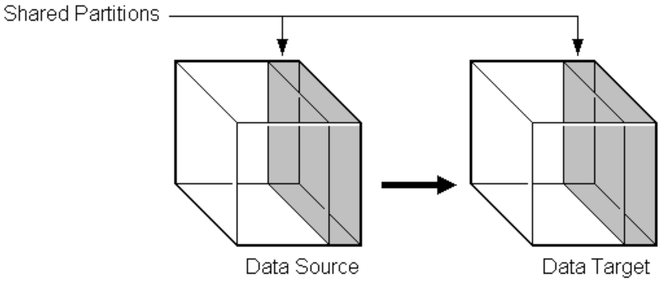


Figure 6-1, Data Source and Data Target

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 can’t 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.

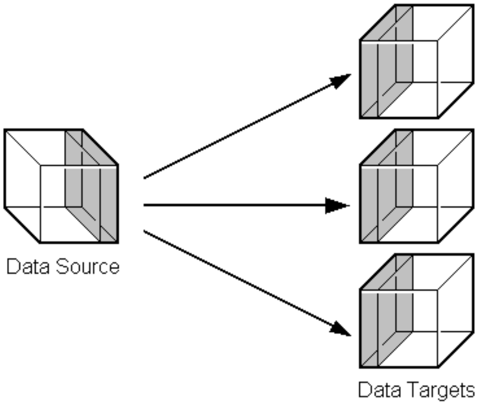


Figure 6-2, Data Shared at Multiple Targets

What Is a Partition?

A *partition* is the piece of a database that is shared with another database. Partitions contain the following parts:

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.

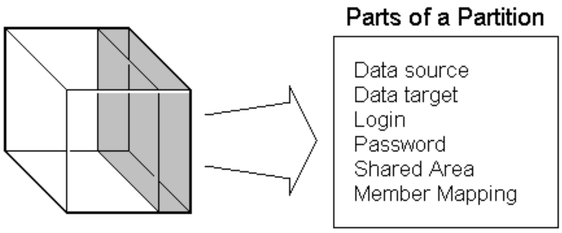


Figure 6-3, Parts of a Partition

Workflow for Partitioning a Database

Here is the suggested workflow for partitioning a database.

1. Design the partitioned database(s). See “Determining Which Data to Partition” on page 6-7 and “Deciding Which Type of Partition to Use” on page 6-8.
2. Edit existing applications that use the individual database(s).
   - Edit the outline(s) and existing calc 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 6-22.
3. Create the partition(s). See Chapter 15, Building and Maintaining Partitions.
4. Load data into the partition(s). Load the data into the new database(s). See Chapter 19, Introducing Data Loading.
5. If necessary, calculate the new database(s). See Chapter 24, Introduction to Database Calculations.
6. Test and maintain the partition(s). Test your partitioned databases, including updates, data loads, calc scripts, report scripts, and database restructures. For more information on maintaining partitioned databases, see Chapter 15, 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 database(s) 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? Would you like to speed it up 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 6-24.

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 6-8.
- 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 Essbase server. For more information, see “Transparent Partitions” on page 6-13.
- 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 6-18.

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 6-25.

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 takes precedence over access provided by security filters and is also honored by batch operations such as dataload and calc. 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 15-16.

Rules for Replicated Partitions

Replicated partitions must follow these rules:

- The shared replicated areas of the data source and data target outlines do not have to be identical, but you must be able to map 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.
- 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.

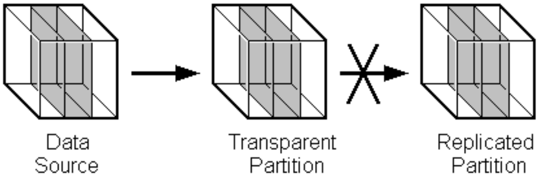


Figure 6-4, Invalid Replicated Partition

- 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.
- For more information on using Dynamic Time Series members in replicated partitions, see “Calculating Time Series Data in Application Partitions” on page 29-14.

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 aren't 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 can't calculate derived data at the data source if the data:

  - Requires data outside the replicated area to be calculated.
  - Requires calc 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.

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.

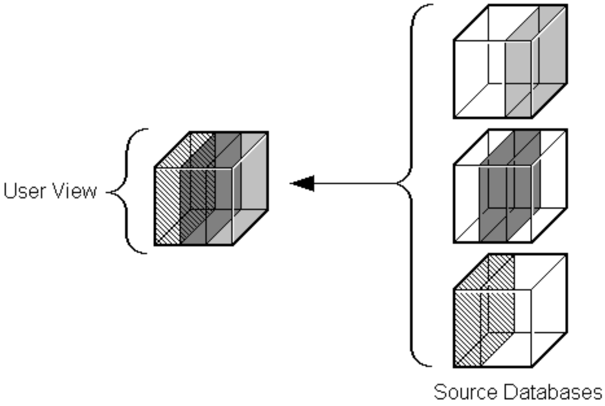


Figure 6-5, Transparent Partitions

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 6-25.

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.
- 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.

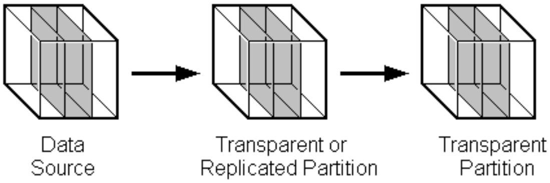


Figure 6-6, Valid Transparent Partition

- 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.

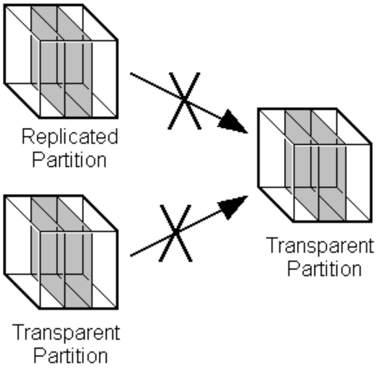


Figure 6-7, Invalid Transparent Partition

- For more information on using Dynamic Time Series members in transparent partitions, see “Calculating Time Series Data in Application Partitions” on page 29-14.

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:

  a. Issue a CALC ALL command for each individual partition.

  b. 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.

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 rules 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 will need to 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.

- Calculating data on the data target can greatly slow performance because:
  - Essbase must probe the outline from the top-down to check for changes made to dependents of the member that you are calculating. When Essbase performs the calculation on the data source, it can perform a bottom-up calculation, which is much faster.
  - Essbase must retrieve each dependent data block across the network and then perform the calculation.

  Consider using:

  - 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 calc. Essbase performs the calculation only when users request it.
  - A replicated layer between the low-level transparent data and high-level local data.
- 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.

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.

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 the Hyperion Essbase 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 6-22.

There are no performance considerations for linked partitions, beyond optimizing the performance of each linked database.

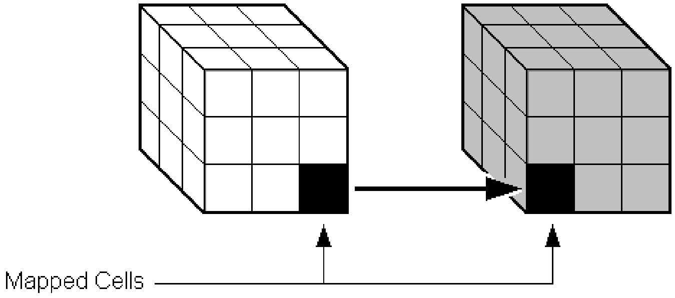


Figure 6-8, Linked Partition

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 6-29.

A user could:

- 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.

Note: 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 source to the data target.

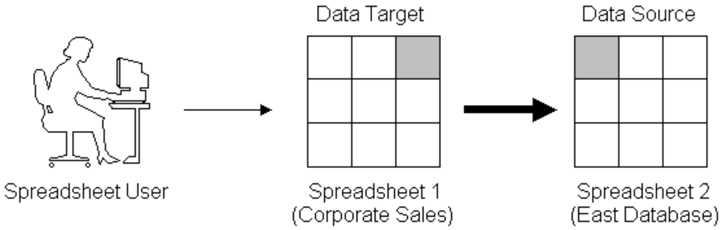


Figure 6-9, Source and Target for Linked Partition

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 6-22.
- Users must access the linked database using Essbase Version 5-aware tools. If you have custom built your tools, you must extend them using the Essbase Version 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 6-1 should help you choose which type of partition to use.

Table 6-1, Types of Partition

If the answer to this question is yes...	Use this partition type...
Is decreasing network activity more important than increasing disk space?	Replicated
Is it acceptable for users to access data that's not up to the minute?	Replicated
Are you concerned about a single failure disrupting an entire OLAP application?	Replicated
Do you do batch updates and simple aggregations?	Replicated
Must you access the data using front-end tools that are not Distributed OLAP-aware?	Replicated or Transparent
Do you perform frequent updates and calculations?	Transparent
Does the user have to update the data at the data target?	Transparent
Is decreasing disk space more important than increasing network activity?	Transparent or Linked
Is it important that the user access the absolute latest version of the data?	Transparent or Linked
Does the user want to view data in different application contexts?	Linked

After you have answered the questions in Table 6-1, you can compare the partition types in the following table.

Feature	Replicated	Transparent	Linked
Up-to-the-minute data		■	■
Reduced network traffic	■		■
Reduced disk space		■	■
Increased calculation speed	■		
Smaller databases		■	■
Improved query speed	■		■

Feature	Replicated	Transparent	Linked
Invisible to end users	■	■	
Access to databases with different dimensionality			■
Easier to recover	■		
Less synchronization required			■

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 16, Managing Security at Global and User Levels.
- 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 15-4.

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 15-6.
- 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 16, Managing Security at Global and User Levels.

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 15-6.
- 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 16-8.
- Create a default account that users can use when accessing target database(s). 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've 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 15-22.

For information on creating user accounts and filters, see Chapter 16, Managing Security at Global and User Levels.

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've 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.

Note: 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.

Here are the steps to take:

1. Determine which data to partition.

   TBC's Sample Basic database contains five 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.

Members to partition	Data source	Data target	Reasons for choosing
Corp_Budget	Company	East	Because the company owns the corporate budget, it is the source.
Eastern Region, Actual	East	Company	Because the eastern region needs to update its actual and market information, it is the source.

Figure 6-10 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.

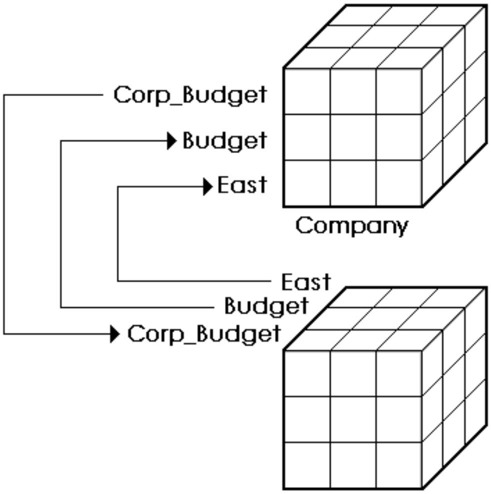


Figure 6-10, Decentralizing a Database

3. Decide the type of partition to use.

Members	Partition type	Reasons
East	Transparent	The data target (Company) needs up-to-the-minute data
Corp_Budget	Transparent	The data target (East) needs up-to-the minute data
East Actual	Replication	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, calc 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’s 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 couldn’t 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.
   - Don't 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 steps 1–3 for each database.
5. Finally, create the partitioned databases by:
   - Editing existing data sources, rules files, calc 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's 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's considered the data target. Likewise, because users move to TBC Demo, it's 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 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 by:
   - Establishing 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, that is, Channel and Product, to `void` in the Sample Basic database. For more information, see "Mapping Data Source Members to Data Target Members" on page 15-10.
   - 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 "Managing Security at the User and Group Layer" on page 16-4. For more information on assigning those accounts to linked partitions, see "Specifying the Partition Type and Connection Information" on page 15-22.

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 is still works.

Part II

Building Hyperion Essbase Applications

Part II describes how to create single server and partitioned Hyperion Essbase OLAP Server applications using the Hyperion Essbase Application Manager or external data sources and rules files. Part II contains the following chapters:

- Chapter 7, Creating Applications and Databases, describes how to create and manage applications and databases.
- Chapter 8, Creating and Changing Database Outlines, describes how to create and modify database outlines using the Application Manager.
- Chapter 9, Setting Dimension and Member Attributes, illustrates how to set attributes for the dimensions and members in an outline 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 Your Essbase Data, describes how to link various kinds of data with any cell in an Essbase database.
- Chapter 12, 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 13, 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 14, Sizing Your Database, describes how to estimate disk and memory requirements and specify settings so that you can allocate the right amount of space to run Essbase efficiently.
- Chapter 15, 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 7 Creating Applications and Databases

A Hyperion Essbase application is a container for a database and its related files. In addition to the database, an application can include scripts that are used to load data into the database, calculate derived values, and prepare reports. This chapter explains how to work with Hyperion Essbase applications and databases. For information on copying, renaming and deleting applications, databases and their associated files, see Chapter 46, Working with Essbase Files and Cross-Platform Environments.

This chapter includes the following sections:

Essbase Applications

The Hyperion Essbase Application Manager provides a Windows-based environment where you can create and maintain Essbase applications. Application development includes building outlines and dimensions, performing data loading and calculations, and defining security access.

An Essbase application is a collection of one or more of the following types of files:

- Multidimensional databases
- Rules for loading data
- Scripts to calculate your data
- Scripts to generate reports on your data
- Security information and application security logs

The application and its parts usually reside on the server machine, but pieces of it can reside on your client machine. The server machine can store multiple applications. The following diagram shows the relationship between the parts of an application:

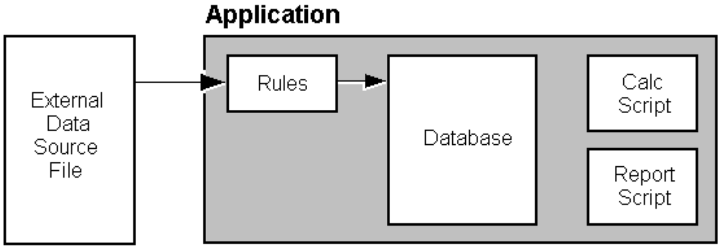


Figure 7-1, Parts of an Essbase Application

When you start an application, the following actions can happen:

- Users can connect to the application.
- The application can respond to commands from the server.
- Users can change the settings of the application.
- Data and user security are enabled.
- Each database in the application can start.

For information on how to start an application, see “Starting and Stopping Applications and Databases” on page 7-4.

Multidimensional Database Outlines

Database outlines define the structure of a multidimensional database, including all the dimensions, members, tags, types, consolidations, and mathematical relationships. Data is stored in the database according to the structure defined in the outline. See Chapter 4, Basic Architectural Elements, for information on creating database outlines.

Data Load and Dimension Build Rules

Data load rules 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. You need to load data into an Essbase database because it contains no data when you create a database for the first time. 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 can also load data into the database, and then modify the database outline based on data in the external data source file.

See Chapter 19, Introducing Data Loading, for information on creating data load rules.

See Chapter 12, Introducing Dynamic Dimension Building, and Chapter 13, Building Dimensions Using a Rules File, for information on creating dimension build rules.

Calc Scripts

Calc scripts are text files that contain instructions to calculate data in the database. Calc (calculation) scripts perform different calculations than the consolidations and mathematical operations that you define in the database outline. Because calc scripts perform specific mathematical operations on members, they are typically associated with a particular database. You can, however, define a calc script for use with multiple databases. See Chapter 30, Developing Calc Scripts, for information on creating calc 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. See Chapter 35, Developing Report Scripts, for information on creating report scripts.

Security Definitions

With the Application Manager, you also maintain user information, security information, application activity logs, and server activity logs. This information is saved on the Essbase server. Within your organization, you can have multiple servers, each with its own users, applications, and databases. See Chapter 16, Managing Security at Global and User Levels, for more information about setting up and maintaining security information.

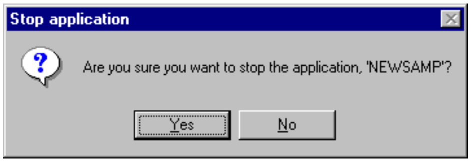
Process for Creating Databases

To implement a multidimensional database you need to complete a number of steps, from creating an application through optimizing your database performance. For detailed information on these steps, see “Design Considerations” on page 5-3.

Starting and Stopping Applications and Databases

To a start or stop an application:

1. Connect to the server on which the application resides.
2. Select the application from the Application Desktop window.
3. Choose Application | Start/Stop. If you are stopping the application, a confirmation dialog box appears:



Are you sure you want to stop the application, 'NEWSAMP'?

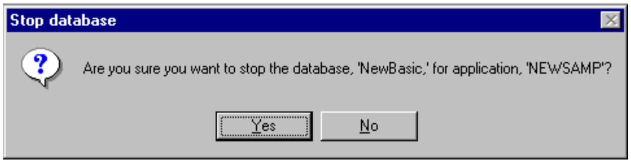
Figure 7-2, Stop Application Dialog Box

4. Click OK to stop the application. Essbase unloads the application from memory.

You can use the LOADAPP command in ESSCMD to start an application, and the UNLOADAPP command to stop an application. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

To a start or stop a database:

1. Select the database in the Application Desktop window.
2. Choose Database | Start/Stop. If you are stopping the database, a confirmation dialog box appears:



Stop database

Are you sure you want to stop the database, 'NewBasic,' for application, 'NEWSAMP'?

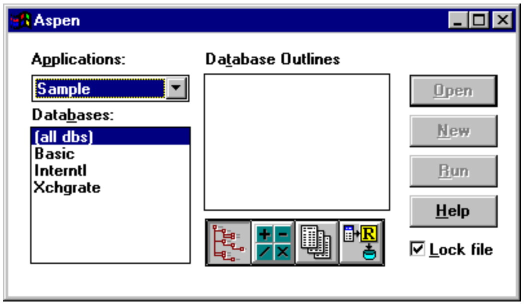
Figure 7-3, Stop Database Dialog Box

3. Click Yes to stop the database. Essbase unloads the database from memory.

You can use the LOADDB command in ESSCMD to start a database, and the UNLOADDB command to stop a database. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Using the Application Desktop Window

When you start the Application Manager, an icon appears at the bottom of the Application Manager window. This is the Application Desktop window for client-based applications. When you connect to an Essbase server, a window appears that lists all the applications on the server. This is the Application Desktop window for server-based applications.



Aspen

Applications: Sample

Databases:
(all dbs)
Basic
Interntl
Xchgrate

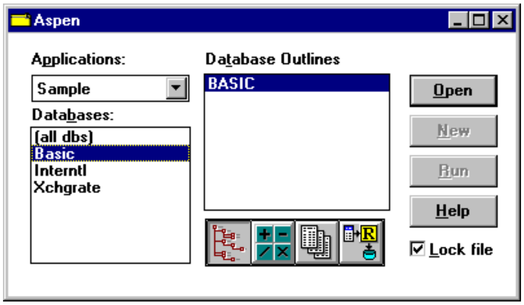
Database Outlines

☑ Lock file

Figure 7-4, Application Desktop Window

Before you can work with database outlines, calc scripts, report scripts, data loading or dimension building rules, you must select an existing application or create a new one.

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 appears as follows:



Applications: Sample

Databases:
(all dbs)
Basic
Interntl
Xchgrate

Database Outlines
BASIC

☑ Lock file

Figure 7-5, Application Desktop Window with Database Selected

When you select an application and database from the list boxes, choose one of the following buttons to work with outlines, scripts, and 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 calc 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.

- To open one of these files, select it from the list, and then choose **Open**.
- To create a new calc 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 7-9.
- To run a calc script or report script, select the file from the list, and then choose **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.

Deciding Where to Build an Application

Before creating an application and database, you should decide:

- Whether to build the application and database on the Essbase 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 an Essbase 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.
  - Start the application automatically when the server starts.
- Whether to copy an existing application and database or to create a new one from scratch.
- What tools to use—Application Manager’s graphical user interface or ESSCMD commands.

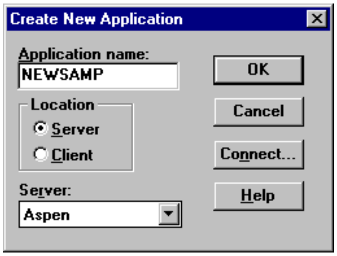
Creating Applications and Databases

You can use either Application Manager or ESSCMD to create an new application or database.

Creating a New Application

To create a new application with Application Manager:

1. Choose File | New | Application. The following dialog box appears:



Create New Application

Application name: NEWSAMP

Location

● Server

○ Client

Server: Aspen

Figure 7-6, Create New Application Dialog 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. Choose 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 within the `essbase\client` directory, where `essbase` is the drive and directory into which you installed your Essbase client. The new subdirectory has the same name as the application.
   - To create a server-based application, click **Server** and choose 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.

You can use the CREATEAPP command in ESSCMD to perform this task. See the online *Technical Reference* in your DOCS directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

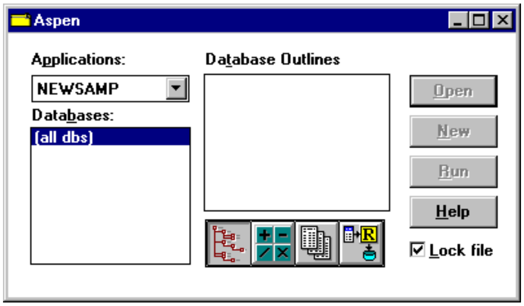
Creating a New Database

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 appears:



Applications: NEWSAMP

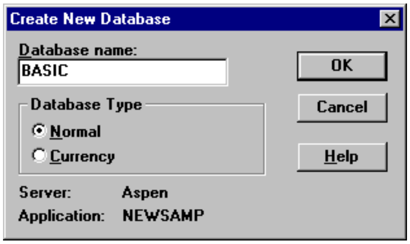
Databases: (all dbs)

☑ Lock file

Figure 7-7, Application Desktop Window

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

3. Choose File | New | Database. The following dialog box appears:



Database name: BASIC

Database Type

● Normal

○ Currency

Server: Aspen

Application: NEWSAMP

Figure 7-8, Create New Database Dialog Box

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

5. In most cases, the **Database Type** is Normal. To create a currency database, choose a **Database Type** of Currency. For more information on currency databases, see Chapter 42, Designing and Building Currency Conversion Applications.

6. 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 7-8. The new subdirectory has the same name as the application.

You can use the CREATEDB command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Annotating a Database

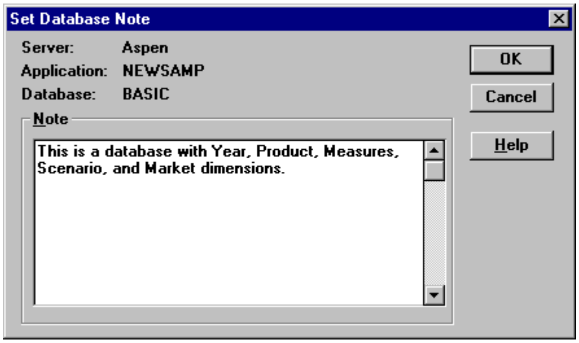
When you have created a database, you should annotate it. A database note can provide useful information in situations where you need to broadcast messages to users about the status of a database, deadlines for updates, and so on.

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

To annotate a database:

1. Select the database you want to annotate from the Application Desktop window.

2. Choose Database | Set Note. The following dialog box appears:



Server: Aspen
Application: NEWSAMP
Database: BASIC
Note
This is a database with Year, Product, Measures, Scenario, and Market dimensions.

Figure 7-9, Set Database Note Dialog Box

3. Type some text in the **Note** text box.

4. Click OK.

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 might 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 then be changed at any time by the database designer, reducing manual changes to a report script.

For example, many reports depend on reporting periods; if you generate a report based on the current month, you 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. By using the variable name in your report script the information is dynamically updated when you run the final report.

You can use substitution variables in calc scripts, report scripts, or in the Essbase Spreadsheet Add-in. For information about using substitution variables in:

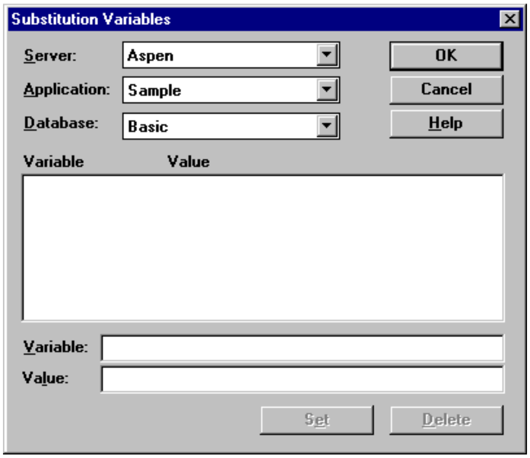
- Calc scripts, see Chapter 30, Developing Calc Scripts.
- Report Writer, see Chapter 35, Developing Report Scripts.
- Essbase Spreadsheet Add-in, see the *Spreadsheet Add-in User's Guide*.

You can set substitution variables on the server using either Application Manager or ESSCMD. Set the variable at any of 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 variable within the specified database.

Setting a Substitution Variable

1. From the Application Desktop window in Application Manager, choose Server | Substitution Variables. The **Substitution Variables** dialog box appears:



Server: Aspen

Application: Sample

Database: Basic

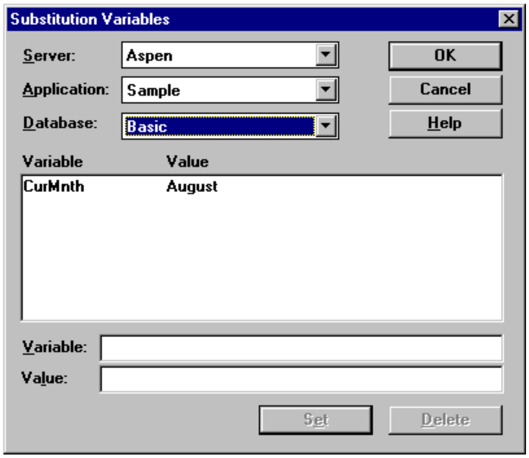
Figure 7-10, Creating a Substitution Variable

2. Choose the server to apply the variable to in the **Server** list box.
3. Choose the application to apply the variable to in the **Application** list box. Choose **(all apps)** to apply the variable to all applications on the server.
4. Choose the database to apply the variable to in the **Database** list box. Choose **(all dbs)** to apply the variable to all databases in the selected application.
5. Type the new variable name in the **Variable** box. The substitution variable must be composed of alphanumeric characters or underscores (_), and cannot exceed 80 characters.
6. Type the value name in the **Value** box. The value name cannot exceed 256 characters. You can use any combination of characters in the value name.
7. Click **Set** to apply the new variable and value.

You can use the CREATEVARIABLE command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Deleting a Substitution Variable

1. From the Application Desktop window in Application Manager, choose Server | Substitution Variables. The **Substitution Variables** dialog box appears:



Substitution Variables

Server: Aspen

Application: Sample

Database: Basic

Variable	Value
CurMnth	August

Variable:

Value:

Figure 7-11, Deleting a Substitution Variable

2. Choose the server to delete the variable from in the **Server** list box.
3. Choose the application to delete the variable from in the **Application** list box. Choose **(all apps)** to delete the variable from all applications on the server, if that is the level at which it was applied.
4. Choose the database to delete the variable from in the **Database** list box. Choose **(all dbs)** to delete the variable from all databases in the selected application, if that is the level at which it was applied.
5. Click on the variable in the list.
6. Click **Delete** to delete the variable and value.

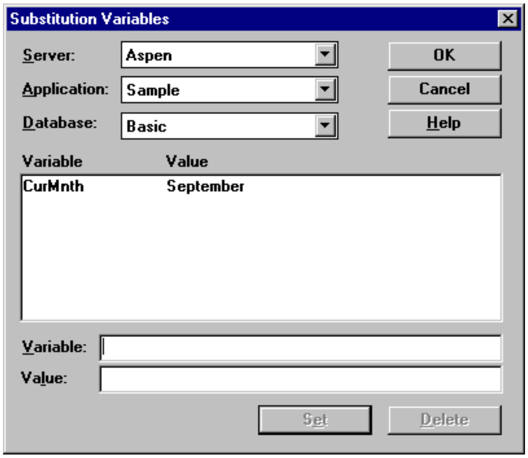
You can use the DELETEVARIABLE command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Updating a Substitution Variable

1. From the Application Desktop window in Application Manager, choose Server | Substitution Variables.
2. Choose the server where the variable is set in the **Server** list box.
3. Choose the application where the variable is set in the **Application** list box. If the variable applies to all applications on the server, choose **(all apps)**.
4. Choose the database where the variable is set in the **Database** list box. If the variable applies to all databases in the server, choose **(all dbs)**.

   The name of the existing variable and its current value appear in the list.
5. Type the name of the existing variable in the **Variable** box, exactly as it appears in the list.
6. Type the value name in the **Value** box. The value name cannot exceed 256 characters. You can use any combination of characters in the value name.
7. Click **Set** to reapply the existing variable with its new value.

   The existing value is replaced in the list with the new value that you set.



Substitution Variables

Server: Aspen

Application: Sample

Database: Basic

Variable	Value
CurMnth	September

Variable:

Value:

Figure 7-12, Updating a Substitution Variable

You can use the UPDATEVARIABLE command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

You can use the LISTVARIABLES command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Chapter 8 Creating and Changing Database Outlines

A database outline is made up of dimensions and members organized hierachically. The Outline Editor provides a graphical representation of the dimension hierarchy. Each dimension and consolidated level in a database is represented 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 background information on outlines, see Chapter 3, Multidimensional Concepts.

This chapter explains how to create a Hyperion Essbase OLAP Server database outline using the Outline Editor. You can also change outlines using data sources and rules files. For more information, see Chapter 12, 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 information on setting attributes, see Chapter 9, Setting Dimension and Member Attributes.

Creating Outlines

The database outline defines the structure of the database. For more information about outlines, see “Arranging Dimensions into Hierarchies” on page 3-3.

To create an outline:

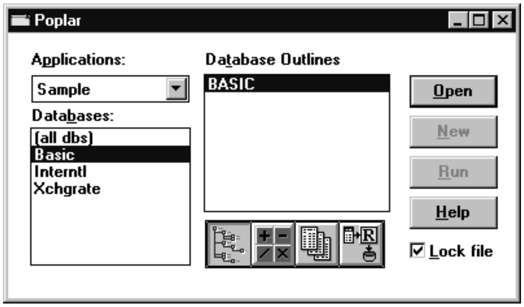
1. Create a new database. See “Creating a New Database” on page 7-9.
2. Open the new, blank outline. See “Opening Outlines” on page 8-2.
3. Add dimensions and members to the outline using one of the following means:
   - Manually add the dimensions and members. See “Adding Dimensions and Members to Outlines” on page 8-13.
   - Copy an existing outline. See “Copying Outlines” on page 8-4.
4. Verify and save the outline. See “Verifying and Saving Outlines” on page 8-5.

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, choose the application to view from the **Applications** list box; for example, the Sample application.
2. Choose the database from the **Databases** list; for example, the Basic database.



Applications: Sample

Databases: (all dbs), Basic, Interntl, Xchgrate

Database Outlines: BASIC

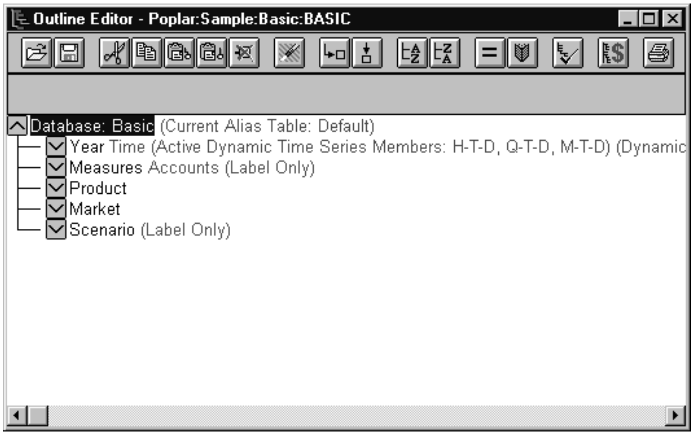
☑ Lock file

Figure 8-1, Application Desktop Window

3. If the **Lock file** option is checked, other users cannot edit the outline until you close the outline again.

4. Click the **Outline** button, .

5. Click **Open** to open the selected outline. Figure 8-2 shows the Sample Basic outline in the Outline Editor.



Outline Editor - Poplar:Sample:Basic:BASIC

```
∧ Database: Basic (Current Alias Table: Default)
   ├─ ∨ Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D, M-T-D) (Dynamic
   ├─ ∨ Measures Accounts (Label Only)
   ├─ ∨ Product
   ├─ ∨ Market
   └─ ∨ Scenario (Label Only)
```

Figure 8-2, Outline Editor

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 Hyperion Essbase 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.

Copying Outlines

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

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

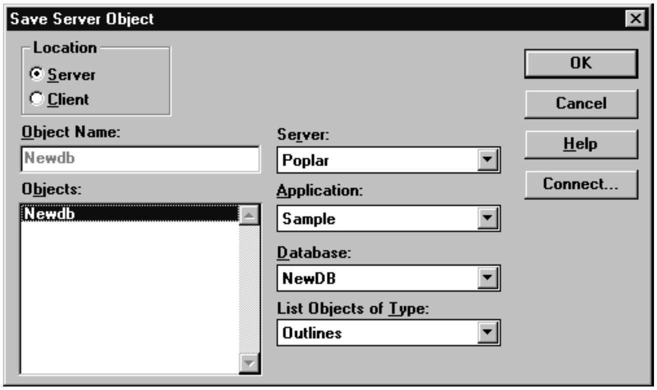
Use the COPYDB command in ESSCMD to copy the database. For example, to copy the Sample Basic database to the NewApp application and the NewDB database:

```
COPYDB Sample Basic NewApp NewDB
```

See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

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 8-2.
3. Choose File | Save As to open the **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

Figure 8-3, Save Server Object Dialog Box

4. Select the name of the *new* database (the one that you created in Step 1) in the **Database** list box.
5. A dialog box appears, asking if you should overwrite the existing database. Click Yes.

Verifying and Saving Outlines

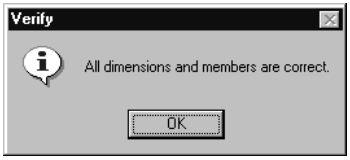
When you save an outline, Essbase verifies it. Essbase checks to ensure that:

- All member and alias names are valid. Members and aliases cannot have the same name as other members, aliases, generations, or levels. See "Naming Conventions" on page 8-13 for naming conventions for member names.
- Only one dimension is tagged as Accounts, Time, Currency Type, or Country.
- Shared members are valid. Shared members cannot occur before the actual members that they are based on. Shared members do not have any user-defined attributes. Shared members must have a real member with the same name.
- Level 0 members are not tagged as Label Only.
- Shared or Label Only members do not have formulas.
- The currency category and currency name are valid for the currency outline.
- Calc strings for members are valid.
- Level 0 Dynamic Calc members must have a formula.
- Dynamic Calc member do not have more than 100 children.

Verifying Outlines

To verify an outline:

1. Choose Outline | Verify or click the **Verify,** ✓**,** button. If the outline has no errors, the **Verify** dialog box appears:



All dimensions and members are correct.

Figure 8-4, Verify Dialog Box

2. If the outline is not valid, a dialog box appears listing the errors that the outline contains. Fix these errors and verify the outline again.

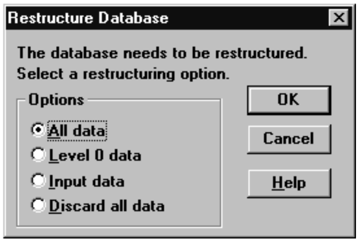
Saving Outlines

You must verify an outline before you can save it. You can save outlines to the client or the server. By default, Essbase saves outlines in the server directory.

To save an outline:

1. Choose 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 appears:



Restructure Database

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

Options

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

Figure 8-5, Restructure Database Dialog Box

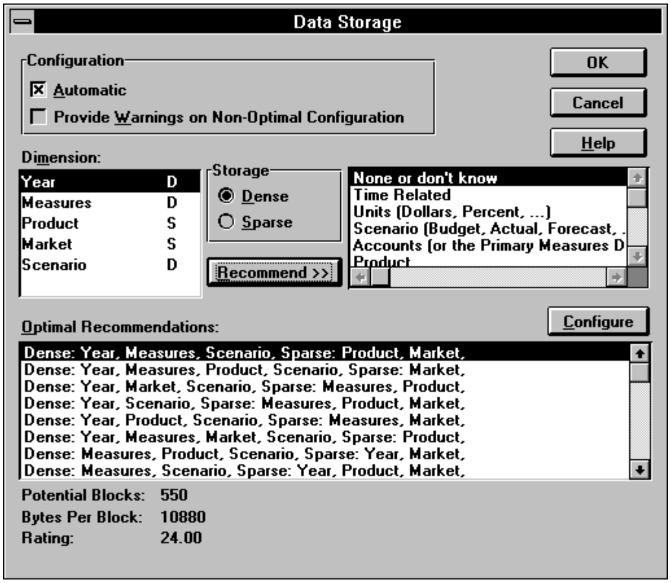
3. Choose the data that Essbase should restructure.
   - Choose **All data** when you want to keep all changes that pertain to the modified outline.
   - Choose **Level 0 data** when you expect to recalculate the database. If all data in the outline is at level 0, choosing **Level 0 data** can save space and improve calculation performance.
   - Choose **Input data** when you expect to calculate and you have loaded data into non-level 0 members.
   - Choose **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 “Restructuring Databases” on page 39-12.
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 *Start Here* booklet.

Setting Dense and Sparse Data Storage

When you create and save an outline, Essbase automatically configures the dimensions in the outline as either dense or sparse. You can view the current storage configuration by choosing Settings | 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
Product

Optimal Recommendations:

Dense: Year, Measures, Scenario, Sparse: Product, Market,
Dense: Year, Measures, Product, Scenario, Sparse: Market,
Dense: Year, Market, Scenario, Sparse: Measures, Product,
Dense: Year, Scenario, Sparse: Measures, Product, Market,
Dense: Year, Product, Scenario, Sparse: Measures, Market,
Dense: Year, Measures, Market, Scenario, Sparse: Product,
Dense: Measures, Product, Scenario, Sparse: Year, Market,
Dense: Measures, Scenario, Sparse: Year, Product, Market,

Potential Blocks:	550
Bytes Per Block:	10880
Rating:	24.00

Figure 8-6, Data Storage Dialog Box

By default, the configuration is set to **Automatic**. You can clear **Automatic** to set the dense and sparse configuration manually.

For information on choosing dense or sparse storage, see “Selecting Sparse and Dense Dimensions” on page 4-6.

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 appears.
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 appears. Click Yes to rename the dimension or member.

To rename a member using the Member Specification dialog box:

1. Select the dimension or member name.
2. Click the **Data Dictionary** button, , press the Enter key, or choose Edit | Attributes to open the **Member Specification** dialog box
3. Enter the new name in the **Name** text field.
4. Click OK.

Importing and Exporting 2.x Outlines

You can import Version 2.x outlines that have been exported to text files. When you export an outline to a text file, Essbase does *not* export any functionality added to the outline since Version 2.x.

Warning: Only use this feature to import or export Version 2.x or earlier outlines.

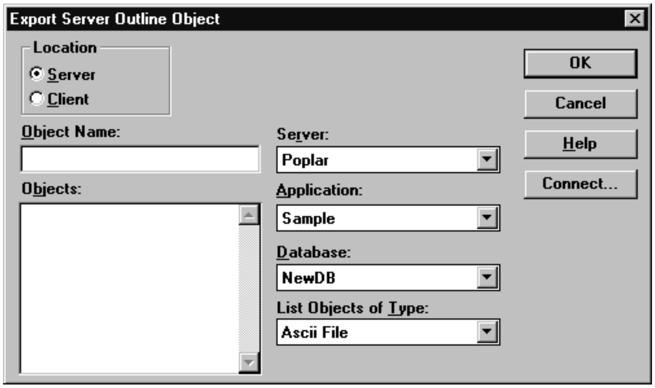
Exporting Outlines

You can export an outline to an ASCII text file. The text file contains the structure of the outline; that is, the organization of the dimensions and members.

Warning: Only use this feature to export Version 2.x or earlier outlines.

To export an outline to a cross-platform text file:

1. Open the outline to export. If you don't know how to do this, see "Opening Outlines" on page 8-2.
2. Choose File | Export | Outline to open the **Export Server Outline Object** dialog box.



Export Server Outline Object

Location: ● Server ○ Client

Server: Poplar

Application: Sample

Database: NewDB

List Objects of Type: Ascii File

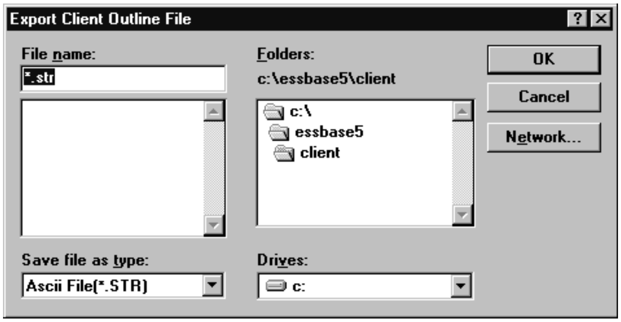
Figure 8-7, Export Server Outline Object Dialog Box

3. Make sure the appropriate Essbase server, application, and database are selected from their respective lists.
4. Specify where to save the outline to by clicking either the **Server** or **Client** button.

   If you select **Server**, the outline can 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 outline in the **Object Name** text box or select it from the **Objects** list box.

   If you select **Client**, the outline can 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 browse the file system to determine where to save the outline.

Note: The `\ESSBASE\APP` and `\ESSBASE\CLIENT` are the default directories specified during installation. You may have set these directories differently.



File name: *.str

Folders: c:\essbase5\client

c:\
essbase5
client

Save file as type: Ascii File(*.STR)

Drives: c:

Figure 8-8, Export Client Outline File Dialog Box

5. Enter the name of the file to which to export the outline. Essbase appends a `.STR` to the file name.
6. Click OK. Essbase exports the outline to the file.

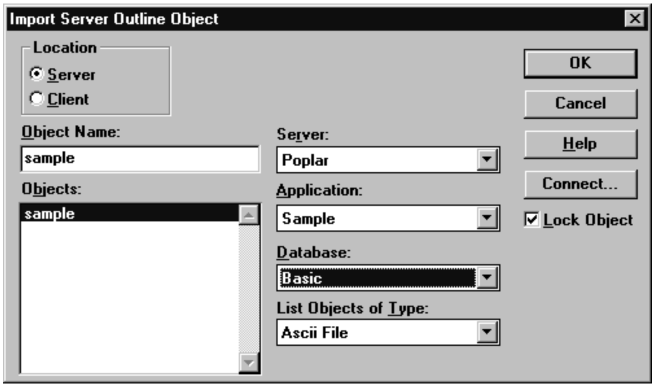
Importing Outlines

After you export an outline to a text file, you can import the file into an outline in the Outline Editor. When you import an outline file into an open outline, the text file replaces the open outline in the Outline Editor. If you save the outline file, it overwrites the existing outline file.

Warning: Only use this feature to import Version 2.x or earlier outlines.

To import an outline:

1. Open the outline into which you want to import the outline. If you don't know how to do this, see "Opening Outlines" on page 8-2.
2. Choose File | Import | Outline to open the **Import Server Outline Object** dialog box.



Location: ● Server ○ Client

Object Name: sample

Objects: sample

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Ascii File

☑ Lock Object

Figure 8-9, Import Server Outline Object Dialog Box

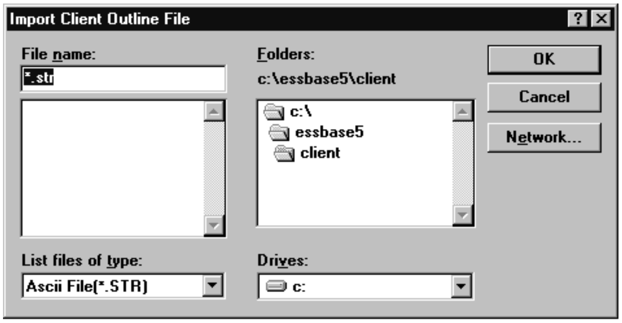
3. Make sure the appropriate Essbase server, application, and database are selected from their respective lists.

4. Specify the location of the outline by clicking either the **Server** or **Client** button.

   If you select **Server**, the outline 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 8-9, for example, you could choose `sample`.

   If you select **Client**, the outline 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 an outline from a standard **Open Client Data Files** dialog box. Choose the outline to open.

Note: The `\ESSBASE\APP` and `\ESSBASE\CLIENT` are the default directories specified during installation. You may have set these directories differently.


Import Client Outline File

File name: *.str

Folders: c:\essbase5\client

List files of type: Ascii File(*.STR)

Drives: c:

Figure 8-10, Open Client Outline File Dialog Box

5. Click OK. Essbase imports the outline.

Use the LISTMEMBERS command in ESSCMD to import outlines. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Adding Dimensions and Members to Outlines

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

- "Naming Conventions" on page 8-13
- "Adding Dimensions to Outlines" on page 8-15
- "Adding Members to Dimensions" on page 8-16

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

Naming Conventions

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

- The maximum length is 79 characters.
- 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)
,	(comma)
{ }	(braces)
-	(dash, hyphen, or minus sign)
=	(equals sign)
<	(less than sign)
( )	(parentheses)
.	(period)
+	(plus sign)
'	(single quotation mark)
_	(underscore)
|	(vertical bar)

Note: For information about using the ampersand (&) in dimension or member names, press the Help button in the **Substitution Variables** dialog box.

- Do not use spaces at the beginning or end of a name.
- Do not use the following words as dimension or member names:
    - `$$$UNIVERSE$$$`
    - `#MISSING` or `#MI`
    - Calc script commands. For a list of calc script commands, see the online *Technical Reference* in your `DOCS` directory.
    - Report script commands. For a list of report script commands, see the online *Technical Reference* in your `DOCS` directory.
    - Names of other dimensions, members (unless the member is shared), generation names, level names, aliases, and combinational aliases in the database.
- In calc scripts, report scripts, or formulas, you must enclose in quotation marks names that contain any of the following characters:

`*`	(asterisk)
`@`	(at sign)
`[ ]`	(brackets)
`:`	(colon)
`,`	(comma)
`{ }`	(braces)
`-`	(dash, hyphen, or minus sign)
`<`	(less than sign)
`( )`	(parentheses)
`+`	(plus sign)
`/`	slash)
`;`	(semicolon)

Note: Tips to make member and alias names unique:

- Concatenate the member and alias name.
- Add prefixes to member names, then create a lookup table that references the actual member names (without the prefix). Let the user view the alias names in the lookup table, not the member names.
- Add a suffix to member names.

Adding Dimensions to Outlines

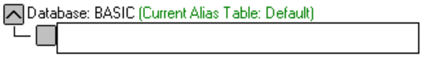
Dimensions are the highest level of organization in an outline. Dimensions are made up of members. For more information on dimensions, see Chapter 3, Multidimensional Concepts.

To add dimensions as children of the outline:

1. Select the database name, for example, Basic.
2. Choose 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. Choose Edit | Add Sibling or click the **Add Sibling** button, , to open a member edit text box in the Outline Editor.

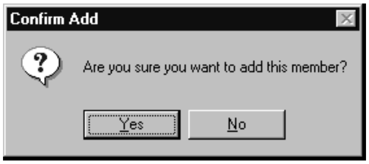


Database: BASIC (Current Alias Table: Default)

Figure 8-11, Member Edit Text Box

To finish adding dimensions, proceed with the following steps:

1. Enter the name of the dimension; for example, `Year`. See “Naming Conventions” on page 8-13 for restrictions.
2. Press Enter. A dialog box opens asking if you are sure that you want to add the dimension.



Confirm Add

Are you sure you want to add this member?

Figure 8-12, Confirm Add Dialog Box for Dimensions and Members

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 8-17.

Adding Members to Dimensions

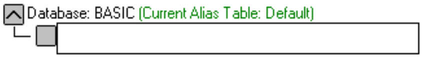
Members are organized into dimensions. You can nest members inside of other members. For more information on members, see Chapter 3, Multidimensional Concepts.

To add members as children of dimensions:

1. Select the dimension; for example, Year.
2. Choose 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. Choose Edit | Add Sibling or click the **Add Sibling** button, ↓□, to open a member edit text box in the Outline Editor.

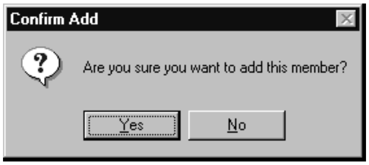


Database: BASIC (Current Alias Table: Default)

Figure 8-13, Member Edit Text Box

To finish adding members, proceed with the following steps:

1. Enter the name of the member; for example, `Qtr2`. See “Naming Conventions” on page 8-13 for restrictions.
2. Press Enter. A dialog box opens asking if you are sure that you want to add the member.



Confirm Add

Are you sure you want to add this member?

Figure 8-14, Confirm Add Dialog Box for Dimensions and Members

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 8-17.

Positioning Dimensions and Members

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

- "Sorting Dimensions and Members" on page 8-17
- "Moving Members" on page 8-18

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

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).

Warning: Moving dimensions and members can affect the results of your calculations. 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 in the outline; for example, Basic.
2. Choose Edit | Sort Ascending to sort the dimensions in alphabetical order or choose 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. Choose Edit | Sort Ascending to sort the members in alphabetical order or choose Edit | Sort Descending to sort the members in reverse alphabetical order.

Moving Members

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

Warning: Moving dimensions and members can affect the results of your calculations. Also, sorting members could move a shared member before the actual member in the outline, something that we do not recommend.

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 ⊞ appears.
3. Place the icon over a member at the new location. A border appears 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 appears 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 appears with a bent line └⊞.
4. Release the mouse button.

Naming Generations and Levels

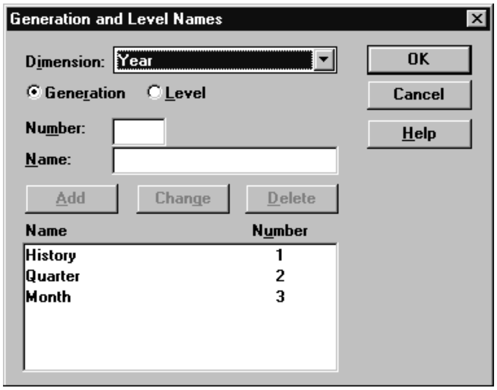
You can create your own names for generations and levels in an outline. The name is a word or phrase that describes the generation or level. For example, you might create a generation name called Cities for all cities in the outline.

Use generation and level names in calc 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 calc script to all members in a specific generation. See Chapter 30, Developing Calc Scripts, for information about developing calc scripts.

See "Member Relationships, Generations, and Levels" on page 3-4 for information about generations and levels.

To create a generation name:

1. Choose Outline | Gen/Level Names to open the **Generation and Level Names** dialog box.



Generation and Level Names

Dimension: Year

● Generation ○ Level

Number:

Name:

Name	Number
History	1
Quarter	2
Month	3

Figure 8-15, Generation and Level Names Dialog Box

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, choose the **Generation** option. To name a level, choose the **Level** option. For example, to name the months in the Sample Basic database, choose **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 "Naming Conventions" on page 8-13.

6. Click **Add**. The new name appears in the list box.
7. Click OK.

To view the list of generation or level names, choose 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 8-20
- "Customizing Your View" on page 8-21
- "Setting the Outline Font" on page 8-22
- "Confirming Changes to Dimensions and Members" on page 8-23

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, check Case Sensitive Members in the Settings menu.

Customizing Your View

You can customize your 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 Attributes” on page 9-13.
- **Formula Objects**—displays formulas to the right of the member. For information on creating formulas, see “Naming Generations and Levels” on page 8-19.
- **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 Dimension Type Attributes” on page 9-1.
- **Aliases**—displays aliases in parentheses to the right of the member. For more information on creating aliases, see Chapter 10, Creating and Managing Aliases.

Use the DISPLAYALIAS command in ESSCMD to view all of the aliases in an alias table. To view all of the aliases in the Long Name alias table:

```
DISPLAYALIAS "Long Names"
```

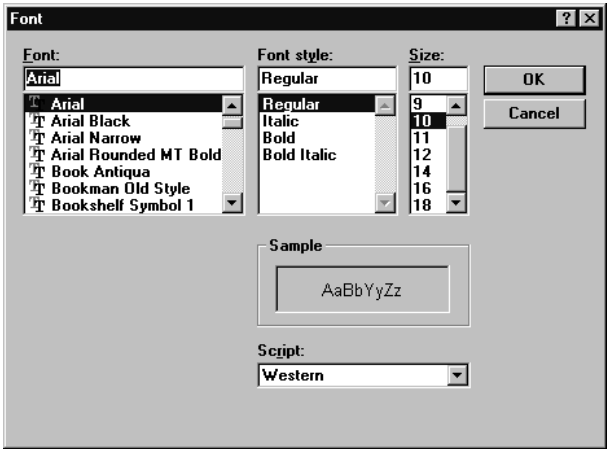
See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

- **Attributes**—displays member attributes to the right of the member. For more information on assigning attributes, see Chapter 9, Setting Dimension and Member Attributes.
- **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 9-22.
- **Toolbar**—displays the Outline Editor toolbar, which contains shortcuts for working with outlines, dimensions, and members.
- **Attribute Bar**—displays the Outline Editor attribute bar, which contains shortcuts for changing or assigning attributes 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. Choose Options | Font to open the **Font** dialog box.



Font

Font: Arial

Font style: Regular

Size: 10

Script: Western

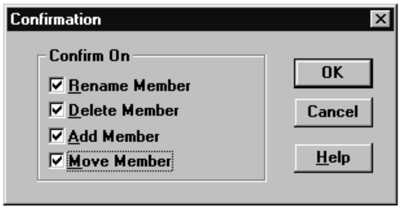
Figure 8-16, Font Dialog Box

2. Choose the desired font from the **Font** list. You can also choose 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.

1. Choose Options | Confirmation to open the **Confirmation** dialog box.



Confirm On

☑ Rename Member

☑ Delete Member

☑ Add Member

☑ Move Member

Figure 8-17, Confirmation Dialog Box

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

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

3. Click OK.

Chapter 9 Setting Dimension and Member Attributes

After you've created and organized your Hyperion Essbase OLAP Server outline, as described in Chapter 8, Creating and Changing Database Outlines, you are ready to start specifying how the dimensions and members in the outline behave. This chapter describes each dimension and member attribute and how to set them.

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

Setting Dimension Type Attributes

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 Time and Accounts dimensions before other dimensions in the database. By default, all dimensions are tagged as None.

Note: The Time and Accounts attributes are inherited by all members that are in Time or Accounts dimensions. The Country and Currency attributes are not inherited by their members.

The following sections describe how to tag dimensions:

- "Tagging a Time Dimension" on page 9-2
- "Tagging an Accounts Dimension" on page 9-3
- "Tagging a Country Dimension" on page 9-9
- "Tagging a Currency Partition" on page 9-10

Tagging a Time Dimension

Use Time dimensions 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 Time dimension:

- 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 attribute.

Notes:

- You can create an outline that does not have a Time dimension.
- You can add time members to dimensions that are not tagged as Time.

Setting the Time Attribute

To tag a dimension as Time in the Outline Editor:

1. Select the dimension that you want to tag; for example, Year.
2. Click the **Time** button, . "Time" appears next to the dimension name.

To tag a dimension as Time in the Dimension Specification 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 choose Edit | Attributes to open the **Dimension Specification** dialog box.
3. Choose **Time** in the **Dimension Type** box.
4. Click OK.

Tagging an Accounts Dimension

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 attribute.
- To calculate members of the Accounts dimension on the second pass through the outline, see “Setting Two-Pass Calculation Attributes” on page 9-11.

Note: You can create an outline that does not have an Accounts dimension.

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

- “Introducing Time Balance Attributes” on page 9-4
- “Introducing Skip Attributes” on page 9-6
- “Setting Time Balance Attributes” on page 9-7
- “Setting Variance Reporting Attributes” on page 9-8
- “Setting Hyperion Essbase Currency Conversion Attributes” on page 9-9

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

Setting the Accounts Attribute

To tag a dimension as Accounts in the Outline Editor:

1. Select the dimension that you want to tag; for example, Measures.
2. Click the **Accounts** button, Acct. “Accounts” appears next to the dimension name.

To tag a dimension as Accounts in the Dimension Specification 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 choose Edit | Attributes to open the **Dimension Specification** dialog box.
3. Choose **Accounts** in the **Dimension Type** box.
4. Click OK.

Introducing Time Balance Attributes

When you set a Time Balance attribute on a member in an Accounts dimension, it affects how Essbase calculates that member's parent 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 attribute causes parents, for example Qtr1, to roll up differently.

Example of Setting the Time Balance as None

This is the default value. When you set the Time Balance attribute 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 attributes 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 9-1 shows this sample consolidation:

```
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
```

Figure 9-1, Consolidation of OpeningInventory Tagged as First

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 9-2 shows this sample consolidation:

```
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
```

Figure 9-2, Consolidation of EndingInventory Tagged as Last

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 9-3 shows this sample consolidation:

```
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
```

Figure 9-3, Consolidation of AverageInventory Tagged as Average

Introducing Skip Attributes

If you set the Time Balance as First, Last, or Average, you must set the Skip attribute to tell Essbase what to do when it encounters missing values or values of 0.

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

If you choose...	**Essbase...**
None	Does not skip data when calculating the parent value. However, if Essbase encounters `#MISSING` data when calculating an average, it doesn't divide by the total number of members. It divides by the number of members with actual values. This means that setting the skip attribute to None or `#MISSING` is the same for Average (but not for First and Last).
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 attribute of Missing or Missing and Zeros, then the parent of that time period matches the last non-missing child. In Figure 9-4, for example, EndingInventory is based on the value for Feb, because Mar does not have a value.

```
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
```

Figure 9-4, Example of Skip Attribute

Setting Time Balance Attributes

To set a Time Balance attribute in the Outline Editor:

1. Select the member that you want to set the attribute for; for example, OpeningInventory.
2. Click the button that corresponds to the attribute that you want to set.
   - To set the Time Balance Attribute as First, click the **First** button, . “TB First” appears next to the member’s name.
   - To set the Time Balance Attribute as Last, click the **Last** button, . “TB Last” appears next to the member’s name.
   - To set the Time Balance Attribute as Average, click the **Average** button, J+F+M/3. “TB Average” appears next to the member’s name.
   - To set the Time Balance Attribute as None, click the **None** button, ⊘.
3. Click the button that corresponds to the Skip attribute 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 **Zero and Missing** button, #MI, 0.

To tag a dimension as Accounts using the Dimension Specification dialog box:

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

Setting Variance Reporting Attributes

Variance Reporting attributes 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 attribute.

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 set an Expense attribute in the Outline Editor:

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

To tag an Accounts member as Expense or Non Expense using the Member Specification dialog box:

1. Select the member that you want to set the attribute for; for example, COGS.
2. Click the **Data Dictionary** button, , press the Enter key, or choose Edit | Attributes to open the **Member Specification** dialog box.
3. Click the **Accounts** tab.
4. Choose **Expense** or **Non Expense**.
5. Click OK.

Setting Hyperion Essbase Currency Conversion Attributes

Currency Conversion attributes define categories of currency exchange rates. These attributes are used only in currency databases. For more information on currency attributes, see Chapter 42, Designing and Building Currency Conversion Applications.

If you choose...	Essbase...
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 attributes 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 choose Edit | Attributes to open the **Dimension Specification** dialog box.
3. Click the **Account Info** tab.
4. Choose **None**, **No Conversion**, or **Category** from the **Currency Conversion** box.

Tagging a Country Dimension

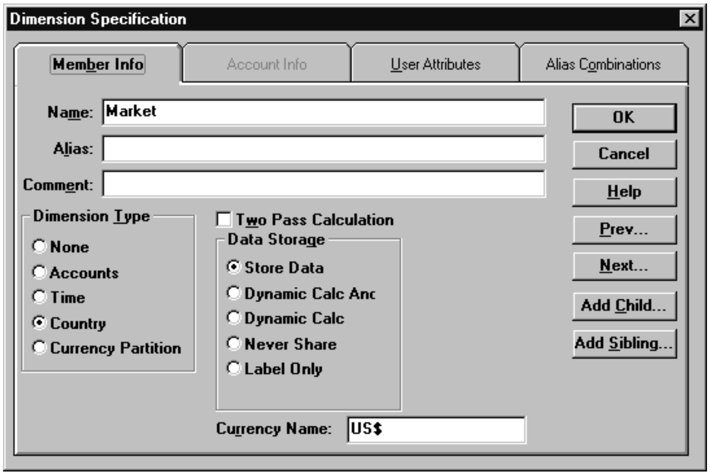
Use Country dimensions to track business activities in multiple countries. If you track business activity in the United States and Canada, for example, your Country dimension should contain states, provinces, and countries. If a dimension is tagged as Country, you can set the Currency Name attribute. Currency Name 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 42, 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, . "Country" appears next to the dimension name.
3. If you want to set the currency name, click the **Data Dictionary** button, , press the Enter key, or choose Edit | Attributes to open the **Dimension Specification** dialog box.



Dimension Specification

Name: Market

Alias:

Comment:

Dimension Type

○ None
○ Accounts
○ Time
● Country
○ Currency Partition

□ Two Pass Calculation

Data Storage

● Store Data
○ Dynamic Calc Anc
○ Dynamic Calc
○ Never Share
○ Label Only

Currency Name: US$

Figure 9-5, Dimension Specification Dialog Box

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 Chapter 42, Designing and Building Currency Conversion Applications.

Setting Two-Pass Calculation Attributes

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 example, to calculate the ratio between Sales and Margin, Essbase needs to first 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: While Two-Pass Calculation is an attribute that you can give to any 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, 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 attribute for; for example, Margin%.
2. Click the **Two-Pass** tag button, . “Two Pass Calc” appears next to the member name.

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

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

Introducing Member Consolidation Attributes

Member consolidation attributes 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.

Table 9-1 describes each operator.

Table 9-1, 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 9-6.

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

Figure 9-6, Sample Roll Up

Essbase calculates Member1 through Member4 in Figure 9-6 as follows:

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

Figure 9-7, Sample Roll Up for Members 1 through 4

If the result of Figure 9-7 is X, then Member5 consolidates as follows:

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

Figure 9-8, Sample Roll Up for Member 5

If the result of Figure 9-8 is Y, then Member6 consolidates as follows:

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

Figure 9-9, Sample Roll Up for Member 6

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

Setting Member Consolidation Attributes

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

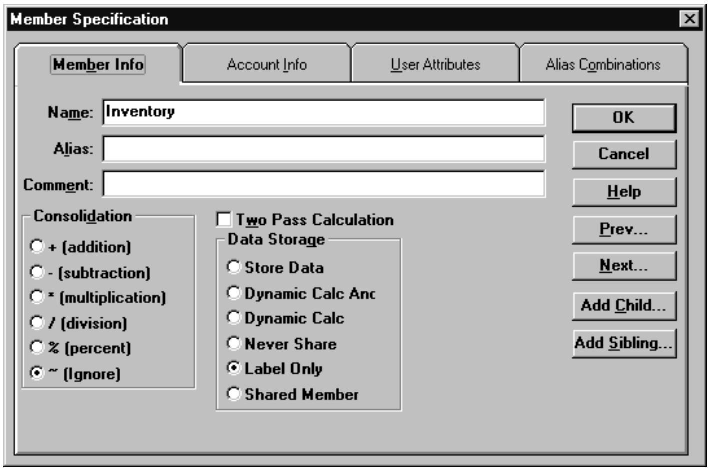
1. Select the member.
2. Click the button corresponding to the consolidation you specify:

   - 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 attribute for a member using the Member Specification dialog box:

1. Select the member.

2. Click the **Data Dictionary** button, , press the Enter key, or choose Edit | Attributes to open the **Member Specification** dialog box.



Member Specification

Name: Inventory

Alias:

Comment:

Consolidation

○ + (addition)

○ - (subtraction)

○ * (multiplication)

○ / (division)

○ % (percent)

● ~ (Ignore)

□ Two Pass Calculation

Data Storage

○ Store Data

○ Dynamic Calc Anc

○ Dynamic Calc

○ Never Share

● Label Only

○ Shared Member

Figure 9-10, Setting Consolidation Attributes

3. Choose the operator from the **Consolidation** box on the **Member Info** page.

Setting Storage Attributes

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 9-2 lists each storage attribute and tells you when to set it and where to go to learn how to set it.

Table 9-2, Choosing Storage Attributes

Use this attribute...	When you want to...	For more information, see...
Store	Store the data value with the member.	“Storing Data” on page 9-15
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 9-16
Dynamic Calc	Not calculate the data value until a user requests it, and then discard the data value.	“Dynamically Calculating Data” on page 9-16
Never Share	Not allow members to be shared implicitly.	“Implied Sharing” on page 9-18
Label Only	Create members for navigation only, that is, members that contain no data values.	“Creating Label Only Members” on page 9-17
Shared member	Share values between members. For example, the 100-20 member is stored under the 100 parent and shared under Diet parent.	“Sharing Members” on page 9-17

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 tag a member as stored:

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

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

1. Select the member.
2. Click the **Data Dictionary** button, , press the Enter key, or choose Edit | Attributes to open the **Member Specification** dialog box.
3. Choose **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.

To tag a member as Dynamic Calc or Dynamic Calc And Store in the Outline Editor:

1. Select the member.
2. Click the **Dynamic Calc** button, , or the **Dynamic Calc and Store** button, . "Dynamic Calc" or "Dynamic Calc and Store" appears next to the member names.

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

1. Select the member.
2. Click the **Data Dictionary** button, , press the Enter key, or choose Edit | Attributes to open the **Member Specification** dialog box.
3. Choose **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 Hyperion Essbase Spreadsheet Add-in. Typically, you should give Label Only members the No Consolidation attribute. For more information, see “Setting Member Consolidation Attributes” on page 9-13.

To tag a member as Label Only:

1. Select the member; for example, Inventory.
2. Click the **Label Only** button, . “Label Only” appears next to the member name.

To tag a member as Label Only using the Member Specification dialog box:

1. Select the member.
2. Click the **Data Dictionary** button, , press the Enter key, or choose Edit|Attributes to open the **Member Specification** dialog box.
3. Choose **Label Only** in the **Data Storage** box on the **Member Info** page.
4. Click OK.

Sharing 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 appears in multiple locations. This offers considerable space saving as well as processing efficiency.

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 user-defined attributes, formulas, consolidation attributes, or account attributes to 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.

Implied Sharing

The shared member attribute 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 attribute on the child and stores the data only once—thus the parent has an implied shared relationship with the child. In Figure 9-11, for example, the parent 500 has only one child, 500-10, so the parent shares the value of that child.

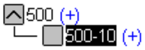


Figure 9-11, Implied Sharing of a Parent with One Child

- **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 attribute on the child and stores the data only once—thus the parent has an implied shared relationship with the child. In Figure 9-12, 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.

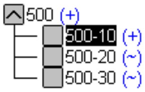


Figure 9-12, Implied Sharing of a Parent with Multiple Children

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 Attribute” on page 9-19.

Setting the Shared Member Attribute

To tag a member as shared in the Outline Editor:

1. Select the shared member.
2. Click the **Shared Member** button, . “Shared Member” appears 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 (~) appears next to the member’s name.

Setting User-Defined Attributes

You can create your own user-defined attributes for members. A *user-defined attribute* is a word or phrase about the member. For example, you might create an attribute called Debit. Use user-defined attributes in:

- Calc scripts. After you define an attribute, you can query a member for its attribute in a calc script. For example, you could multiply all members with the attribute Debit by -1 so that they appear as either positive or negative (depending on how the data is currently stored). See Chapter 30, Developing Calc Scripts.
- Report scripts. In financial reports, you could list debits as positive, whereas in accounting reports, you could list them as negative. See Chapter 34, Quick Start to Report Scripts.
- Data loading. You can change the sign of the data as it is loaded into the database based on its user-defined attribute. See "Flipping Field Signs" on page 21-24.

Rules for User-Defined Attributes

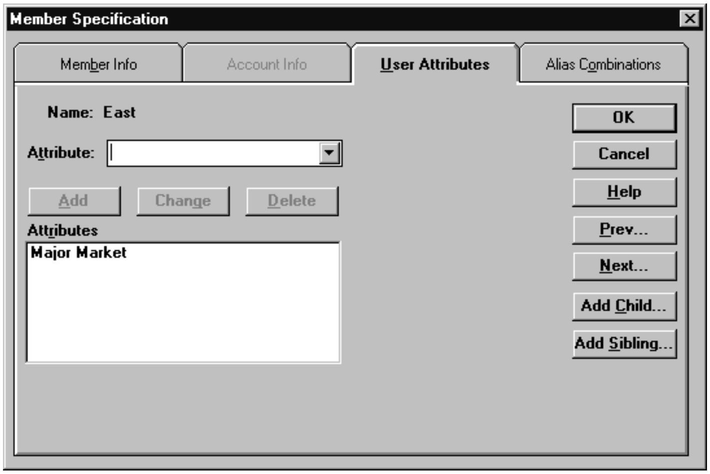
Follow these rules when creating user-defined attributes:

- You can define multiple user-defined attributes per member.
- You can't set the same user-defined attribute twice for one member.
- You can set the same user-defined attribute for different members.
- A user-defined attribute name can use the same name as a member, alias, level, or generation name. When you name user-defined attributes, follow the same naming rules as for members. See "Naming Conventions" on page 8-13.
- You can't create a user-defined attribute on shared members.
- A user-defined attribute applies to the specified member only. Descendants and ancestors of the member don't automatically receive the same attribute.

Creating User-Defined Attributes

To create a user-defined attribute in the Outline Editor:

1. Select the member to create the user-defined attribute for; for example, East.
2. Click the **Data Dictionary** button, , press the Enter key, or choose Edit | Attributes to open the **Member Specification** dialog box and click the **User Attributes** tab.



Member Specification

Name: East

Attribute:

Attributes

Major Market

Figure 9-13, User Attributes Page

3. Enter the user-defined attribute in the **Attribute** text box or choose it from the list box; for example, MajorMarket.
4. Click the **Add** button to add it to the **Attributes** list. The **Attributes** box lists all user-defined attributes for the selected member.
5. Click OK.

Setting Comments on Dimensions and Members

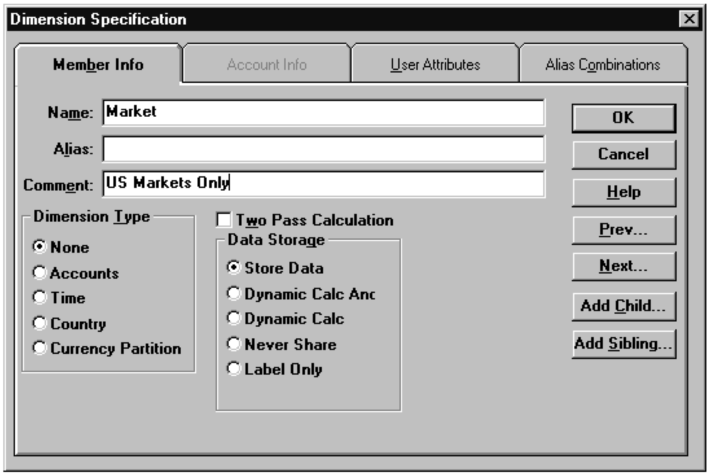
You can add comments to dimensions and members. The Outline Editor displays these comments to the right of the dimension or member in the following format:

```
/* 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 choose Edit | Attributes to open the **Dimension Specification** dialog box.



Name: Market

Alias:

Comment: US Markets Only

Dimension Type: ● None ○ Accounts ○ Time ○ Country ○ Currency Partition

☐ Two Pass Calculation

Data Storage: ● Store Data ○ Dynamic Calc And ○ Dynamic Calc ○ Never Share ○ Label Only

Figure 9-14, Comments in the Dimension Specification Dialog Box

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 appears to the right of the dimension or member in the Outline Editor.

Market /* US Markets Only */

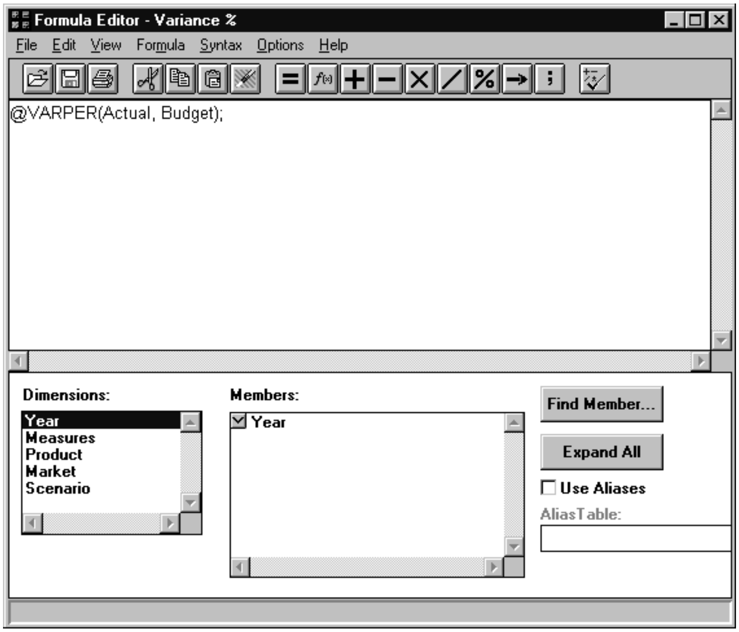
Figure 9-15, Sample Dimension Comment

Setting Formulas for Dimensions and Members

You can associate formulas with dimensions and 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 to open the Formula Editor:



```
@VARPER(Actual, Budget);
```

Figure 9-16, Formula Editor

3. Type the formula into the edit field; for example, `@VARPER(Actual, Budget)`.
4. Choose File | Close to close the Formula Editor. Essbase checks the formula's syntax. If the syntax is correct, the formula appears 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 "Building Formulas in the Formula Editor" on page 25-6.

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 a Hyperion Essbase 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 “Naming Conventions” on page 8-13.
- You can choose different aliases based on the combination of other dimension members. Table 10-1 shows members with a different alias for each market in which they are sold:

  Table 10-1, Different Aliases Based on Member Combinations

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 aliases based on member combinations, see “Creating Aliases for Member Combinations” on page 10-5.
- 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. Table 10-2 shows some products in the Sample Basic database that have two descriptive alias names:

  Table 10-2, Members with Two Aliases

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

  For information on creating multiple aliases, see “Creating New Alias Tables” on page 10-6.

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. Table 10-2, 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 10-6 and "Setting Alias Tables" on page 10-7.

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.

Figure 10-1 shows a sample alias table:

```
$ALT_NAME
100               Cola
"member name 2"  "Alias Name 2"
$END
```

Figure 10-1, Sample Alias Table

Use the DISPLAYALIAS command in ESSCMD to view the all aliases in an alias table. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

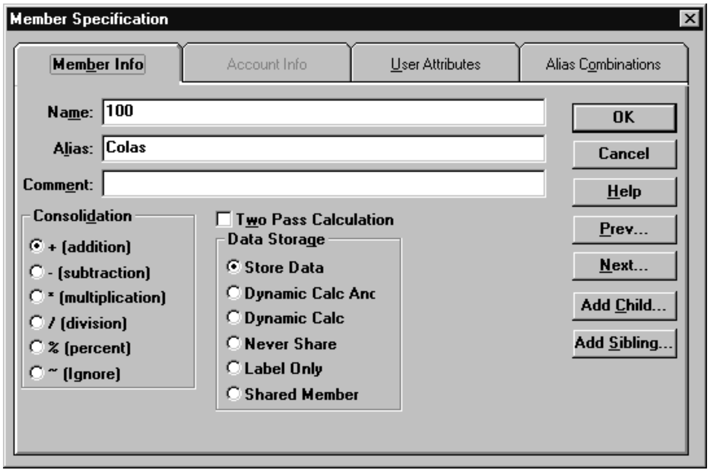
Creating Aliases for Members

You can create aliases for members. For more information on aliases, see "Introducing Aliases and Alias Tables" on page 10-1.

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 choose Edit | Attributes to open the **Member Specification** dialog box.



Name: 100

Alias: Colas

Comment:

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

□ Two Pass Calculation

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

Figure 10-2, Member Specification Dialog Box

3. Enter the alias name (for example, Colas) into the **Alias** text box on the **Member Info** page.

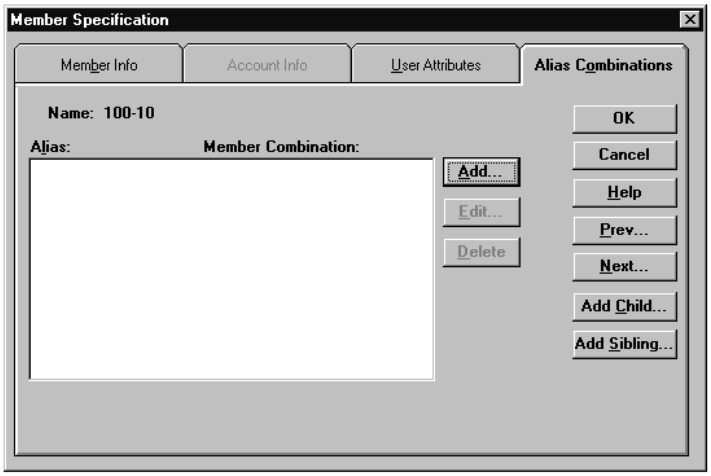
Note: Member names and alias names can be up to 79 characters long

Creating Aliases for Member Combinations

You can create aliases for members that are based on member combinations. For more information on aliases, see “Introducing Aliases and Alias Tables” on page 10-1.

To set an alias based on a member combination:

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 choose Edit | Attributes to open the **Member Specification** dialog box.
3. Click the **Alias Combinations** tab.



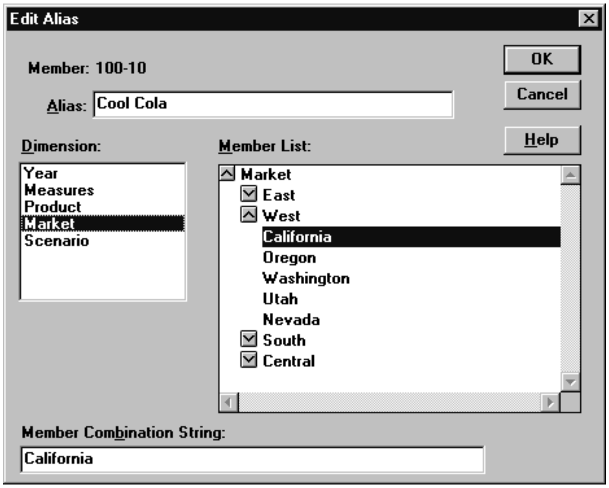
Name: 100-10

Figure 10-3, Alias Combinations Page

The **Alias Combinations** page contains the following items:

- **Alias** list box, which shows each alias name and the alias member combination string on which it is based.
- **Add** button, which lets you add a new alias to the list. Member names and alias names can be up to 79 characters long.
- **Edit** button, which lets you edit the definition for the currently selected item.
- **Delete** button, which lets you delete the currently selected item.

4. Click the **Add** button to open the **Edit Alias** dialog box:



Edit Alias

Member: 100-10

Alias: Cool Cola

Dimension:

- Year
- Measures
- Product
- Market
- Scenario

Member List:

- Market
  - East
  - West
    - California
    - Oregon
    - Washington
    - Utah
    - Nevada
  - South
  - Central

Member Combination String: California

Figure 10-4, Edit Alias Dialog Box

5. Enter the alias in the **Alias** text box; for example, `Cool Cola`.

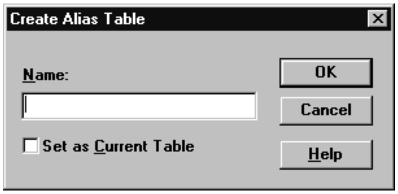
6. Either enter the member combination string in the **Member Combination String** text box or choose the members from the **Dimension** and **Member** list boxes. To expand or contract the **Member List**, double-click the expand/contract box in front of the member name. To add the member to the combination list, select the member.

7. Click OK.

Creating New Alias Tables

To create a new alias table for the selected outline:

1. Choose Outline | Aliases | Create Table to open the **Create Alias Table** dialog box.



Create Alias Table

Name:

□ Set as Current Table

Figure 10-5, 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 checking the **Set as Current Table** box.
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 10-3.

To set an alias table to be the current alias table:

1. Choose Outline | Aliases | Set Table to open the **Set Current Alias Table** dialog box.

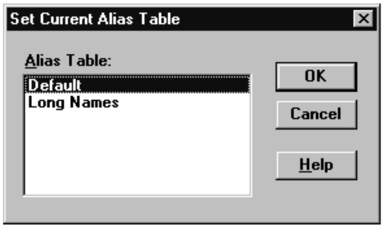


Figure 10-6, Set Current Alias Table Dialog Box

2. Choose 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 10-6.
3. Click OK.

You can view and set the current alias table using ESSCMD, as explained in the following table.

To perform this task...	Use this ESSCMD...	For example...
View a list of available alias tables	`LISTALIASES`	`LISTALIASES`
Set the current alias table	`SETALIAS`	`SETALIAS "Long Names"`

For help about an individual command, see the online *Technical Reference* in your `DOCS` directory. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Copying Existing Alias Tables

To copy an alias table to a different name in the same outline:

1. Choose Outline | Aliases | Copy Table to open the **Copy Alias Table** dialog box.

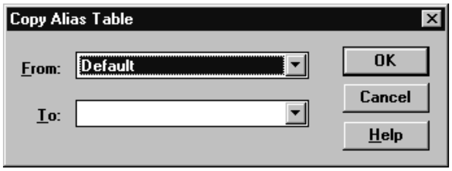


Figure 10-7, Copy Alias Table Dialog Box

2. Choose 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. Choose Outline | Aliases | Rename Table to open the **Rename Alias Table** dialog box.

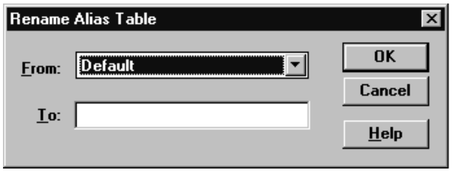


Figure 10-8, Rename Alias Table Dialog Box

2. Choose 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. Choose Outline | Aliases | Delete Table to open the **Delete Alias Table** dialog box.

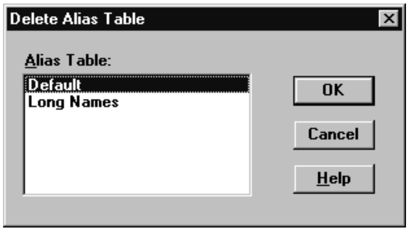


Figure 10-9, Delete Alias Table Dialog Box

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. Choose Outline | Aliases | Clear Table to open the **Clear Alias Table** dialog box.

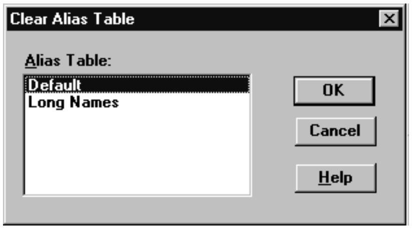


Figure 10-10, Clear Alias Table Dialog Box

2. Select the alias table to clear from the **Alias Table** list box.
3. Click OK.

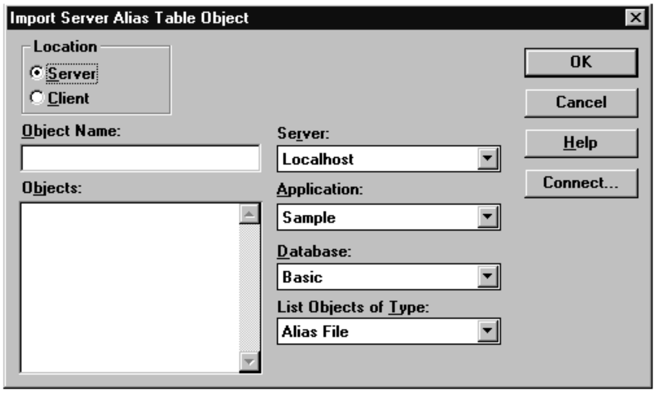
Importing and Exporting Alias Tables

You can import or export alias tables into an outline. For more information on alias tables, see “Introducing Aliases and Alias Tables” on page 10-1.

Importing an Alias Table

To import an alias table:

1. Choose File | Import | Alias Table to open the **Import Server 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

Figure 10-11, Import Alias Table Object Dialog Box

2. Select the alias table name. All alias table names end in `.ALT`.
3. Specify the location of the alias table by clicking either the **Server** or **Client** button.

   If you select **Server**, the alias table to import 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 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.

Choose the alias table to open.

Note: The `\ESSBASE\APP` and `\ESSBASE\CLIENT` are the default directories specified during installation. You may have set these directories differently.

4. Click OK.

Use the LOADALIAS command in ESSCMD to perform this task. To unload an alias table, use the UNLOADALIAS command in ESSCMD.

See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Exporting an Alias Table

To export an alias table:

1. Choose File | Export | Alias Table to open the **Export Server Alias Table Object** dialog box.
2. Save the alias table to the desired location.

Use the LISTLINKEDOBJECTS and PURGELINKEDOBJECTS commands in ESSCMD to perform these tasks. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD

Chapter 11 Linking Objects to Your Essbase Data

This chapter describes Linked Reporting Objects (LRO). This feature, similar to the file attachment feature in many e-mail software packages, lets you link various kinds of data with any cell in a Hyperion Essbase database. Linked Reporting Objects provides improved support for planning and reporting applications and can enhance your data analysis capabilities.

This chapter contains the following sections:

- "What Are Linked Reporting Objects?" on page 11-1
- "How Do Linked Reporting Objects Work with Essbase?" on page 11-3
- "Managing Linked Reporting Objects" on page 11-4
- "Viewing and Deleting Linked Reporting Objects" on page 11-6

What Are Linked Reporting Objects?

Linked reporting objects are objects that you associate with specific data cells in an Essbase database. An object can be a paragraph of descriptive text, a separate file such as a graph, bitmap image, or audio clip, a Uniform Resource Locator (URL), or a link to data in another Essbase database.

Essbase supports the following types of linked objects:

Table 11-1, 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, or an audio clip. The file can also be an individual HTML file (for example, `mypage.htm`), but it cannot be a full Web address (for example, `http://www.hyperion.com`).

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. Examples of URLs are:
	`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 6, 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.

Users create linked objects through the Hyperion Essbase 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 Essbase server where they are available to any user with the appropriate access privileges. Users retrieve and edit the objects through the Linked Object Browser, which displays all objects linked to the selected cell.

The next section describes in more detail how Essbase manages linked reporting objects. For more information on how end users work with these objects, see the *Spreadsheet Add-in User's Guide*.

How Do Linked Reporting Objects Work with Essbase?

Linked reporting objects 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 Hyperion Essbase API functions to refer to the object.

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 Storage Manager 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 Essbase 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.

The third kind of linked object, linked partition, is available through the Hyperion Essbase Partitioning feature. For more information on linked partitions, see Chapter 6, Designing Partitioned Applications.

Notes:

- 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 linked reporting objects. This is because linked reporting objects 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.

Managing Linked Reporting Objects

This section discusses storing and maintaining linked reporting objects on the server.

Providing Access to Linked Reporting Objects

Users who add, edit, and delete linked reporting objects through client interfaces need to have the appropriate security privileges 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. The following table lists the privileges required for several different tasks.

Table 11-2, Privileges Required for LRO Tasks

Task	Access Level
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

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).

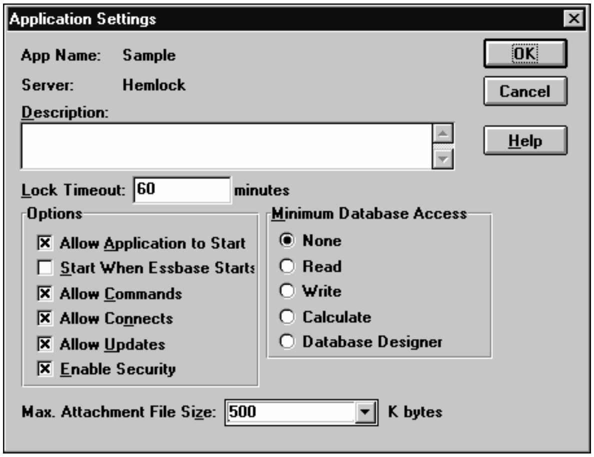
Limiting LRO File Sizes

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 appears.

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.

By default, the maximum file size for linked files is unlimited. To specify a maximum file size for an application:

1. In Application Manager, connect to the appropriate server and select the name of the application.
2. Choose Application | Settings. The following dialog box appears:



Application Settings

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

Figure 11-1, Application Settings Dialog Box

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.

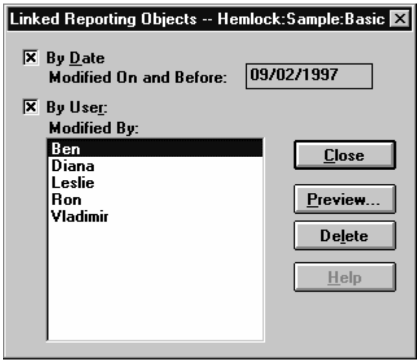
4. Click OK to save your setting.

Viewing and Deleting Linked Reporting Objects

Users work with linked reporting objects 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 the Application Manager, you can view and delete all linked reporting objects for the entire database. You can also view linked reporting objects 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. Choose Database | Linked Reporting Objects. The following dialog box appears:



Linked Reporting Objects -- Hemlock:Sample:Basic

☒ By Date
Modified On and Before: 09/02/1997

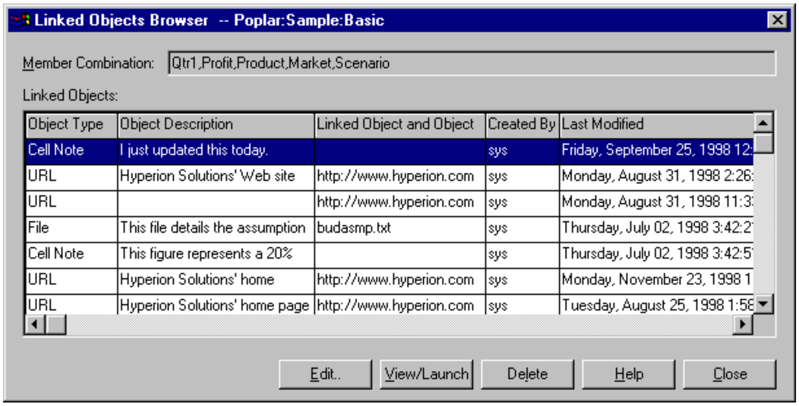
☒ By User:
Modified By:
Ben
Diana
Leslie
Ron
Vladimir

Figure 11-2, Linked Reporting Objects Dialog Box

3. Specify your selection criteria as follows:
   - To select by modification date, check the **By Date** check box and enter a modification date. Objects modified on or before the date are selected.
   - To select by user name, check the **By User** check box and select a user name. Objects last modified by the user are selected.

   If you check *both* boxes, objects matching both criteria are selected. If you leave both boxes unchecked, 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**:



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

Figure 11-3, Linked Objects Browser

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 appears 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 Essbase 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.

Notes:

- 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 6, Designing Partitioned Applications.
- You cannot change the member combination associated with any linked object through 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.

Use the LISTLINKEDOBJECTS and PURGELINKEDOBJECTS commands in ESSCMD to perform these tasks. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD for information about ESSCMD.

Chapter 12 Introducing Dynamic Dimension Building

Some Hyperion Essbase OLAP Server 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 the Outline Editor.

- The data source can specify the member names and aliases, the formulas and consolidation attributes to associate with them, the names of generations and levels, currency name and category, data storage attributes, and 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. It's best to create a separate rules file for each dimension. For more information on data sources and rules files, see Chapter 19, Introducing Data Loading.

This chapter provides a background on dynamic dimension building. For information on how to build dimensions dynamically, see Chapter 13, 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 19, Introducing Data Loading.

To create dimensions using a rules file:

1. Determine whether to use the same rules file for dimension building and data loading.

Use the same rules file to...	Use a different rules file to...
• Modify the outline based on the contents of the data source. • Load the data source and build new dimensions at the same time.	• Build an outline from scratch. • Re-use the dimension build or data load separately. • Perform different field operations. • Use data sources that contain no data values, only dimensions.

Note: You can neither merge nor separate rules files after you create them.

2. Use the Data Prep Editor to create a dimension build rules file. Choose View | Dimension Building Fields or click the **Dimension Build** button, , on the toolbar to put the 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 20, Setting up a Rules File to Manipulate Records.

3. Choose 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 attributes.
   - 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 choose the dense/sparse configuration, and how to combine selection and rejection criteria.

4. Validate and save the rules file using the Options | Validate menu command. See “Validating and Saving Data Load Rules” on page 20-18.

5. Perform the dimension build. See Chapter 22, Performing a Data Load.

Introduction to Build Methods

The build method that you choose 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 choose 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 being 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 12-1, Build Method Guidelines

If your data source contains...	**Such as...**	**And you want to...**	**Use this build method...**	**And specify...**
Top-down data	Year, Quarter, Month	Modify the attributes 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 attributes 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 attributes of existing dimensions and members	Parent/Child References	Whether each field is the parent or child
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	

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 20, 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.

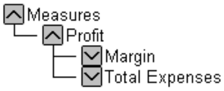
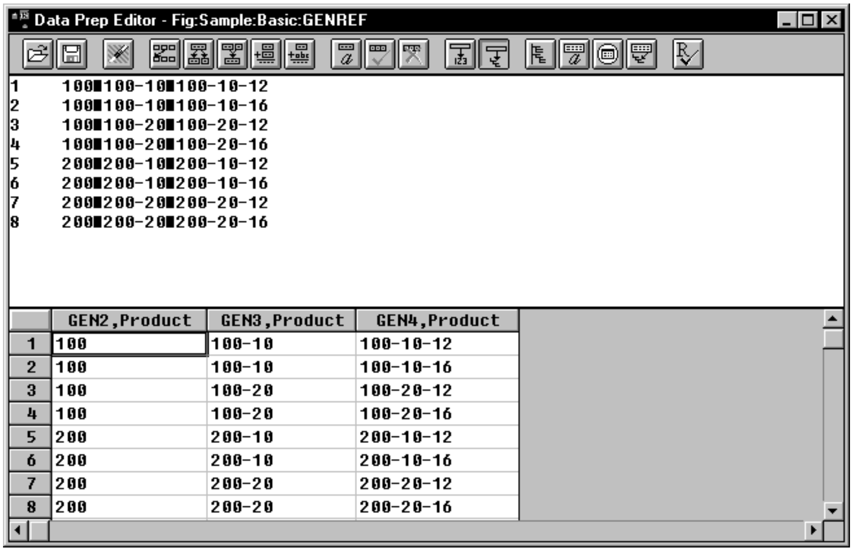


Figure 12-1, Generations

The top half of Figure 12-2 shows a top-down data source used to build the Product dimension, GENREF.TXT. The bottom half of Figure 12-2 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 “Setting Field Types” on page 13-9.



```
1     100■100-10■100-10-12
2     100■100-10■100-10-16
3     100■100-20■100-20-12
4     100■100-20■100-20-16
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
1	100	100-10	100-10-12
2	100	100-10	100-10-16
3	100	100-20	100-20-12
4	100	100-20	100-20-16
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

Figure 12-2, Rules File for Generation Build

Figure 12-3 shows the tree Essbase builds from this data source and rules file:

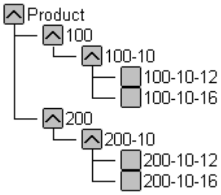


Figure 12-3, Generation References

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 fields are Alias, Attribute, Formula, DuplicateGeneration, DuplicateGenerationAlias, CurrencyName, CurrencyCategory, and User-DefinedAttribute.

  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.

Level data sources are defined from a bottom-up hierarchical structure. That is, the higher numbered levels are specified first in the data source. In Figure 12-4 for example, the Measures dimension has a Profit member which is Level 2. Profit has Margin and TotalExpenses members, which are Level 1. Margin has Sales and CostofGoodsSold members, which are Level 0. This is the opposite of how data is specified for generation references (top-down).

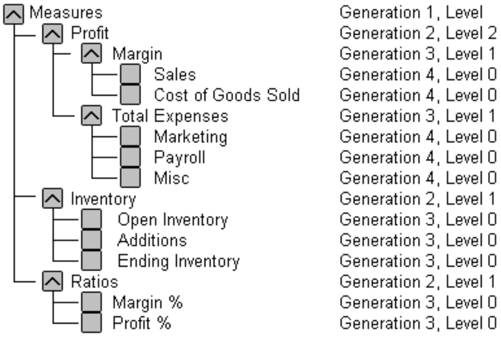
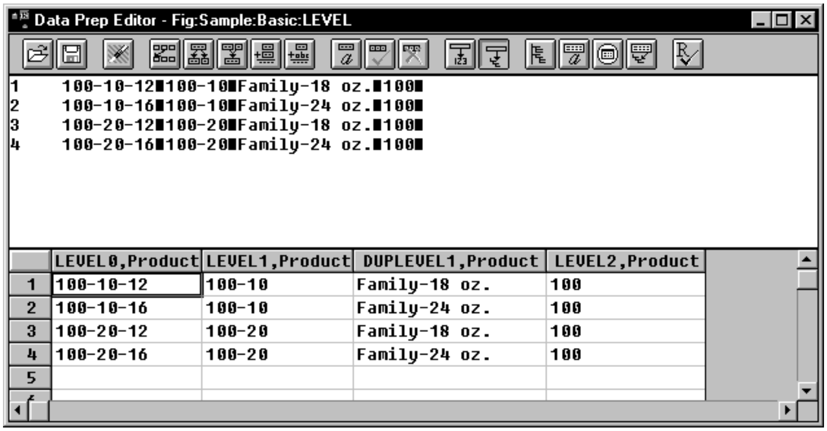


Figure 12-4, Generation and Level Numbers

For example, the last column of the data source in Figure 12-5 contains new shared members (100-10-12, 100-10-16, 100-20-12, and 100-20-16) and their aliases (Family-18 oz., and Family-24 oz.) to add to the Product dimension in the Sample Basic database.

The rules file specifies the level number and field type for each field in the data source. For more information, see “Setting Field Types” on page 13-9. To build the tree in Figure 12-6 for example, use Figure 12-5 to set up the data source, LEVEL.TXT, and rules file, LEVEL.RUL to match.



Data Prep Editor - Fig:Sample:Basic:LEVEL

```
1     100-10-12■100-10■Family-18 oz.■100■
2     100-10-16■100-10■Family-24 oz.■100■
3     100-20-12■100-20■Family-18 oz.■100■
4     100-20-16■100-20■Family-24 oz.■100■
```

	LEVEL0,Product	LEVEL1,Product	DUPLEVEL1,Product	LEVEL2,Product
1	100-10-12	100-10	Family-18 oz.	100
2	100-10-16	100-10	Family-24 oz.	100
3	100-20-12	100-20	Family-18 oz.	100
4	100-20-16	100-20	Family-24 oz.	100
5				

Figure 12-5, Rules File for Level Build

Figure 12-6 shows the tree Essbase builds from this data source and rules file.

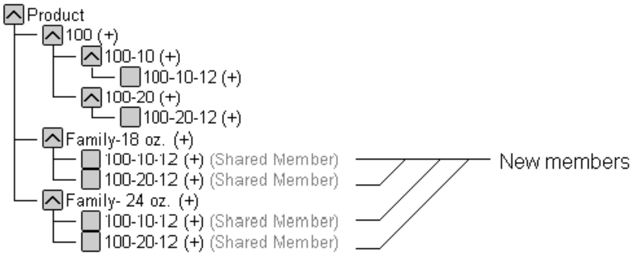

Product
100 (+)
100-10 (+)
100-10-12 (+)
100-20 (+)
100-20-12 (+)
Family-18 oz. (+)
100-10-12 (+) (Shared Member)
100-20-12 (+) (Shared Member)
Family- 24 oz. (+)
100-10-12 (+) (Shared Member)
100-20-12 (+) (Shared Member)
New members

Figure 12-6, 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 fields are Alias, Attribute, Formula, DuplicateLevel, DuplicateLevelAlias, CurrencyName, CurrencyCategory, and User-DefinedAttribute.

  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.

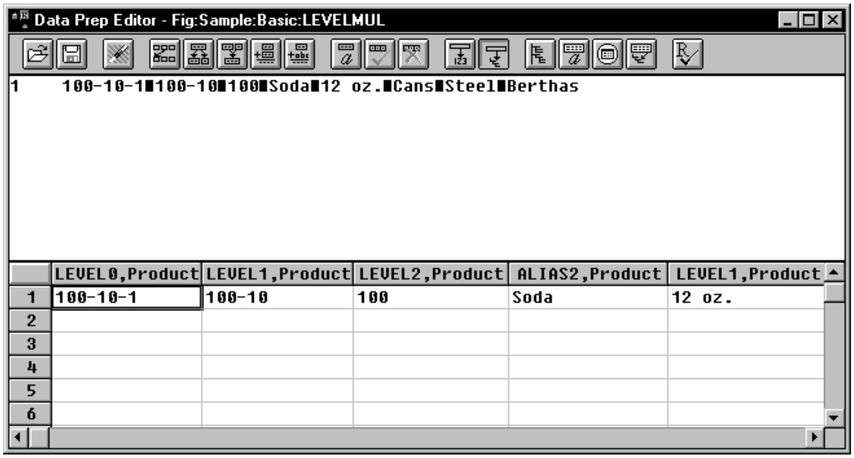
- If the null occurs where Essbase expects a secondary field, Essbase ignores that field and continues loading. For example:

  ```
  LEVEL0,Products  ALIAS0, Products  LEVEL1,Products  LEVEL2,Products
  100-10a          100-10                             100
  ```

  When Essbase reads the above file, it ignores the ALIAS0 field.

Building Multiple Roll-Ups Using Level References

You can also put shared members at different levels in the outline using the Level References build method. The following data file, `LEVELMUL.TXT` and rules file, `LEVELMUL.RUL`, in Figure 12-7 specifies the build instructions for levels in the Product dimension:

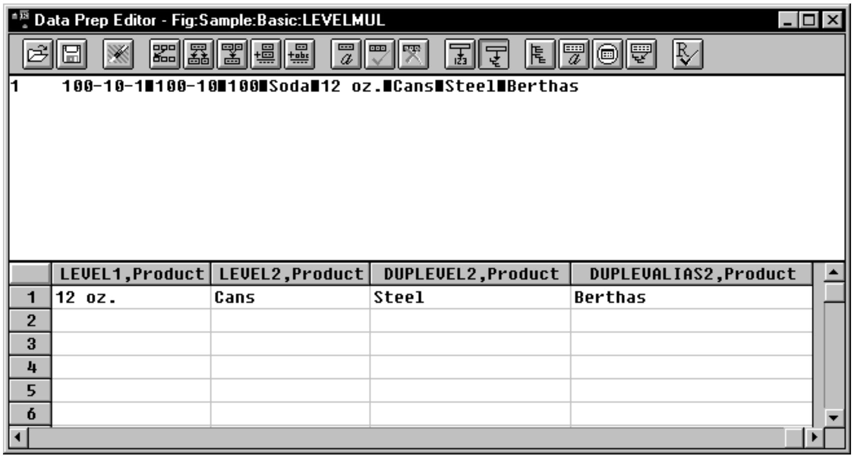


```
1     100-10-1■100-10■100■Soda■12 oz.■Cans■Steel■Berthas
```

	LEVEL0,Product	LEVEL1,Product	LEVEL2,Product	ALIAS2,Product	LEVEL1,Product
1	100-10-1	100-10	100	Soda	12 oz.
2					
3					
4					
5					
6					

Figure 12-7, Rules File Fields Set to Build Multiple Roll-Ups Using Level References

Because the record is so long, this second graphic shows the rules file after it has been scrolled to the left to show the extra members:



```
1     100-10-1■100-10■100■Soda■12 oz.■Cans■Steel■Berthas
```

	LEVEL1,Product	LEVEL2,Product	DUPLEVEL2,Product	DUPLEVALIAS2,Product
1	12 oz.	Cans	Steel	Berthas
2				
3				
4				
5				
6				

Figure 12-8, Scrolled Window

When you load the data in Figure 12-7, Essbase builds the following member tree:

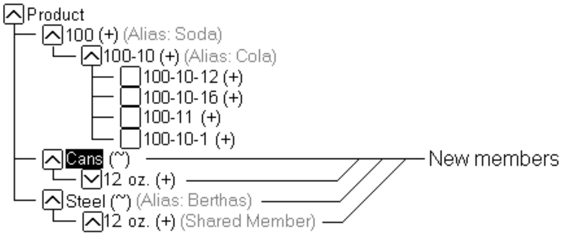


Figure 12-9, Multiple Roll-Ups

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 12-10 shows part of the Product dimension with its parent and children relationships identified.

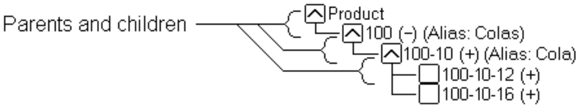
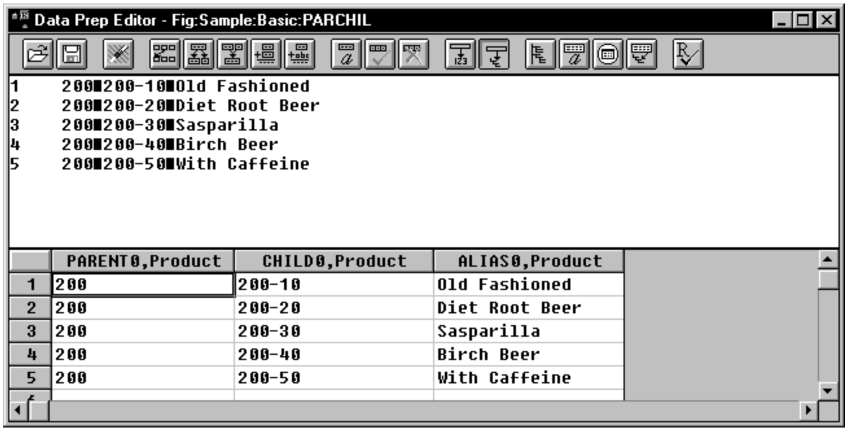


Figure 12-10, Parents and Children

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 or its attribute). 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 “Setting Field Types” on page 13-9. For example, the top half of Figure 12-11 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 the child:

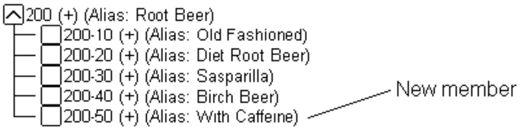


```
1     200■200-10■Old Fashioned
2     200■200-20■Diet Root Beer
3     200■200-30■Sasparilla
4     200■200-40■Birch Beer
5     200■200-50■With Caffeine
```

	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 12-11, Rules Files for Parent/Child Build

Figure 12-12 shows the tree that Essbase builds from this data source and rules file.



200 (+) (Alias: Root Beer)
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)

New member

Figure 12-12, Parents and Children

Creating Shared Roll-Ups from Multiple Data Sources Using Parent/Child References

If you are building dimensions using more than one data source at a time and want to create multiple roll-ups, load the first data source using the most appropriate build method. Load all other data sources using the Parent/Child References build method. Make sure that **Do Not Share** is not checked so that Essbase creates duplicate members as shared when it encounters them under a new parent.

For example, if you have the data source in Figure 12-13:

```
"Soft Drinks"    Cola
"Soft Drinks"    "Root Beer"
Cola             TBC
"Root Beer"      Grandma’s
```

Figure 12-13, Soft Drinks Data Source

Essbase builds the following tree:

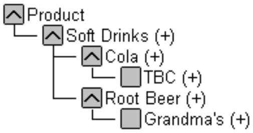


Figure 12-14, Soft Drinks Tree

Now, if your second data source contains the following information:

```
Vendor    TBC
Vendor    Grandma’s
```

Figure 12-15, Second Shared Roll-Ups Data Source

To create a member tree where TBC and Grandma’s 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 checked so that Essbase creates duplicate members as shared when it encounters them under a new parent.

Essbase builds the following tree:

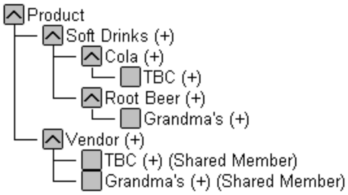


Figure 12-16, Shared Roll-Ups Tree

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 choose your build method, you must specify the dimension to which each field maps. See "Setting Field Types" on page 13-9.

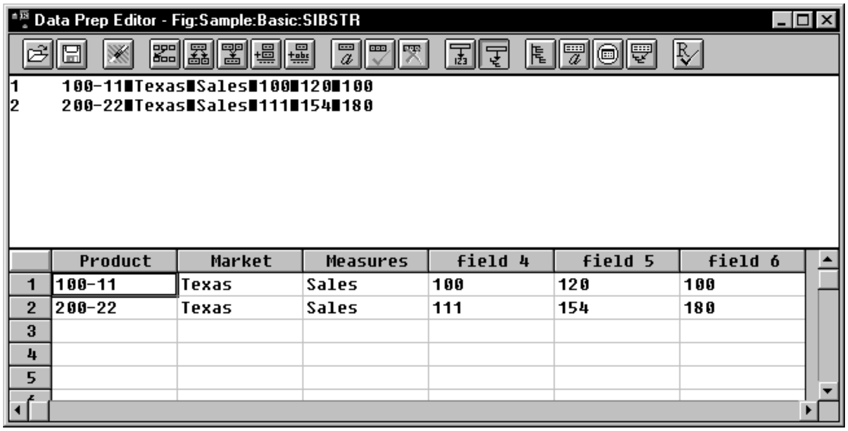
After Essbase has added all the new members to the outline, use the Outline Editor to examine the outline. If necessary, move the new members into their correct positions. See Chapter 8, Creating and Changing Database Outlines.

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 12-17, `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 dimension for each field to match the rules file, SIBSTR.RUL, in Figure 12-17.



```
1    100-11■Texas■Sales■100■120■100
2    200-22■Texas■Sales■111■154■180
```

	Product	Market	Measures	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						

Figure 12-17, Rules File Fields Set to Add Members as Siblings with String Matches

When you load the data in Figure 12-17, Essbase builds the following member tree:

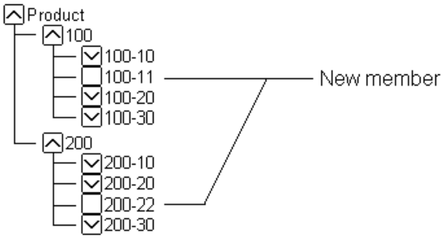


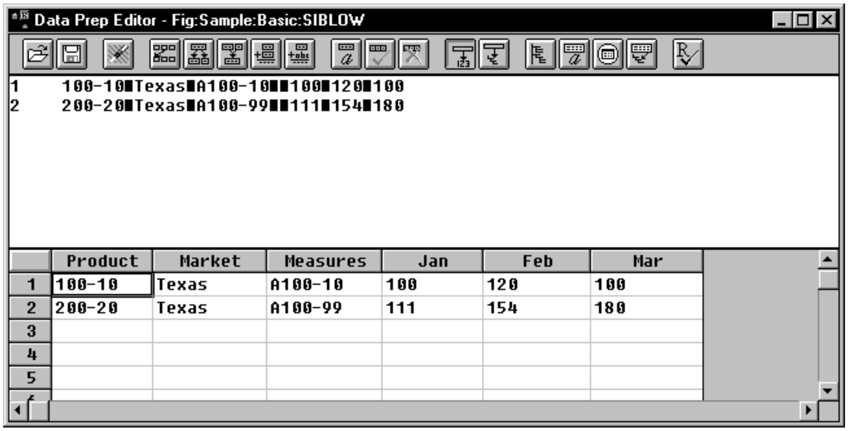
Figure 12-18, Adding Members as Siblings with String Matches Tree

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 12-19 contains new members (A100-10 and A100-99) to add to the Measures dimension in the Sample Basic database.



```
100-10■Texas■A100-10■■100■120■100
200-20■Texas■A100-99■■111■154■180
```

	Product	Market	Measures	Jan	Feb	Mar
1	100-10	Texas	A100-10	100	120	100
2	200-20	Texas	A100-99	111	154	180
3						
4						
5						

Figure 12-19, Rules File Fields Set to Add Members as Siblings of the Lowest Level

When you load the data in Figure 12-19, Essbase builds the following member tree:

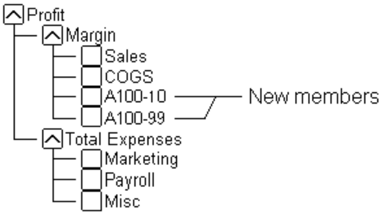


Figure 12-20, Adding Members as Siblings of the Lowest Level

Adding All New Members to a Specified Parent

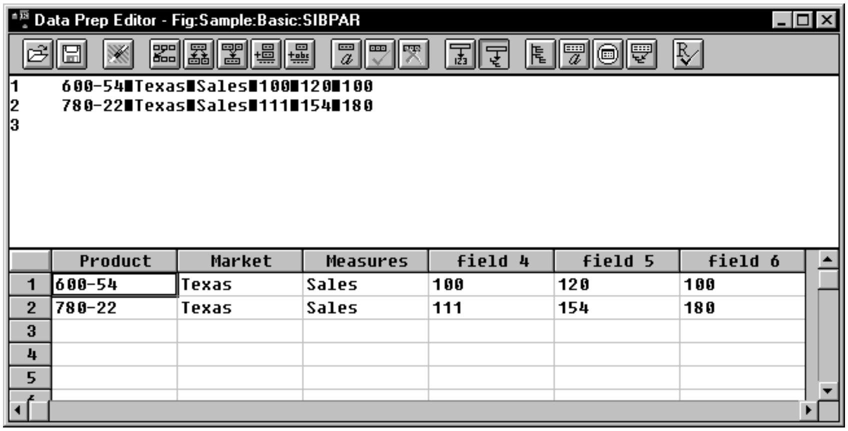
You can add all new members as children of a specified parent, generally a "dummy" parent. When Essbase adds all new members to the outline, review the added members and move or delete them in the Outline Editor.

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 12-21, 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:

- In the **Field Attributes** dialog box for the Product column (field 1):
  - 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 12-21.
- 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.



```
1    600-54■Texas■Sales■100■120■100
2    780-22■Texas■Sales■111■154■180
3
```

	Product	Market	Measures	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						

Figure 12-21, Rules File Fields Set to Add Members as a Child of a Specified Parent

When you load the data in Figure 12-21, Essbase builds the following member tree:

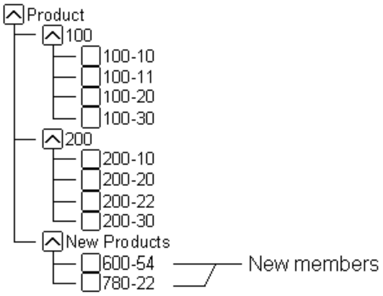


Figure 12-22, Adding Members as a Child of a Specified Parent

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.

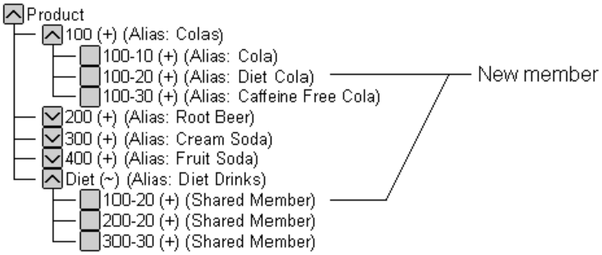

Product
100 (+) (Alias: Colas)
100-10 (+) (Alias: Cola)
100-20 (+) (Alias: Diet Cola)
100-30 (+) (Alias: Caffeine Free Cola)
200 (+) (Alias: Root Beer)
300 (+) (Alias: Cream Soda)
400 (+) (Alias: Fruit Soda)
Diet (~) (Alias: Diet Drinks)
100-20 (+) (Shared Member)
200-20 (+) (Shared Member)
300-30 (+) (Shared Member)
New member

Figure 12-23, Shared Members in the Sample Basic Database

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.

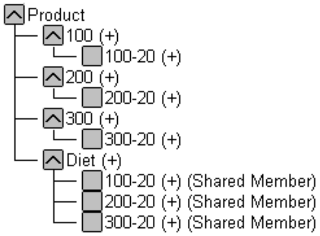
You can share members at multiple generations in the outline. In Figure 12-23, Diet Cola is shared by two members at Generation 2 in the outline, but it could be shared by a member at Generation 2 and a member at Generation 3 as in Figure 12-31.

While creating shared members at different generations in the outline is easy in the 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.



```
Product
  100 (+)
    100-20 (+)
  200 (+)
    200-20 (+)
  300 (+)
    300-20 (+)
  Diet (+)
    100-20 (+) (Shared Member)
    200-20 (+) (Shared Member)
    300-20 (+) (Shared Member)
```

Figure 12-24, Members Shared at the Same Generation

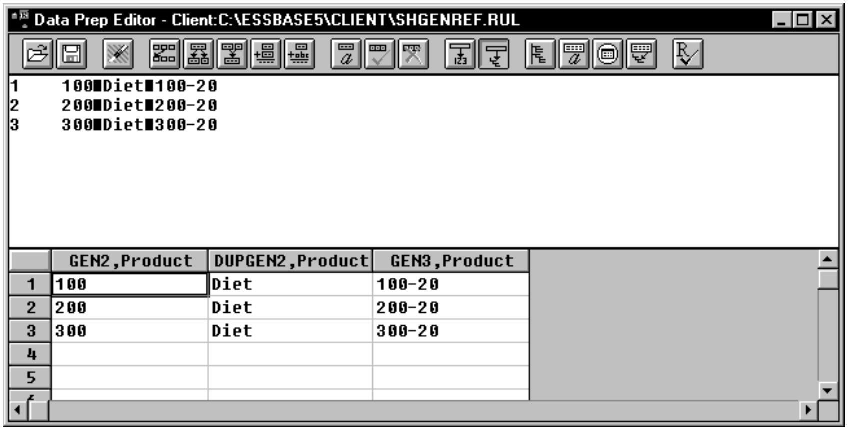
This is the simplest way to share members. You can share members at the same generation using the following build methods:

- “Creating Shared Members Using Generation References” on page 12-20
- “Creating Shared Members Using Level References” on page 12-21
- “Creating Shared Members Using Parent/Child References” on page 12-22

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, and 300-20 members, use the sample file, SHGENREF.TXT, and set up the rules file so that the fields look like SHGENREF.RUL, shown in Figure 12-25. 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.



Data Prep Editor - Client:C:\ESSBASE5\CLIENT\SHGENREF.RUL

```
1    100■Diet■100-20
2    200■Diet■200-20
3    300■Diet■300-20
```

	GEN2,Product	DUPGEN2,Product	GEN3,Product
1	100	Diet	100-20
2	200	Diet	200-20
3	300	Diet	300-20
4			
5			

Figure 12-25, Sample Generation Shared Member Rules File

This builds the following tree:

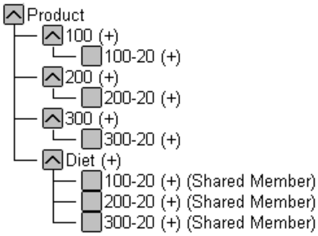

Product
100 (+)
100-20 (+)
200 (+)
200-20 (+)
300 (+)
300-20 (+)
Diet (+)
100-20 (+) (Shared Member)
200-20 (+) (Shared Member)
300-20 (+) (Shared Member)

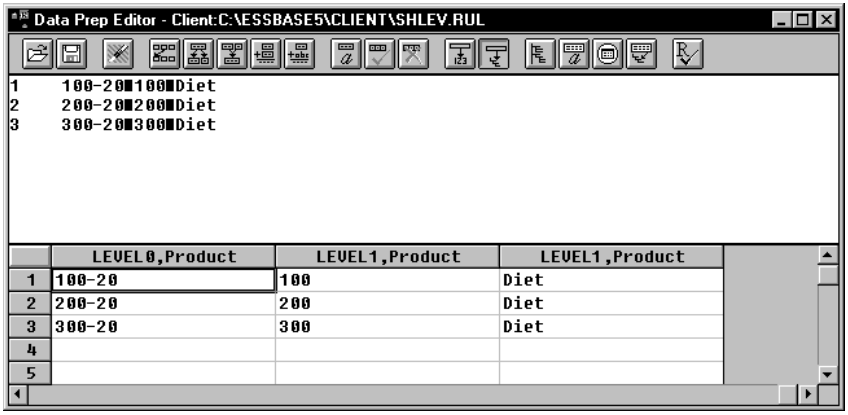
Figure 12-26, Sample Generation Shared Member Rules Tree

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), and 300-20 (Diet Cream 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 12-27.



Data Prep Editor - Client:C:\ESSBASE5\CLIENT\SHLEV.RUL

```
1    100-20■100■Diet
2    200-20■200■Diet
3    300-20■300■Diet
```

	LEVEL0,Product	LEVEL1,Product	LEVEL1,Product
1	100-20	100	Diet
2	200-20	200	Diet
3	300-20	300	Diet
4			
5			

Figure 12-27, Sample Level Shared Member Rules File

This builds the following tree:

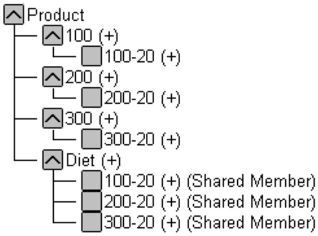
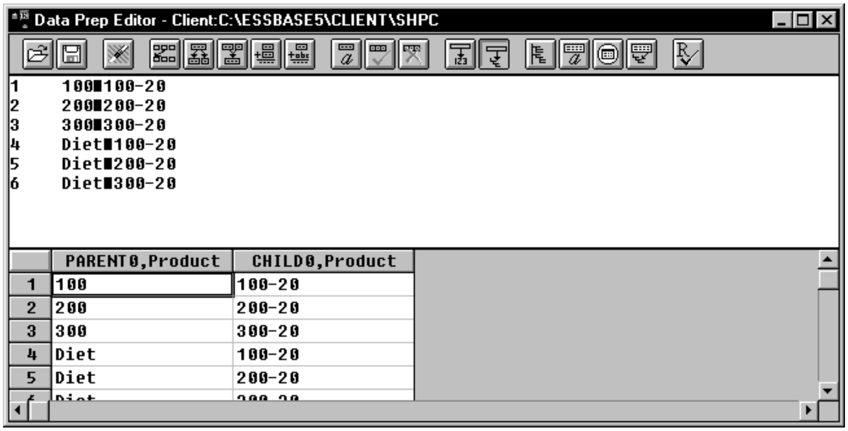

Product
100 (+)
100-20 (+)
200 (+)
200-20 (+)
300 (+)
300-20 (+)
Diet (+)
100-20 (+) (Shared Member)
200-20 (+) (Shared Member)
300-20 (+) (Shared Member)

Figure 12-28, Sample Level Shared Member Rules Tree

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 not checked in the **Dimension Build Settings** page of the **Dimension Build Settings** dialog box. When **Do Not Share** is not checked, Essbase automatically creates duplicate members under a new parent as shared members.



Data Prep Editor - Client:C:\ESSBASE5\CLIENT\SHPC

```
1     100■100-20
2     200■200-20
3     300■300-20
4     Diet■100-20
5     Diet■200-20
6     Diet■300-20
```

	PARENT0,Product	CHILD0,Product
1	100	100-20
2	200	200-20
3	300	300-20
4	Diet	100-20
5	Diet	200-20
6	Diet	300-20

Figure 12-29, Sample Parent/Child Shared Members Rules File

This builds the following tree:

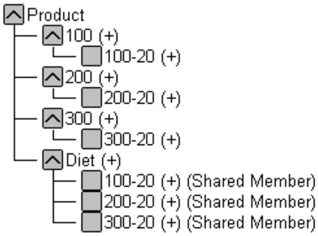
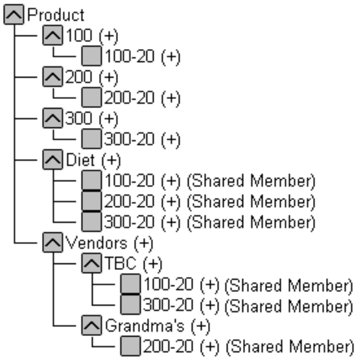

Product
100 (+)
100-20 (+)
200 (+)
200-20 (+)
300 (+)
300-20 (+)
Diet (+)
100-20 (+) (Shared Member)
200-20 (+) (Shared Member)
300-20 (+) (Shared Member)

Figure 12-30, Sample Parent/Child Shared Member Rules Tree

Sharing Members at Different Generations

Sometimes you want shared members to roll up into parents at different generations in the outline. In Figure 12-31, 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.



```
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)
```

Figure 12-31, Members Shared at Different Generations

To share members across parents at different generations in the outline, use the following build methods:

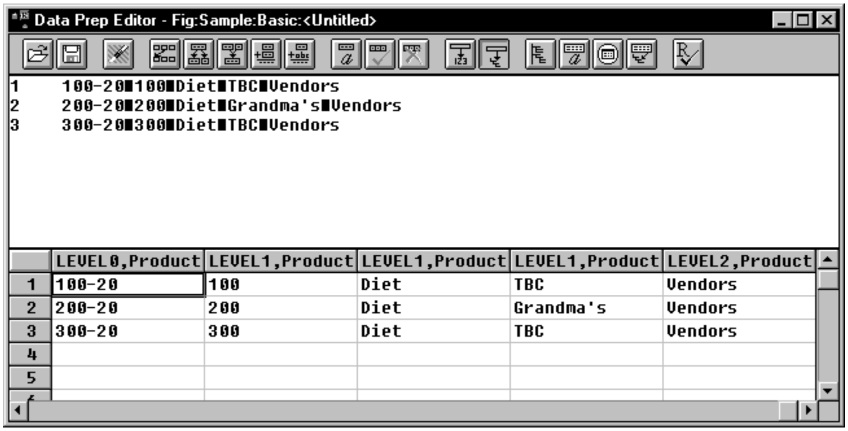
- "Creating Shared Members Using Level References" on page 12-24
- "Creating Shared Members Using Parent/Child References" on page 12-25

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. 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 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 12-32.



```
1     100-20■100■Diet■TBC■Vendors
2     200-20■200■Diet■Grandma's■Vendors
3     300-20■300■Diet■TBC■Vendors
```

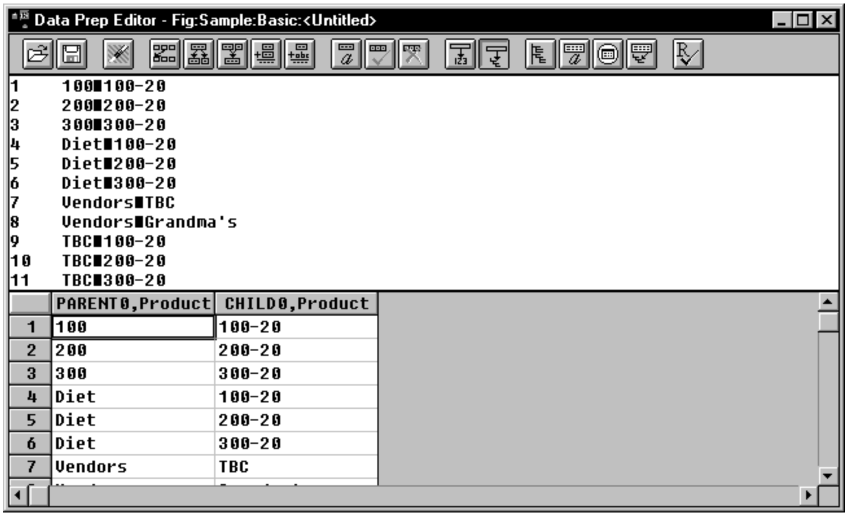
	LEVEL0,Product	LEVEL1,Product	LEVEL1,Product	LEVEL1,Product	LEVEL2,Product
1	100-20	100	Diet	TBC	Vendors
2	200-20	200	Diet	Grandma's	Vendors
3	300-20	300	Diet	TBC	Vendors
4					
5					

Figure 12-32, Level References Sample Rules File for Shared Members at Different Generations

This builds the tree in Figure 12-31.

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 checked 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.



```
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
TBC■200-20
TBC■300-20
```

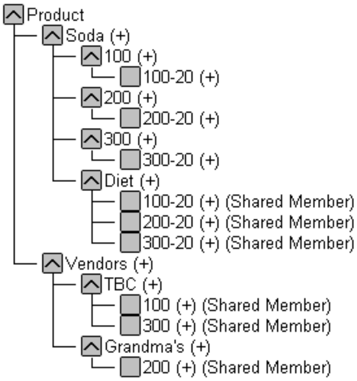
	PARENT0,Product	CHILD0,Product
1	100	100-20
2	200	200-20
3	300	300-20
4	Diet	100-20
5	Diet	200-20
6	Diet	300-20
7	Vendors	TBC

Figure 12-33, Parent/Child References Sample Rules File for Shared Members at Different Generations

This builds the tree in Figure 12-31.

Sharing Members with Branches

Sometimes you want to share non-leaf members (members that are not at the lowest generation). In Figure 12-34 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.



```
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)
```

Figure 12-34, Members with Branches Shared at Different Generations

To share members in the outline with branches below them, use the following build methods:

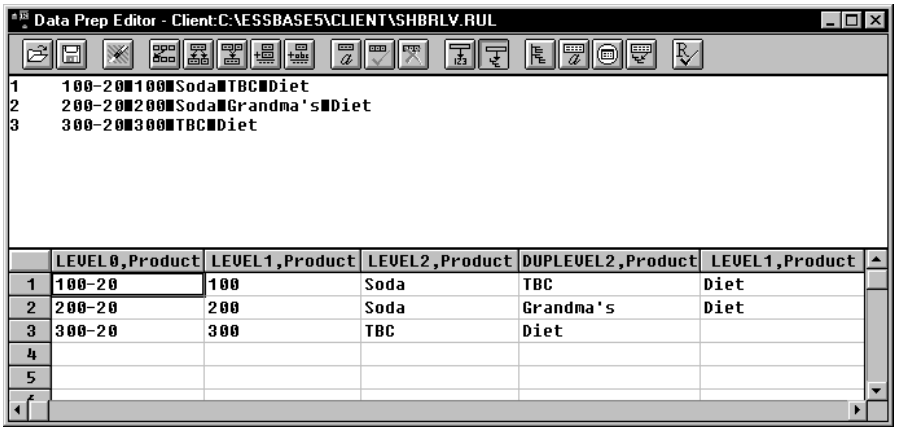
- "Creating Shared Members Using Level References" on page 12-27
- "Creating Shared Members Using Parent/Child References" on page 12-28

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 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 Figure 12-35. This data source and rules file only work if the 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.



```
1     100-20■100■Soda■TBC■Diet
2     200-20■200■Soda■Grandma's■Diet
3     300-20■300■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	TBC	Diet	
4					
5					

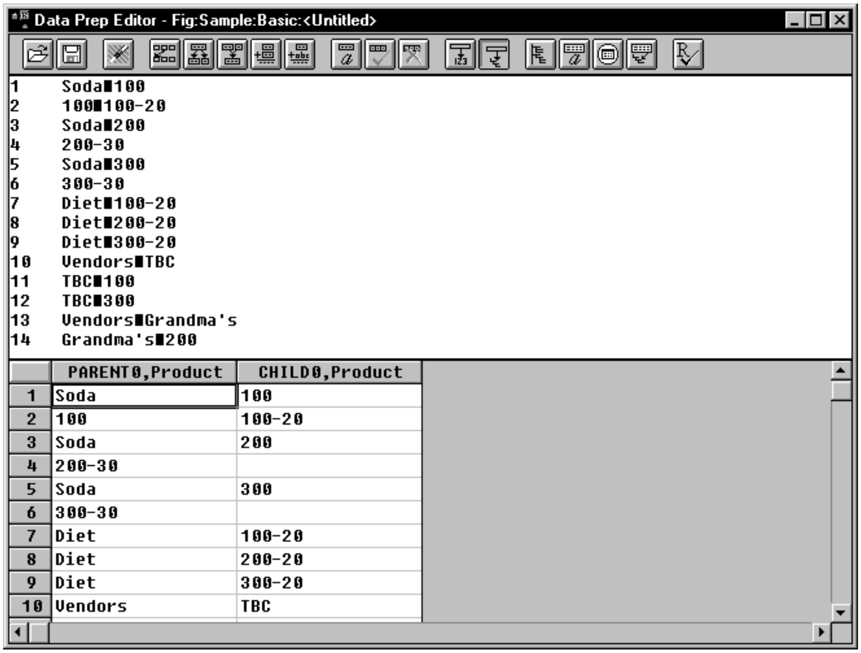
Figure 12-35, Level References Sample Rules File for Shared Members at Different Generations with Branches

This builds the tree in Figure 12-34.

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 checked 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.

Note: 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.



Data Prep Editor - Fig:Sample:Basic:<Untitled>

```
1    Soda■100
2    100■100-20
3    Soda■200
4    200-30
5    Soda■300
6    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	200-30	
5	Soda	300
6	300-30	
7	Diet	100-20
8	Diet	200-20
9	Diet	300-20
10	Vendors	TBC

Figure 12-36, Parent/Child Sample Rules File for Shared Members at Different Generations with Branches

This builds the tree in Figure 12-34.

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.

- **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.
- **Multi-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 16, Managing Security at Global and User Levels.

Chapter 13 Building Dimensions Using a Rules File

This chapter describes how to build dimensions using a rules file in Hyperion Essbase OLAP Server. For background information on dynamic dimension building, see Chapter 12, Introducing Dynamic Dimension Building.

About Dimensions and Rules Files

You can build dimensions dynamically in the following ways:

- Create a rules file using the **Dimension Build Settings** dialog box in the Data Prep Editor by defining new dimensions, specifying changes to existing dimensions, setting global options, and validating the dimension build rules.
- Manipulate the data source and create a dynamic reference in the rules file.

Then 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 some debugging tips.

Use the BUILDDIM and INCBUILDDIM commands in ESSCMD to build dimensions dynamically. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Step 1: Defining Dimensions

To define a dimension:

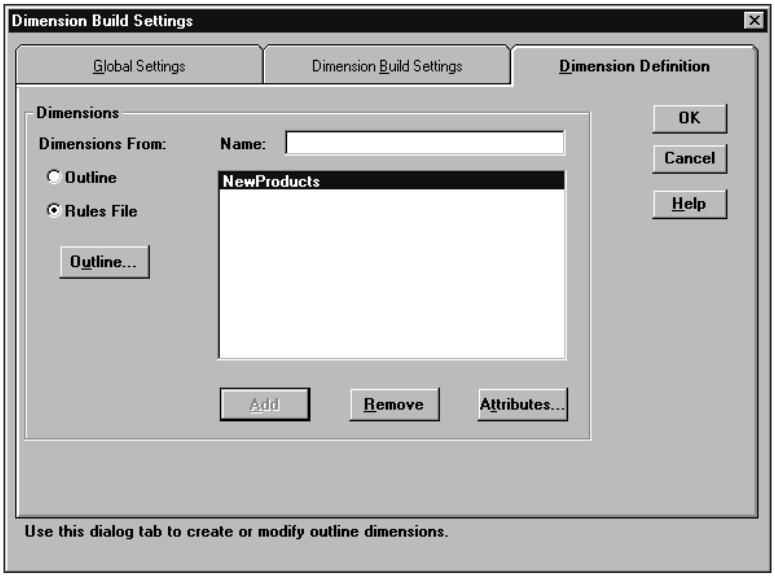
- Name new dimensions.
- Specify whether a dimension comes from the outline or a rules file.
- Set the attributes for new dimensions or change the attributes of existing dimensions.

Naming New Dimensions

To name a new dimension:

1. Choose the application and database in the Application Desktop window.
2. 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.
3. Choose View | Dimension Building fields or click the **Dimension Build** button , to make sure that the Data Prep Editor is displaying dimension building fields and not data load fields.

4. Choose Options | Dimension Build Settings to open the **Dimension Build Settings** dialog box. Click the **Dimension Definition** tab.



Dimensions From:

○ Outline

● Rules File

Name:

NewProducts

Use this dialog tab to create or modify outline dimensions.

Figure 13-1, Dimension Definition Page

5. Choose **Rules File** to indicate that the dimension is defined in the rules file.

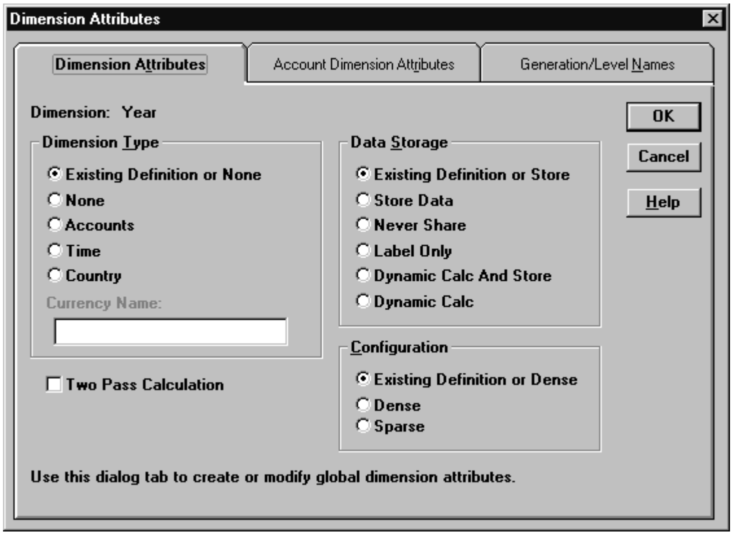
6. Enter the name of the new dimension, such as `NewProducts`, and click **Add** to add it to the end of the outline. Dimension names can be up to 79 characters long.

Setting Dimension Attributes

To set the attributes of a new dimension or modify the attributes of an existing dimension:

1. Choose Options | Dimension Build Settings to open the **Dimension Build Settings** dialog box.
2. 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.

   If the dimension is new, click **Rules File**. The list box displays the new dimensions you named.
3. Click the dimension name in the list box.
4. Click **Attributes** to open the **Dimension Attributes** dialog box.
   - Set the **Dimension Type**.
   - Choose the **Data Storage** attributes.
   - Choose a **Dense** or **Sparse** configuration.
   - If you're not sure what settings to use, click Help.



Dimension: Year

Dimension Type
● Existing Definition or None
○ None
○ Accounts
○ Time
○ Country
Currency Name:

□ Two Pass Calculation

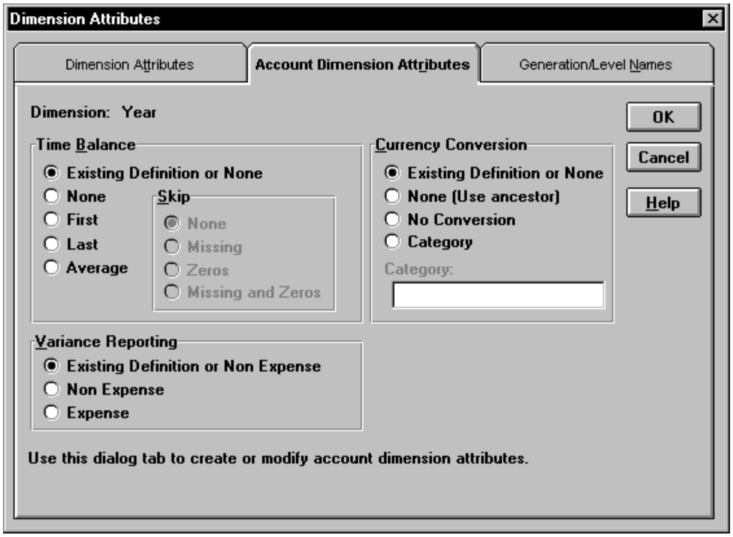
Data Storage
● Existing Definition or Store
○ Store Data
○ Never Share
○ Label Only
○ Dynamic Calc And Store
○ Dynamic Calc

Configuration
● Existing Definition or Dense
○ Dense
○ Sparse

Use this dialog tab to create or modify global dimension attributes.

Figure 13-2, Dimension Attributes Page

5. For an Accounts dimension, click the **Account Dimension Attributes** tab.
   - Set the Accounts dimension attributes and click OK.
   - If you're not sure what settings to use, click Help.



Dimension Attributes

Dimension Attributes | Account Dimension Attributes | 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

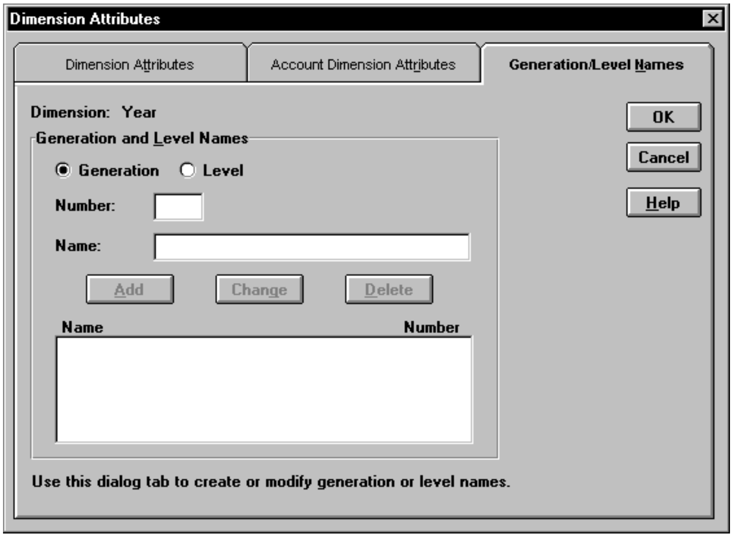
Category:

Variance Reporting
- ● Existing Definition or Non Expense
- ○ Non Expense
- ○ Expense

Use this dialog tab to create or modify account dimension attributes.

Figure 13-3, Account Dimension Attributes Page

6. To name the generations and levels in the current dimension, click the **Generation/Level Names** tab.
   - Set the generation/level names and click OK.
   - If you're not sure what settings to use, click Help.



Dimension Attributes

Dimension: Year

Generation and Level Names

● Generation ○ Level

Number:

Name:

Use this dialog tab to create or modify generation or level names.

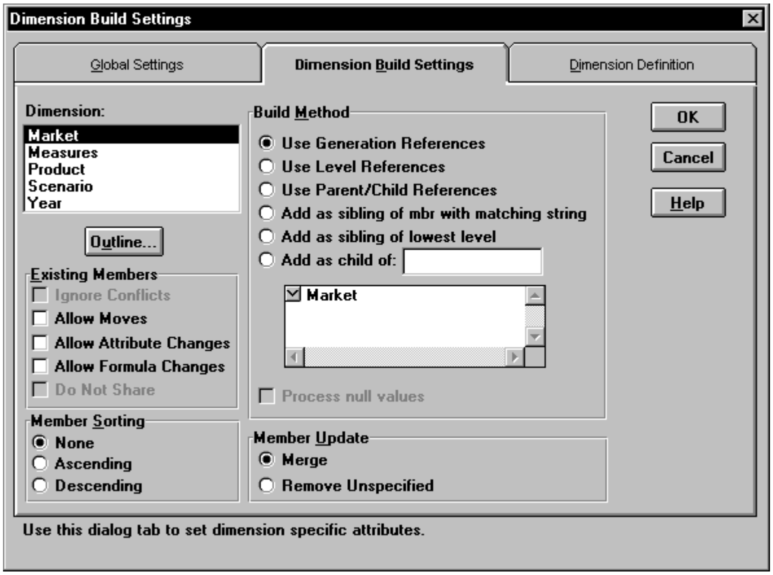
Figure 13-4, Generation/Level Names Page

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: Specifying Changes to Existing Dimensions

To specify the changes that you will allow Essbase to make to dimensions in the outline:

1. Click the **Dimension Build Settings** tab of the **Dimension Build Settings** dialog box.



Dimension Build Settings

Global Settings | Dimension Build Settings | Dimension Definition

Dimension:
Market
Measures
Product
Scenario
Year

Outline...

Existing Members
□ Ignore Conflicts
□ Allow Moves
□ Allow Attribute Changes
□ Allow Formula Changes
□ Do Not Share

Member Sorting
● None
○ Ascending
○ Descending

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:
☑ Market
□ Process null values

Member Update
● Merge
○ Remove Unspecified

OK
Cancel
Help

Use this dialog tab to set dimension specific attributes.

Figure 13-5, Dimension Build Settings Page

2. Choose 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 any new dimensions defined in the rules file.

3. Choose the types of changes to allow to existing members in the outline:

Check the option...	When you want to...
Ignore Conflicts (valid only with the Add as... build methods)	Ignore member names that already exist in the outline under different dimensions.

Check the option...	When you want to...
Allow Moves	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 the Outline Editor.
Allow Attribute Changes	Allow changes to the attributes, user-defined attributes, and aliases of existing members.
Allow Formula 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.
Do Not Share (valid only with the Parent/child references build method)	Reject records that specify a new parent for an existing member. If you do not check this box, each time a member is repeated with another parent, it is created as a shared member.

4. To sort the members of a dimension after building it, choose either **Ascending** (A to Z) or **Descending** order (Z to A). To leave the members unsorted, choose **None**.
5. Choose the build method from the **Build Method** box. If you are not sure which method to use, see “Introduction to Build Methods” on page 12-3. The following build methods require that you specify additional information:

Build Method	What to Specify
Add as child of	The parent to which the new members are added.
Use Generation References	Whether to use null processing. See “Null Processing with Generation References” on page 12-6.
Use Level References	Whether to use null processing. See “Null Processing with Level References” on page 12-9.

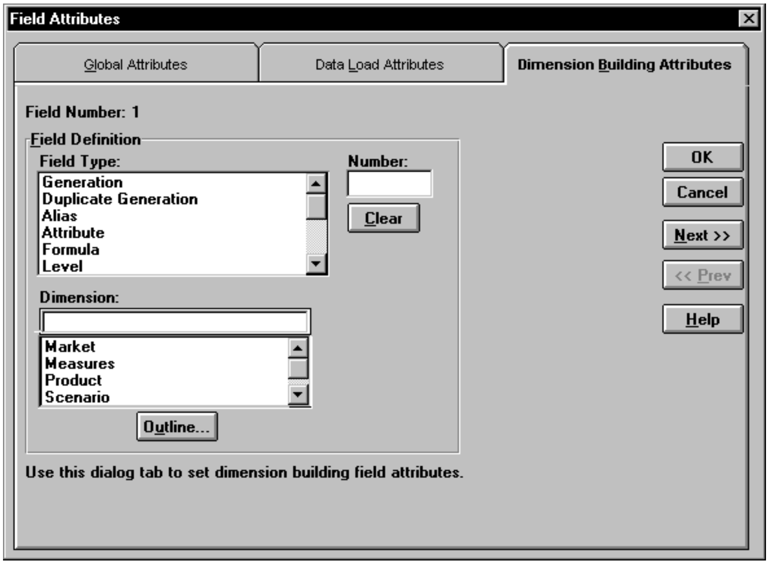
6. By default, Essbase merges new members found in the data source into the dimension. To remove existing members if Essbase does not encounter them in the data source, check **Remove Unspecified**.
7. Click OK.

Setting Field Types

Setting the field type tells Essbase what kind of field to expect, such as a Generation field or an Alias. You must use field types that are valid for the build method you choose.

To set field types:

1. Select the field for which to set the field type.
2. Choose Field | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Dimension Building Attributes** tab.



Field Number: 1

Field Type:
Generation
Duplicate Generation
Alias
Attribute
Formula
Level

Dimension:
Market
Measures
Product
Scenario

Use this dialog tab to set dimension building field attributes.

Figure 13-6, Dimension Building Attributes Page

3. Choose the field type from the **Field Type** list box. The following table lists valid field types for each build method.

Build Method	Valid Field Types	Specifies That the Field Contains...
Level references	Levels	Name of member in a level.
	DuplicateLevel	Name of member that has duplicate parents; that is, a member that is shared by more than one parent.
	DuplicateLevelAlias	Alias for the new parent of the shared member as the member is created.
Generation references	Generation	Name of a member in a generation.
	DuplicateGeneration	Name of a member that has duplicate parents; that is, a member that is shared by more than one parent.
	DuplicateGenerationAlias	Alias for the new parent of the shared member as the member is created.
Parent/child references	Parent	Name of a parent.
	Child	Name of a child.
Level, generation, and parent/child references	Alias	An alias.
	Attribute	An attribute.
	Formula	A formula.
	Currency Name	A currency name.
	Currency Category	A currency category.
	User-Defined Attribute	A user-defined attribute.

4. Enter the generation or level number in the **Number** text box. The generation or level number must correspond to the generation or level of the specified member in the outline.

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 check the **Allow Moves** box. See "Step 2: Specifying Changes to Existing Dimensions" on page 13-7. • 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.
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, you could have a valid data source with: `GEN2, Dimension 2  GEN3, Dimension 2  GEN4, Dimension 2`

5. Enter the dimension the field corresponds to, or choose it from the **Dimension** list. If the **Dimension** list is empty, click the **Outline** button to associate the rules file with an outline.

6. To move to the next field, click **Next**.

7. When you are finished setting the field types, click OK.

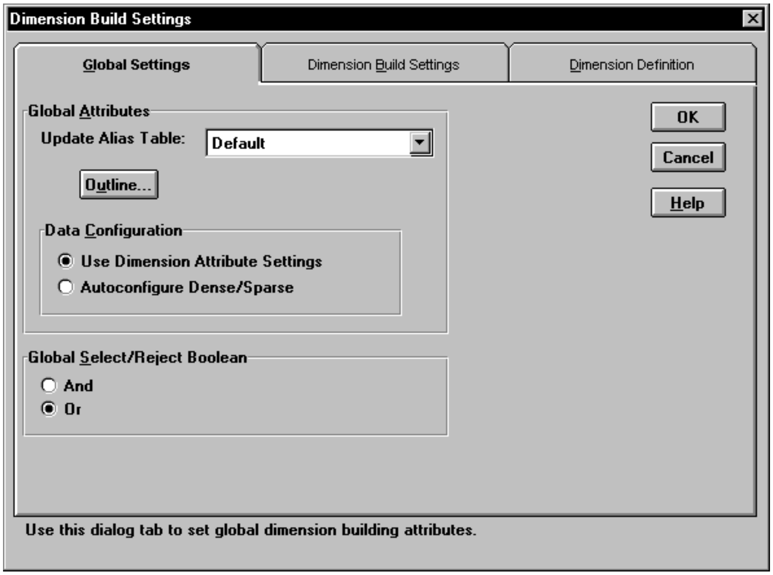
Step 3: Setting Global Options

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/reject criteria between fields

To set global build options:

1. Click the **Global Settings** tab of the **Dimension Build Settings** dialog box.



Dimension Build Settings

Global Settings | Dimension Build Settings | Dimension Definition

Global Attributes

Update Alias Table: Default

Data Configuration

● Use Dimension Attribute Settings

○ Autoconfigure Dense/Sparse

Global Select/Reject Boolean

○ And

● Or

Use this dialog tab to set global dimension building attributes.

Figure 13-7, Global Settings Page

2. Choose 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. Choose either:

   - The **Use Dimension Attribute 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 4-1.

   When you change the configuration settings, Essbase restructures the database.

4. Choose 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 20-16.

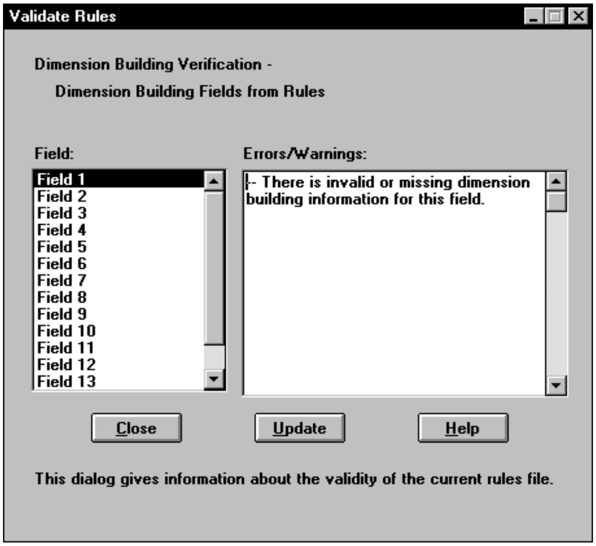
5. Click OK to save the changes.

Step 4: 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 the Data Prep Editor is in dimension building mode.

2. Choose 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.



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.

Figure 13-8, Validate Rules Dialog Box

If the rules file is correct, you can use it perform a dimension build. For more information, see "Performing Dimension Builds" on page 13-18.

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 13-8, 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.

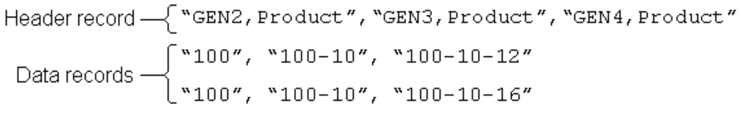
Manipulating the Data Source

You can also build dimensions or change the attributes 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 attributes

Using Dynamic References

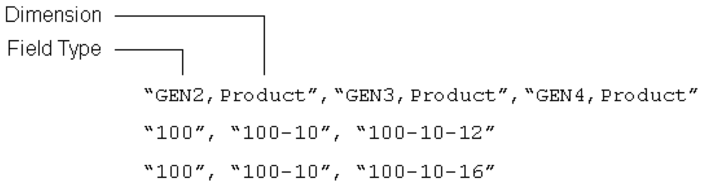
You can dynamically build dimensions by adding header information to the top of the data source and specifying the location of the header information in the rules file as a dynamic reference. Figure 13-9 contains an example of a header record.


Header record
Data records

```
"GEN2,Product","GEN3,Product","GEN4,Product"
"100", "100-10", "100-10-12"
"100", "100-10", "100-10-16"
```

Figure 13-9, Header Record

The header record lists the field type for each field and the dimension name into which to load the fields. The header record must be in the following format:



```
“GEN2,Product”,“GEN3,Product”,“GEN4,Product”
“100”, “100-10”, “100-10-12”
“100”, “100-10”, “100-10-16”
```

Figure 13-10, Field Types in Header Records

Note: Member and alias names can be up to 79 characters long.

After you set the header record in the data source, you must specify the location of the header record(s) in the rules file using a dynamic reference. If the 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 and are:

- GEN, DUPGEN, and DUPGENALIAS
- LEVEL, DUPLEVEL, and DUPLEVELALIAS
- PARENT, CHILD
- ATTRIB
- ALIAS
- FORMULA
- CURNAME
- CURCAT
- USERATTRIB

Once you set the field type, you must also type in the field number. For more information on field numbers, see “Setting Field Types” on page 13-9.

Note: If the file delimiter is a comma, enclose the field type in quotes.

Setting Member Attributes

You can modify the attributes of both new and existing members during a dimension build by setting the attributes directly in the data source. Put attributes 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 attributes field to `Attribute`. To set the field type to attribute, see "Manipulating the Data Source" on page 13-15.

The following table lists all member codes used to assign attributes 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	Include 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 update dimensions in the database in the following ways:

- Using the **Data Load** dialog box. You must have at least one dimension defined in the database. For more information on loading data, see Chapter 22, Performing a Data Load.
- In batch mode using ESSCMD. You must have at least one dimension defined in the database. In ESSCMD, you can also do multi-pass dimension builds.

Use the BUILDDIM command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

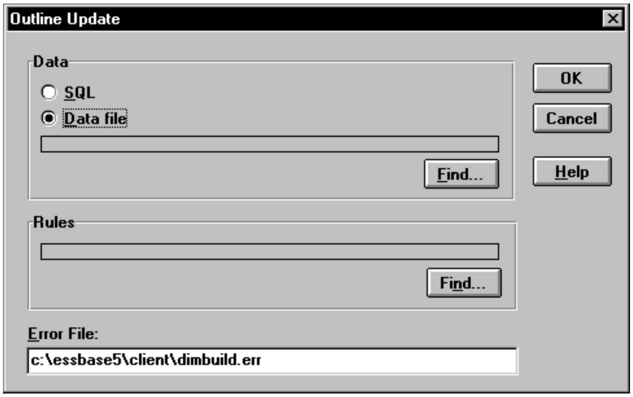
- Using the Outline Editor. For more information, see “Updating Dimensions in the Outline Editor” on page 13-18.

Updating Dimensions in the Outline Editor

The outline must have at least one dimension defined before you can do a dynamic dimension build. Define this dimension using the Outline Editor. Then you can start building dimensions dynamically.

To update dimensions using the Outline Editor:

1. Open the outline in the Outline Editor. See “Opening Outlines” on page 8-2.
2. Choose File | Update Outline to open the **Outline Update** dialog box.



Outline Update

Data

○ SQL

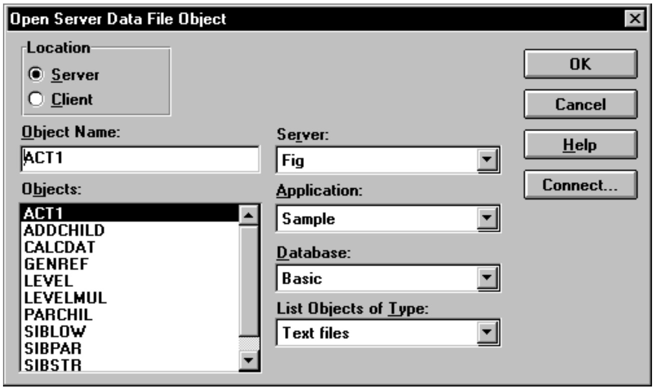
● Data file

Rules

Error File: c:\essbase5\client\dimbuild.err

Figure 13-11, Outline Update Dialog Box

3. Choose the kind of data source to load 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 chose **Data File**, choose the data source to load by clicking **Find**. The **Open Server Data File Object** dialog box appears.



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

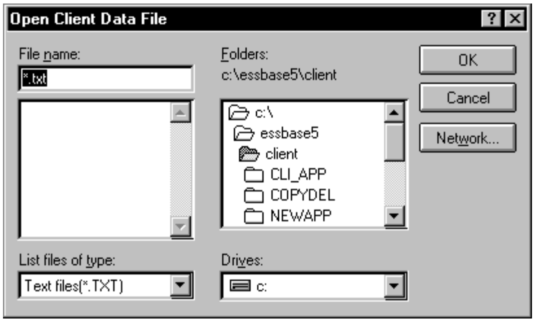
Figure 13-12, Open Server Data File Object Dialog Box

6. Make sure the appropriate Essbase server, application, and database are selected from their respective lists.

   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.

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. Choose 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.



Open Client Data File

File name: *.txt

Folders: c:\essbase5\client

List files of type: Text files(*.TXT)

Drives: c:

Figure 13-13, Open Client Data File Dialog Box

7. Choose the dimension build rules file to load by clicking **Find** and then selecting it 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 couldn't 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. Do the dimension build using the 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 described in “Finishing the Data Load or Dimension Build” on page 22-10. If errors occurred during dimension building, Essbase logs them in the `\ESSBASE\CLIENT\DIMBUILD.ERR` file.

- 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.

To fix the data source:

1. Open the error log located in the file `\ESSBASE\CLIENT\DIMBUILD.ERR`.

Note: `\ESSBASE` is the default directory specified during installation. You may have set this directory differently.

2. Browse through the error log file. It contains a list of each of the data sources and records that didn’t load. Figure 13-14 is the error log that Essbase created while trying to load the `Act1` file.

```
\\ Member  Sales Not Found In Database
California,Caffeine Free Cola, Sales,
145,132,125,110,106,96,87,87,109,109,116,102

\\ Member  COGS Not Found In Database
California,Caffeine Free Cola, COGS,
95,104,109,123,127,141,154,154,122,122,113,127

\\ Member  Marketing Not Found In Database
California,Caffeine Free Cola, Marketing, 30,33,34,39,40,45,49,49,39,39,36,40

\\ Member  Payroll Not Found In Database
California,Caffeine Free Cola, Payroll, 22,22,22,23,23,23,22,22,22,22,22,22

\\ Member  Misc Not Found In Database
California,Caffeine Free Cola, Misc, 0,0,1,1,0,0,0,1,0,0,1,1
```

Figure 13-14, Error Log

3. Open the data sources or records that didn't load completely and fix them.

   The dimension build fails completely if Essbase cannot initialize the rules file. Check the following:

   - Can you validate the rules file? See "Step 4: Validating Dimension Build Rules" on page 13-14.
   - Is the data source available? See Chapter 23, Debugging and Optimizing Data Loads.
   - Is the server available? See Chapter 23, Debugging and Optimizing Data Loads.

   If the dimension build fails while processing the records, Essbase writes the errors to the error log. Check the outline to determine how many dimensions changed.

4. Reload the records that failed.

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 and/or member information to the outline.

Chapter 14 Sizing Your Database

The Essbase Storage Manager helps you store your data as efficiently as possible.

This chapter introduces you to the Essbase Storage Manager, tells you how to estimate disk and memory requirements, and tells you about your database settings and how to change them. It contains the following sections:

- "Units of Essbase Data Storage" on page 14-1
- "Determining Disk Space Requirements" on page 14-2
- "Determining Memory Requirements" on page 14-11

Note: If you are migrating from an earlier version of Essbase, see the *Start Here* booklet for guidance in estimating space requirements.

For information about specifying and changing Storage Manager settings, see Chapter 40, Specifying Storage Manager Settings.

Units of Essbase Data Storage

This section briefly explains the units of storage you need to know about to size your database. This section assumes you are familiar with basic concepts of the Essbase Storage Manager. See Chapter 39, Introducing the Essbase Storage Manager, to learn about the Storage Manager.

When you load a database, Essbase creates and populates index and data files on the disk. When you access data, Essbase executes an index search and loads index information and data into an index cache and a data cache, respectively.

Essbase uses the following storage units:

Storage Unit	Description
Index file	A file Essbase uses to store data retrieval information. Resides on disk.
Index page	A subdivision of an index file. Contains index entries that point to data blocks.
Index cache	A buffer in memory that holds index pages for the current operation.
Data block	A multidimensional array representing all cells of the dense dimensions for a particular intersection of sparse dimensions.
Data cache	A buffer in memory that holds data blocks for the current operation.
Calculator cache	A buffer in memory that creates and tracks data blocks during calculation.

Essbase loads index information from the disk into the index cache on demand.The index information points to data, which Essbase retrieves from data files into the data cache.

Note: Essbase uses another, optional cache in memory, called a *calculator (calc) cache*, for calculations. It is usually much smaller than the index and data caches. For information on the calc cache, see Chapter 32, Optimizing Your Calculations.

Determining Disk Space Requirements

This section provides both general background information and more detailed calculations for estimating disk space requirements.

General Disk Space Requirements

This section provides *general* guidelines for calculating adequate disk space. For a more detailed scenario, see “Estimating the Necessary Disk Space” on page 14-6.

To determine the amount of disk space you need for Essbase, consider Table 14-1.

Table 14-1, Base Requirements for Disk Storage

Software or Data	Minimum Disk Space Required
Base installation of Essbase	21–33 megabytes
Server software	5 megabytes
Sample applications	18.6 megabytes after fully loaded, with default settings (6 megabytes prior to loading)
Essbase databases	See discussion below

This section focuses on the database, as that is the most variable element in Table 14-1.

The total amount of disk space required for your data on the server depends on these factors:

- The number of existing blocks in the database. An *existing block* contains at least one non-missing value.
- The number of data values (cells) that populate the blocks.
- Whether data blocks are compressed, and with what compression scheme. See Chapter 39, Introducing the Essbase Storage Manager, for information about data compression.
- Your Isolation Level settings. If you choose *committed* access, or your Commit Blocks or Commit Rows setting is 0 under *uncommitted* access (the default), you need to allow space on your disk for double the size of the database. Under these conditions, Essbase protects against data corruption by maintaining redundant data. For more information, see Chapter 41, Ensuring Data Integrity.
- Whether you are using Hyperion Essbase Partitioning to store a replicated partition. See “Considerations When Using Partitioning” on page 14-9.
- Whether you are using stored Linked Reporting Objects. See “Considerations When Using Linked Reporting Objects” on page 14-10.
- Whether you have tagged any members as Dynamic Calc or Dynamic Calc And Store. See “Considerations When Using Dynamic Calculations” on page 14-10 for more information.

Most multidimensional applications tend to be sparse and the amount of space required difficult to determine precisely. The following guidelines, in conjunction with information presented in the **Database Information** dialog box, help provide a rough estimate of disk space requirements.

Note: This section and its calculations assume the following conditions, and thus provides a “worst-case” scenario:

- No data compression is enabled. By default, bitmap compression is enabled for your database. Your database probably uses much less space than this calculation allows for. The actual disk space consumed by a data file varies depending on the type of compression, if any, Essbase used for each block. For more information about data compression, see Chapter 39, Introducing the Essbase Storage Manager.
- You are not using dynamic calculations (that is, that you have not tagged any members as dynamic calculation members). Tagging members as Dynamic Calc can vastly reduce the size of your database. For information about dynamic calculation, see Chapter 28, Dynamically Calculating Data Values.

Determining Potential Data Block Size

The *actual* size of a data block depends on the amount of data that exists where dimensions intersect. To determine the *potential, expanded* (uncompressed) size of each data block, follow these steps.

1. Multiply the number of stored members in each dense dimension together.

   For example, the Sample Basic database contains:

   - 12 stored members from the Year dimension
   - Eight stored members from the Measures dimension
   - Two stored members from the Scenario dimension

   This represents 12 x 8 x 2 = 192 data cells that potentially populate each data block.

2. Multiply the figure you derived in Step 1 by the size of each cell (8 bytes).

   Using our example:

   ```
   192 x 8 = 1,536 bytes
   ```

Each block has a 72-byte header. Add 72 to the amount you calculated in Step 2.

```
1,536 + 72 = 1,608 bytes
```

Note: Label members and shared members do not use any storage space. Dynamic Calc members only use space if they are *stored*; such members are called Dynamic Calc And Store. For information on Dynamic Calc members, see Chapter 28, Dynamically Calculating Data Values.

Determining the Potential Number of Data Blocks

To determine the potential *number* of blocks, multiply the number of members in each sparse dimension together.

For example, the Sample Basic database contains:

- 19 stored members from the Product dimension
- 25 stored members from the Market dimension

19 x 25 = 475 potential data blocks.

Remember, label members, shared members, and Dynamic Calc members do not use storage space. (Essbase does store Dynamic Calc And Store members.)

Estimating Space Required for the Data Files

Here is one method for estimating the space required to store the data files (ESSxxxxx.PAG). For a more comprehensive estimate, see "Estimating the Necessary Disk Space" on page 14-6.

Multiply the potential block size by the potential number of blocks. For block size, see "Determining Potential Data Block Size" on page 14-4; for number of blocks, see "Determining the Potential Number of Data Blocks" on page 14-5.

For the Sample Basic database:

```
1,608 (potential block size) x 475 (number of potential data
blocks) = 763,800 bytes.
```

Essbase retains redundant data in some situations to ensure data integrity. To allow for redundant data storage, allocate enough disk space for double your database. In the example above, allow

```
763,800 x 2 = 1,527,600 bytes.
```

Estimating the Necessary Disk Space

The following formula provides safe, "worst-case" guidelines for allocating enough disk space:

```
         Fixed size overhead (header and bitmap)
+        Size of compressed data blocks
+        Fragmentation allowance
+        Estimated index size
=>       Estimated size of combined index and data files per
         database
```

Then,

```
         Estimated size of combined index and data files per
         database
x        2 (Data redundancy overhead)
=>       Estimated disk space needed
```

Note: Unlike the calculations in "General Disk Space Requirements" on page 14-2, the calculations used in this section assumes that:

- You are not using dynamic calculations. See "Considerations When Using Dynamic Calculations" on page 14-10 for more information.
- You are not including the sample applications (such as Sample Basic). This section is meant to help you size your own databases.
- Bitmap compression is enabled for your database by default. The actual disk space consumed by the data file varies depending on the type of compression, if any, Essbase used for each block. For more information about data compression, see "General Disk Space Requirements" on page 14-2.

Fixed-Size Overhead

To calculate the fixed-size overhead for a database, use this formula:

```
((Expanded block size in bytes / 64) + 72) x Number of blocks
```

Take the result value and round it up to the nearest multiple of eight, as this value must be expressed in bytes.

- `Expanded block size:` Although this section assumes you have compression enabled, the formula starts with expanded block size. An *expanded block* is a non-compressed block. To estimate expanded block size, see "Determining Potential Data Block Size" on page 14-4.

- `“64”:` The compression bitmap uses one bit for each cell in a block. Divide the expanded block size by 8 to get the number of cells; this equals the number of bits in a bitmap. Divide this value by 8 to get the number of bytes; hence we divide expanded block size by 64 to obtain the bitmap size, or fixed-size overhead.
- `“72”:` The block header size is 72.

For example:

Assume the expanded block size is 50,000 bytes and you have 400,000 blocks.

`50,000 / 64 = 781.25.` Round up to nearest multiple of eight: `784`. This is the bitmap size per block.

`784 + 72 = 856` bytes per block. This is the total amount of fixed overhead *per block*.

To calculate the fixed-size overhead for the entire database, multiply the overhead per block by the number of blocks:

```
856 x 400,000 = 342,400,000 bytes
```

Size of Compressed Data Blocks

This formula is very general. It assumes that bitmap compression is enabled.

To get a rough estimate of compressed block size, use this formula:

```
Expanded block size x Block density x Number of data blocks
```

For example, assuming expanded block size of 50,000 bytes, block density of 50%, and 400,000 blocks:

```
(50,000 x .5) x 400,000 = 10,000,000,000 bytes
```

To determine block density, you can load your database and choose Database | Information in Application Manager, click the **Statistics** page, and look at the **Block density** label. If you want to estimate block density prior to loading data, estimate the ratio of existing values to potential data values.

Note: Actual block density varies widely from block to block. The calculations in this section are for estimation purposes only.

Fragmentation Allowance

Whether you are using bitmap or RLE compression type, a certain amount of fragmentation occurs. The exact amount is based on individual database and operating system configurations and cannot be precisely forecast.

To cover worst-case scenarios, allocate an extra 20% of the expanded (uncompressed) block size to the estimated disk space.

Calculate 20% of `Block size` x `Number of blocks`. For example, using block size and number of blocks from our earlier examples:

```
(50,000 x 400,000) x .2 = 4,000,000,000
```

Estimating Index Size

The following formula helps you determine the total size of your index, including all index files. *This formula is for rough estimation purposes only*.

```
Number of blocks x 56
```

This formula is based on the size of an index leaf (Level 0) entry, which is a constant. To estimate the potential number of blocks, do one of the following:

- Use “Determining Potential Data Block Size” on page 14-4 to arrive at an estimate.
- Load your database and take the number of blocks from the **Potential number of blocks** label in the **Statistics** page of the **Database Information** dialog box in the Application Manager.

For example, assume you have 400,000 blocks:

```
400,000 x 56 = 22,400,000B
```

In this example, allow about 22 megabytes for the entire Essbase index.

Data Redundancy Overhead

During migration, and for seamless recovery purposes, Essbase temporarily retains duplicate blocks under some circumstances. To calculate the disk space overhead for data redundancy, use the following formula:

```
Index size + Combined size of data files
```

In other words, add together all your totals so far.

Putting It All Together

Using the results of the previous examples, the “worst-case” formula for determining the total disk space needed for the database described in this chapter is:

	Fixed size overhead (header and bitmap)	342,400,000
+	Size of compressed data blocks	10,000,000,000
+	Fragmentation allowance	4,000,000,000
=	Estimated size of combined data files per database	14,324,000,000
	Space needed for index	22,400,000
+	Estimated size of combined index data files per database	14,346,800,000
+	Data redundancy overhead	14,346,800,000
=	**Total estimated disk space needed**	**28,693,600,000**

So, in this example, you should allow about 28–29 gigabytes on disk for the data file, if you use bitmap compression (the default compression scheme).

Considerations When Using Partitioning

If you are using *replicated* partitions, you need to allocate enough disk space at the target database to hold double the total size of all replicated partitions. (You need to allocate double the size of all replicated partitions because, to aid in seamless recovery, Essbase temporarily retains duplicate blocks before committing transactions.)

When you use any type of partition, Essbase stores certain data (such as connection and time stamp information) at both the source and target databases. However, the amount of storage Essbase uses for these items is insignificant.

For information about Hyperion Essbase Partitioning, see Chapter 6, Designing Partitioned Applications.

Considerations When Using Linked Reporting Objects

The Linked Reporting Objects (LRO) feature lets you associate objects, such as flat files, with data cells.

If the linked object associated with a cell is a flat file, Essbase copies the file to the server. Therefore, you need to know the combined size of all flat files you are using as linked reporting objects.

You can set a limit on the size of a linked object, if it is a file (as opposed to a cell note). In the Application Manager, choose Application | Settings and enter the desired limit in the **Max. Attachment File Size** text box.

You can use the SETAPPSTATE command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

The Essbase Storage Manager stores information about linked reporting objects in a Linked Reporting Objects catalog. Linked Reporting Objects uses:

- 1K for every catalog entry. A catalog entry is stored as an index page.
- 600 bytes for every cell note. The cell note is stored in the catalog.

For more information, see Chapter 11, Linking Objects to Your Essbase Data.

Considerations When Using Dynamic Calculations

Dynamic Calc and Dynamic Calc And Store members are used to streamline calculations in some situations.

- With Dynamic Calc, Essbase stores fewer calculated data values than with regular calculations. You can conserve disk space when you specify certain members as Dynamic Calc.
- With Dynamic Calc And Store, Essbase stores the dynamically calculated data values.

For information on dynamic calculations, see Chapter 28, Dynamically Calculating Data Values.

Determining Memory Requirements

The base recommendation for running Essbase and its sample applications is 32 megabytes; start with this amount **per database**. Then, allocate additional memory if you need it after completing the calculations in this section.

Note: On UNIX systems, you may want to start with 64 megabytes of memory.

The following table displays the major components of the Essbase server, from a memory usage point of view.

Essbase Component	Approximate Amount of Memory Required
Code and static data	3–4 megabyte
Database outline	200 bytes per stored member
Index, data, and calculator caches	Minimum of 40 megabytes. See discussion below.

Start by allocating 3–4 megabytes for the code and static data; then allocate 200 bytes for each stored member in your database outline, for each database in your application. Then consider the cache allocations in the following discussion. The cache size settings can have a significant impact on database and general server performance. See Chapter 39, Introducing the Essbase Storage Manager, for general performance information regarding cache sizes.

Essbase uses buffers called *caches* to hold data and index information in memory while database operations are being performed. For more information on caches, see “Units of Essbase Data Storage” on page 14-1.

Cache Type	Default Setting
Index	1024 kilobytes (1 megabyte)
Data	3072 kilobytes (3 megabytes)
Calculator	200 kilobytes

The exact setting you should use for each cache depends on several factors:

- Data distribution
- Dense/sparse configuration
- Data load and calculation properties
- The characteristics of your applications and databases

If you are migrating your database from a previous version of Essbase, see the *Start Here* booklet.

Sizing Your Index and Data Caches

This section provides information and considerations for setting up your cache sizes, and recommends starting sizes. This section does not provide many details about where and how to change these settings; see Chapter 40, Specifying Storage Manager Settings for information.

Notes:

- If you are migrating from a previous release of Essbase, see the *Start Here* booklet for important migration information.
- As you use Essbase, you might need to fine-tune these settings. Keep in mind that a number of factors besides storage settings affect performance; for example, calculator settings and calc script commands. For information about calculations, see Part V, Calculating Your Data.

The index and data cache sizes are the most critical Essbase memory 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. When the index cache size equals or exceeds the index size (including all index files on all volumes), performance does not improve. Set your index cache size no larger than the index size. See “Estimating Index Size” on page 14-8 for an example of estimating index size.

The following illustration represents this guideline in a general way. Actual results would depend on the type of operation and on the dense/sparse configuration.

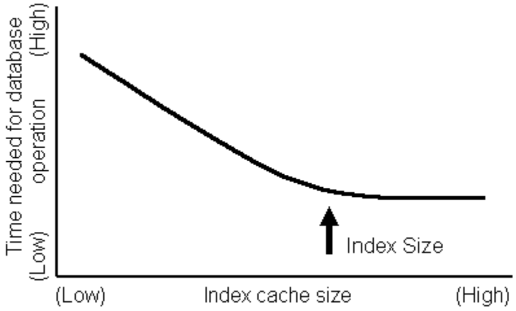


Figure 14-1, Effect of Index Cache Size on Performance

The data cache should be about two to three times larger than the index cache. The more concurrent users you expect on your database, the larger the data cache setting should be.

Setting Initial Cache Sizes

Essbase automatically uses default settings: 1 megabyte for the index cache, 3 megabytes for the data cache, 200 kilobytes for the calculator cache, 1 kilobyte for the index page. These settings are adequate for many installations, but because each site's needs vary widely, you need to follow the recommendations in this list to ensure optimal performance.

For definitions of index cache, index page, and data cache, see "Units of Essbase Data Storage" on page 14-1.

For more examples of setting cache sizes, see "Examples of Setting Cache Sizes" on page 14-15.

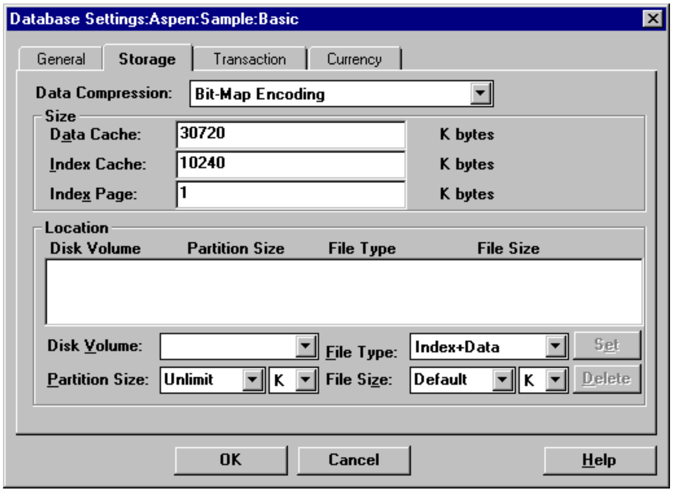
1. Before you start, take note of the available RAM on your server. As you do the following calculations, make sure you will not exceed the available RAM, or be prepared to add memory.
2. Specify your index cache size. Choose Database | Settings | Storage.

   Start with a value of at least 10240 kilobytes (10 megabytes). Allocate 10–20 megabytes to start with, but not do not set this value higher than the total index size, as no performance improvement would result.

To determine the total index size:

- Before loading data, see “Estimating Index Size” on page 14-8.
- When the database is already loaded, choose Database | Information | Files.

3. Specify your data cache size. Choose Database | Settings | Storage. Start with a value of at least 30720 kilobytes (30 megabytes). If you have many concurrent users, you might want to start with up to 60 megabytes.



Database Settings:Aspen:Sample:Basic

Data Compression: Bit-Map Encoding

Size

Data Cache: 30720 K bytes

Index Cache: 10240 K bytes

Index Page: 1 K bytes

Location

Disk Volume | Partition Size | File Type | File Size

Disk Volume:

File Type: Index+Data

Partition Size: Unlimit K

File Size: Default K

Figure 14-2, Setting Index and Data Cache Sizes

4. Use the default calculator cache size, 200 kilobytes. The **Database Settings** dialog box does not display this value. See Chapter 32, Optimizing Your Calculations for information how to set this value.

5. Now, repeat Steps 1–4 for every database in that application. For *n* databases, the total of:

```
(Index Cache size + Data Cache size + Calc Cache size) for Database_1
+ (Index Cache size + Data Cache size + Calc Cache size) for Database_2
.
.
.
+ (Index Cache size + Data Cache size + Calc Cache size) for Database_n
+ (RAM required to run Essbase before loading data)
```

must not exceed available RAM.

If your databases have very similar cache sizes, you can use this formula:

```
(Index Cache size + Data Cache size + Calc Cache size) x n
+ (RAM required to run Essbase before loading data)
```

6. Start the database (or, if it is already running, stop and re-start it) so the settings take effect. See Chapter 44, Running Essbase, Applications, and Databases, for information about stopping and starting databases.

7. After using the database for a few days under your installation's usual conditions, check and see how Essbase is performing.

   You can check Essbase performance in several ways. Chapter 45, Monitoring Performance Using Diagnostics, provides more information, but here is an example:

   Check the **Run-time** page of the **Database Information** dialog box (choose Database | Information from the Application Manager menu). In particular, the **Hit ratio on Index Cache** indicates the success rate of Essbase in locating index information in the index cache without having to retrieve another Index Page from disk. This value should be a high number (100 is the highest possible value). If it is not at least 80%, you may want to increase the index cache size.

Examples of Setting Cache Sizes

This section provides examples of setting index and data cache sizes.

Example 1: One Database

This is a basic example with one database in the application.

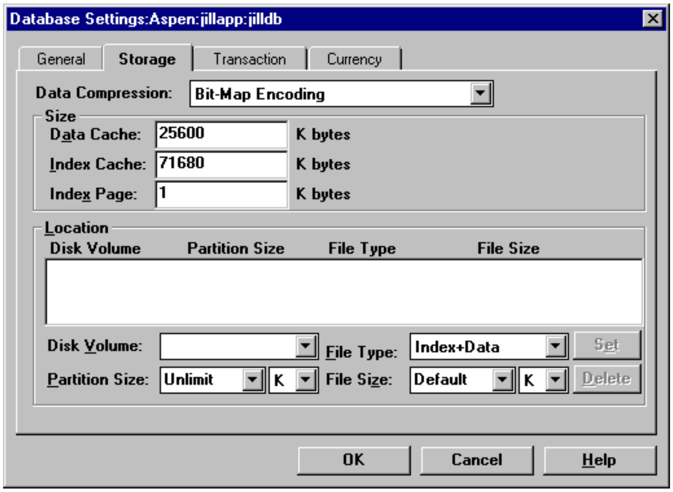
In this example, assume the following statistics:

- Available RAM—512 megabytes
- Memory needed to run Essbase—approximately 32 megabytes
- Number of databases in this application—1
- Index size—70 megabytes

1. Start the application and the database.
2. Choose Database | Settings | Storage in the Application Manager. Essbase displays the default values of 3072 kilobytes for the data cache and 1024 kilobytes for the index cache.
3. Type over the existing index cache value, replacing it with the equivalent of 70 megabytes in kilobytes: `71680`.

4. In this example, set the data cache size equal to 25 megabytes in kilobytes. Replace the existing data cache value with `25600`.

   Our **Database Settings** dialog box now looks like this:



Database Settings:Aspen:jillapp:jilldb

Data Compression: Bit-Map Encoding

Size

Data Cache: 25600 K bytes

Index Cache: 71680 K bytes

Index Page: 1 K bytes

Location

Disk Volume:

File Type: Index+Data

Partition Size: Unlimit K

File Size: Default K

Figure 14-3, Example of Setting Cache Sizes

5. Stop and restart the database to make the changed settings effective. See Chapter 45, Monitoring Performance Using Diagnostics, for information on stopping and starting databases.

Example 2: Multiple Databases

This example features two databases of different sizes.

In this example, assume the following statistics:

- Available RAM—800 megabytes
- Memory needed to run Essbase—approximately 32 megabytes
- Number of databases in this application—2
- Index size for database DB_1—150 megabytes
- Index size for database DB_2—60 megabytes

Ideally, you would create a 150-megabyte index cache for DB_1, to match the size if DB_1's index, and a 60-megabyte index cache for DB_2. However, you have two databases to fit into available RAM, so you need to be conservative in allocating memory. Assume that DB_1 is more heavily used, often by concurrent users, than DB_2. Allocate adequate memory to DB_2, and give DB_1 the balance of available memory as follows.

1. Start the application and database DB_1. Choose Database | Settings in the Application Manager. Essbase displays the default values of 3072 kilobytes for the data cache and 1024 kilobytes for the index cache.
2. Replace the existing index cache value with `125000`.
3. The data cache should be about three times the size of the index cache, possibly larger because it supports many users.

   Replace the existing data cache value with `400000.`
4. Let's look at the amount of RAM you have left to work with.

   ```
   125 + 400 = 525 megabytes
   ```

   ```
     800  Total RAM available on the operating system
   - 525  Memory allocated for DB_1
   -  32  Approximate memory needed to run Essbase
     243  Balance of available RAM in megabytes
   ```
5. Go to the Application Desktop window and select DB_2 from the **Databases** list box.

With 243 megabytes of RAM left, you need to be conservative allocating memory for DB_2. Ideally, you would use about 60 megabytes for the index cache and around three times that, 180 megabytes, for the data cache. Assuming DB_2 rarely has concurrent users, you can use a smaller data cache size.

1. Choose Database | Settings | Storage.
2. Enter `50000` for the index cache size for DB_2 (50 megabytes).
3. Enter `100000` for the data cache size for DB_2 (100 megabytes).
4. Calculate the RAM allocated for DB_2.

   ```
   50 + 100 + 32 (base requirement) = 182 megabytes
   ```

   You do not want to use all of the available RAM, so you are finished, for now.
5. Test your databases to make sure the cache settings are adequate. Of course, the best option is to add more RAM if you need to.
6. Stop and restart the databases to make the changed settings effective. See Chapter 44, Running Essbase, Applications, and Databases, for information on stopping and starting databases.

Chapter 15 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.

Warning: You must design partitions carefully. We strongly recommend that you read Chapter 6, Designing Partitioned Applications before creating partitions.

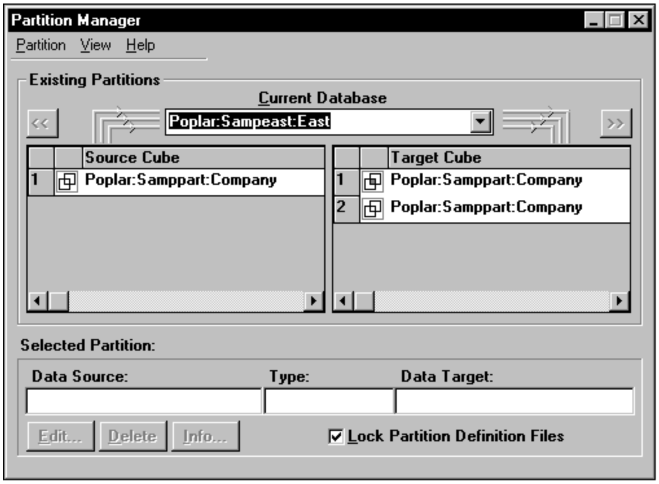
This chapter contains the following sections on creating partitions:

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:

Opening the Partition Manager

To open the Partition Manager:

1. Select the database to partition; for example, East.
2. Choose Database | Partition Manager. Click Help for information about items in the 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

Figure 15-1, 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.

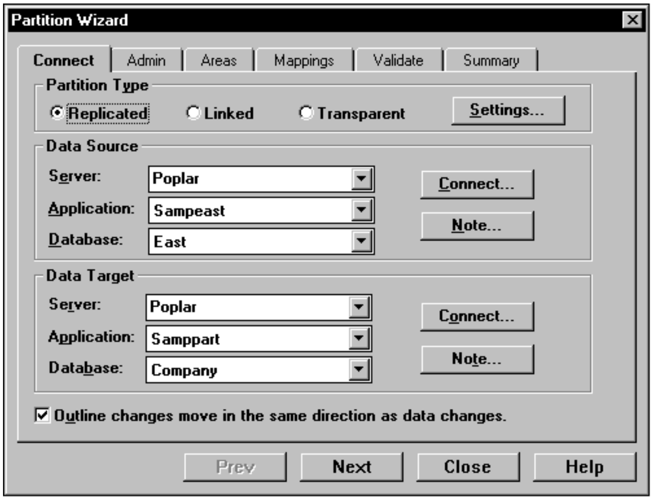
Creating a New Partition

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.

The **Connect** page of the Partition Wizard shown in Figure 15-2 is where you:

- 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.



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.

Figure 15-2, Connect Page of the Partition Wizard

Note: For more information on using Dynamic Time Series members in partitions, see "Calculating Time Series Data in Application Partitions" on page 29-14.

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 6-8.

The examples used in this section are based on the example described in “Scenario 1: Partitioning an Existing Database” on page 6-25. This example replicates the Eastern Region’s Actual member in the Sampeast database (the data source) to the Samppart Company database (the data target).

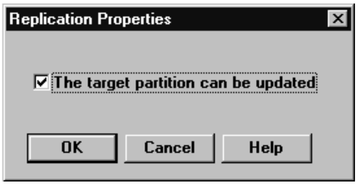
Creating a replicated partition involves the following steps:

- “Specifying the Partition Type and Connection Information” on page 15-4
- “Setting the User Name and Password” on page 15-6
- “Defining the Areas in a Partition” on page 15-7
- “Mapping Data Source Members to Data Target Members” on page 15-10
- “Validating the Partition” on page 15-29
- “Saving the Partition Definition” on page 15-31

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 15-3.
2. Choose **Replicated** from the **Connection** page of the Partition Wizard.
3. Click the **Settings** button to open the **Replication Properties** dialog box.


Replication Properties

☑ The target partition can be updated

Figure 15-3, Replication Properties Dialog Box

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 Hyperion Essbase 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.

   Click OK.

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 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.

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.

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 15-38.

Setting the User Name and Password

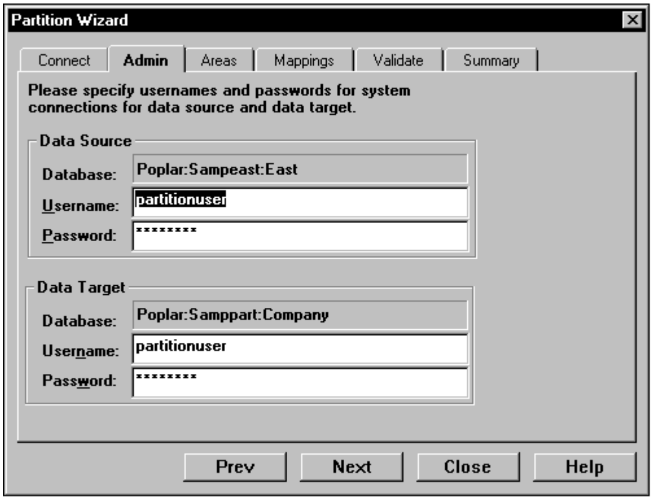
You must specify a user name and password for Essbase to use when communicating between the data source and the data target. Essbase uses this user name and password to:

- 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 6-22.

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.



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: ********

Figure 15-4, Admin Page of the Partition Wizard

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.

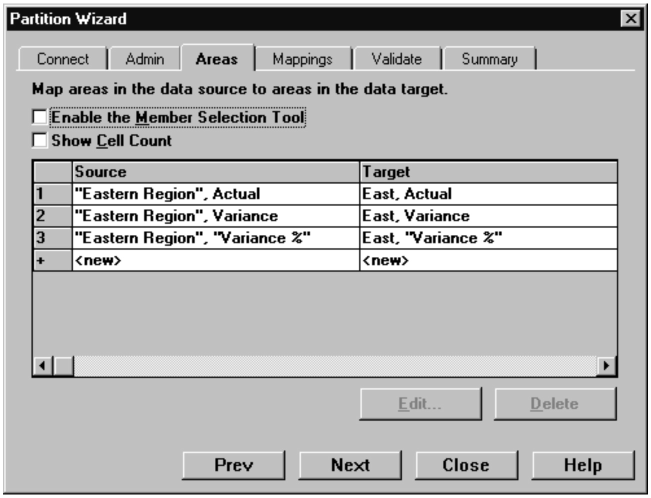
Defining the Areas in a Partition

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.

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.



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>

Figure 15-5, Areas Page of Partition Wizard

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 6-7.

   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 (Figure 15-5) 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. For more information on Essbase functions, see the online *Technical Reference* in your `DOCS` directory.

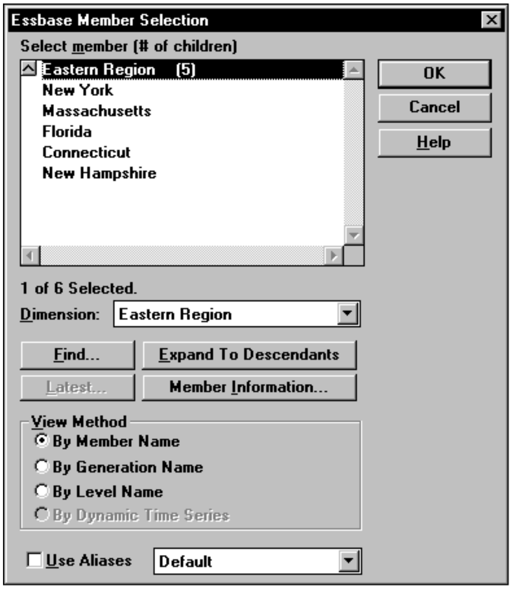
4. Repeat steps 2 and 3 for the **Target** box. For our example, enter:

   ```
   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 15-5) to open the **Member Selection** dialog box. For more information about the parts of the **Member Selection** dialog box, click **Help**.



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

Figure 15-6, Member Selection Dialog Box

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 15-6. 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 steps 2 through 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 15-8.

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 15-29.

If the data source and data target outlines contain different members or the members have different names in each outline, you need to map the data source members to the data target members. For example, if your data source contains the following dimensions and members:

```
Product
       Cola

Year
       1996
Market
       East
```

And your data target contains:

```
Product
       Cola

Year
       1996
Market
       Eastern Region
```

You will need to map data source and data target members.

Because you know that East in the data source corresponds to Eastern Region in the data target, map East to Eastern Region. Now Essbase loads all data corresponding to East in the data source into Eastern Region in the data target. So, for example, if the data value for Cola, 1996, East is 15 in the data source, the data value for Cola, 1996, Eastern Region will be 15 in the data target.

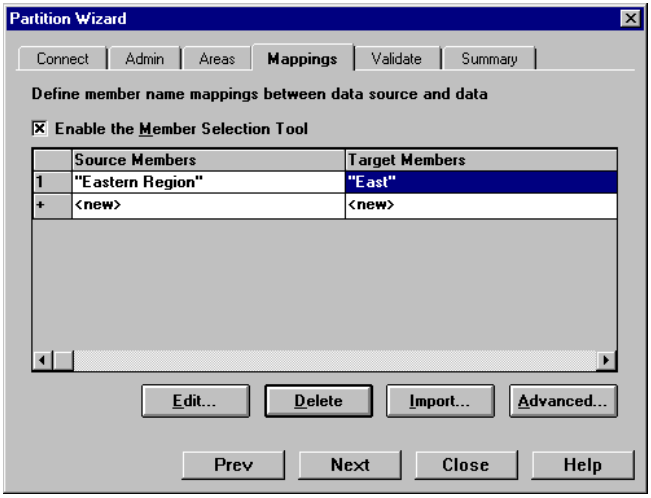
The following table lists more examples:

If the data source has...	Such as...	You should...
Different member names than the data target	A member is named Eastern Region in the data source and East in the data target.	Map Eastern Region to East.
An extra dimension member	A dimension exists in the data source and not the data target. For example, suppose that the data source has Market and Year dimensions. The Year dimension has 1996 and 1997 members. Now, suppose that the data target tracks information for 1997 only. It has only Market dimensions (since all data corresponds to Market, 1997).	Map the 1997 member in the data source to `void` in the data target.
A missing dimension member	A dimension does not exist in the data source but exists in the data target. For example, suppose that the data source has Market and 1997 dimensions. Now, suppose that the data target has Market, 1997, and Budget dimensions.	Map `void` in the data source to Budget in the data target.

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:

1. From the **Areas** page of the Partition Wizard, click **Next** or **Mappings** to move to the **Mappings** page.



Define member name mappings between data source and data

☒ Enable the Member Selection Tool

	Source Members	Target Members
1	"Eastern Region"	"East"
+	<new>	<new>

Figure 15-7, Mappings Page of the Partition Wizard

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 15-12.
   - Select the member name in the **Member Select** dialog box. See “Entering the Mapping Using the Member Selection Dialog Box” on page 15-13.
   - Import the member mappings from an external data file. See “Importing the Member Mapping” on page 15-14.

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 15-7) 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"`

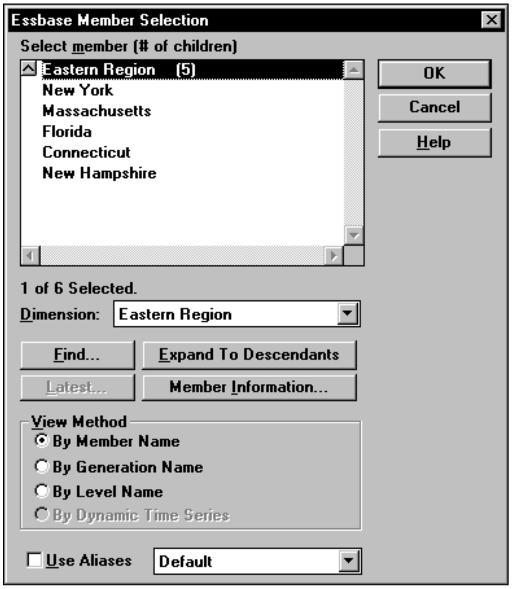
4. Double-click the **New** field in the **Target Members** box to open the **Member Name** dialog box.
5. Enter the member name to map it to in the data target and click OK. For our example, enter:

   `East`

Entering the Mapping Using the Member Selection Dialog Box

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 15-7) to open the **Member Selection** dialog box. For more information about the parts of the **Member Selection** dialog box, click Help.



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

Figure 15-8, Member Selection Dialog Box

3. Choose the dimension or member to map. For our example, choose the Eastern Region dimension.

4. Click OK.

5. Repeat steps 2 through 4 for the **Target Members** box. For our example, choose East in the Market dimension.

Note: For the linked partition described in “Scenario 3: Linking Two Databases” on page 6-29, you must also map the Package dimension to `void`.

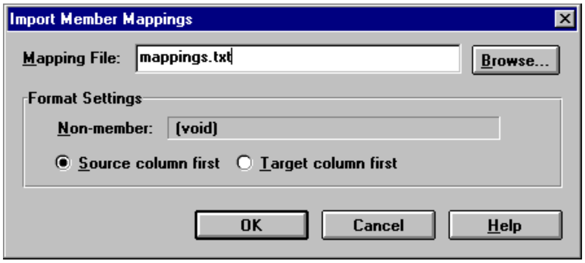
6. When you finish, click OK.

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 15-8.

Importing the Member Mapping

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.



Import Member Mappings

Mapping File: mappings.txt

Format Settings

Non-member: (void)

● Source column first ○ Target column first

Figure 15-9, Import Member Mappings Dialog Box

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):

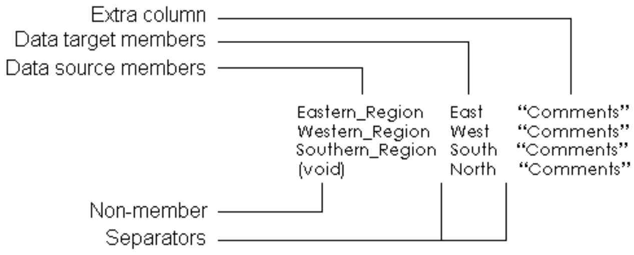


Figure 15-10, Member Mapping Import File

- *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 Essbase server. For more information, see “Transparent Partitions” on page 6-13.

The examples used in this section are based on the example described in “Scenario 1: Partitioning an Existing Database” on page 6-25. 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 15-16
- “Setting the User Name and Password” on page 15-17
- “Defining the Areas in a Transparent Partition” on page 15-17
- “Mapping Data Source Members to Data Target Members” on page 15-20
- “Validating the Partition” on page 15-29
- “Saving the Partition Definition” on page 15-31

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 15-3.
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 15-38.

Setting the User Name and Password

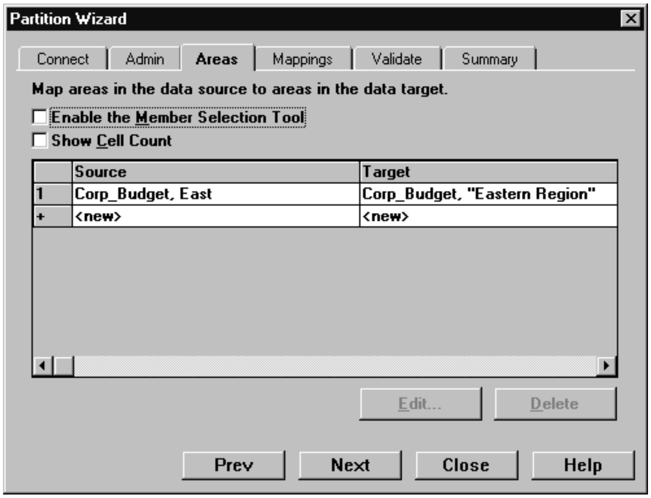
The procedure for this is identical to creating the user name and password for a replicated partition. See "Setting the User Name and Password" on page 15-6.

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 15-7.

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.



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>

Figure 15-11, Areas Page of Partition Wizard

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 6-7.

   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 user-defined attributes. For more information on Essbase functions, see the online *Technical Reference* in your `DOCS` directory.

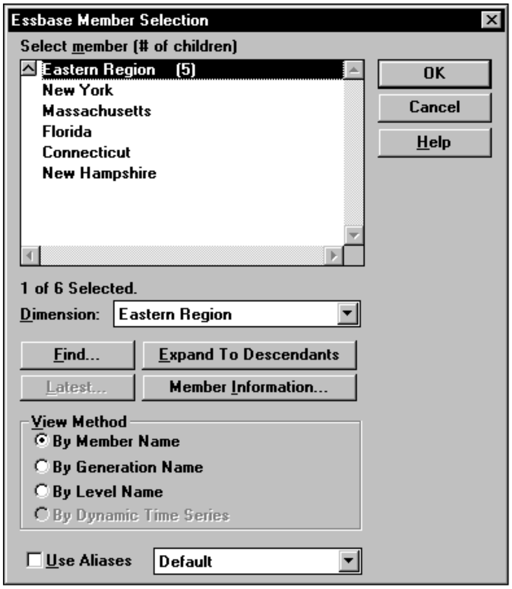
4. Repeat steps 2 and 3 for the **Target** box. For our example, enter:

```
Corp_Budget, "Eastern Region"
```

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 to open the **Member Selection** dialog box. For more information about the parts of the **Member Selection** dialog box, click **Help**.



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

Figure 15-12, Member Selection Dialog Box

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 steps 2 through 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 15-8.

Mapping Data Source Members to Data Target Members

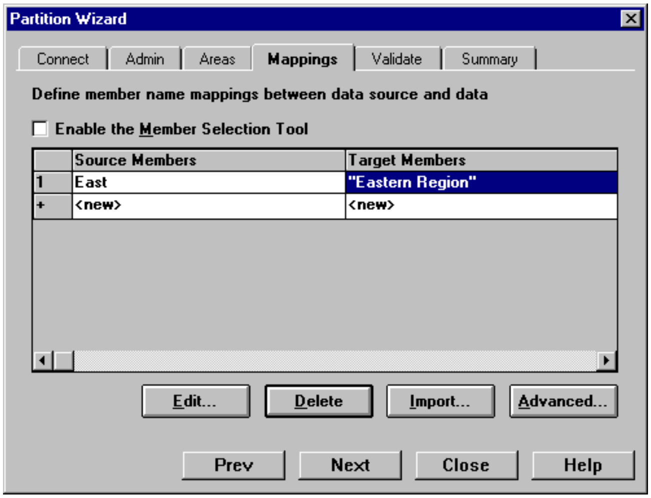
Essbase must be able to map all shared data source members to data target members to create a partition. If both the data source and the data target contain the same number of members and use the same member names, Essbase automatically maps the members. Skip to "Validating the Partition" on page 15-29.

For a brief introduction to member mapping, see "Mapping Data Source Members to Data Target Members" on page 15-10.

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.



Define member name mappings between data source and data

□ Enable the Member Selection Tool

	Source Members	Target Members
1	East	"Eastern Region"
+	<new>	<new>

Figure 15-13, Mappings Page of the Partition Wizard

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 15-21.
   - Select the member name in the **Member Select** dialog box. See "Entering the Mapping Using the Member Selection Dialog Box" on page 15-21.

- Import the member mappings from an external data file. See “Importing the Member Mapping” on page 15-14.

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 15-13) 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"
   ```

Entering the Mapping Using the Member Selection Dialog Box

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 15-13) 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 steps 2 through 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 15-8.

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 6-18.

The examples used in this section are based on the example described in "Scenario 3: Linking Two Databases" on page 6-29. This example links the Product dimension in the Sample Basic database to the TBC Demo database. This scenario is *not* shipped with Essbase.

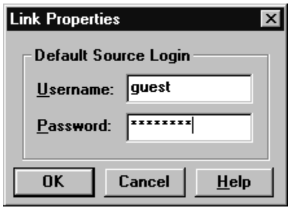
Creating a linked partition involves the following steps:

- "Specifying the Partition Type and Connection Information" on page 15-22
- "Setting the User Name and Password" on page 15-23
- "Defining the Areas in a Partition" on page 15-24
- "Mapping Data Source Members to Data Target Members" on page 15-26
- "Validating the Partition" on page 15-29
- "Saving the Partition Definition" on page 15-31

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 15-3.
2. Choose **Linked** from the **Connection** page of the Partition Wizard.
3. Click the **Settings** button to open the **Link Properties** dialog box.



Link Properties

Default Source Login

Username: guest

Password: ********

Figure 15-14, Link Properties Dialog Box

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 6-24.

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 appear 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 use as the source outline, see "Introducing Outline Synchronization" on page 15-38.

Setting the User Name and Password

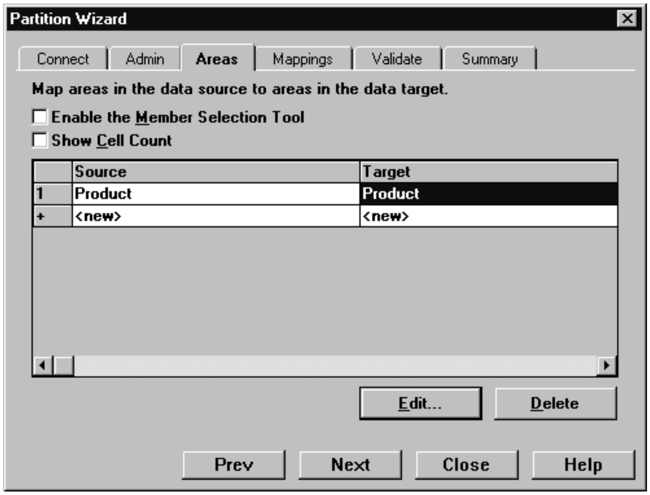
The procedure for this is identical to creating the user name and password for a replicated partition. See "Setting the User Name and Password" on page 15-6.

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 15-7.

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.



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>

Figure 15-15, Areas Page of Partition Wizard

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 6-7.

   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 Wizard, 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 user-defined attributes. For more information on Essbase functions, see the online *Technical Reference* in your `DOCS` directory.

4. Repeat steps 2 and 3 for the **Target** box. For our example, enter:

   ```
   Product
   ```

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** 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** 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 steps 2 through 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 15-8.

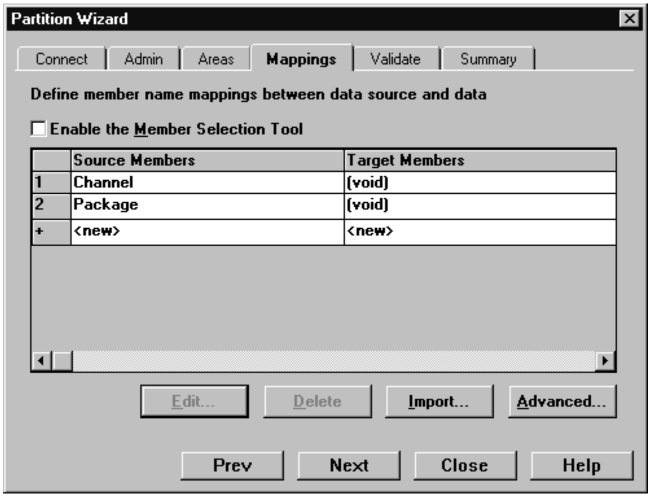
Mapping Data Source Members to Data Target Members

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 15-29.

For a brief introduction to member mapping, see "Mapping Data Source Members to Data Target Members" on page 15-10.

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.



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>

Figure 15-16, Mappings Page of the Partition Wizard

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 15-27.
   - Select the member name in the **Member Select** dialog box. See "Entering the Mapping Using the Member Selection Dialog Box" on page 15-28.
   - Import the member mappings from an external data file. See "Importing the Member Mapping" on page 15-14.

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 15-16) 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 6-29, you must also map the Package dimension to `void`.

Entering the Mapping Using the Member Selection Dialog Box

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 15-16) 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 steps 2 through 4 for the **Target Members** box. For our example, choose `void`. For the linked partition described in “Scenario 3: Linking Two Databases” on page 6-29, 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 15-8.

Validating the Partition

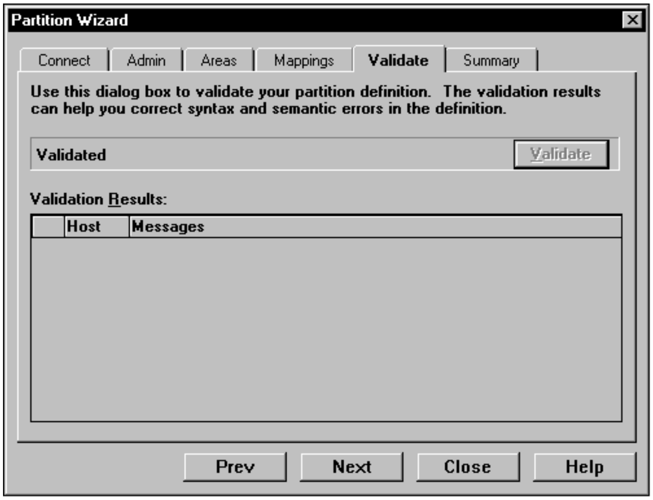
When you create a partition, validate it to ensure that it's correct before you use it.

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 15-31.

To validate a partition:

1. From the **Mappings** page in the Partition Wizard, click **Next** or **Validate** to move to the **Validate** page.



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

Figure 15-17, Validate Page of the Partition Wizard

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.

Use the VALIDATEPARTITIONDEFFILE command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, 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:

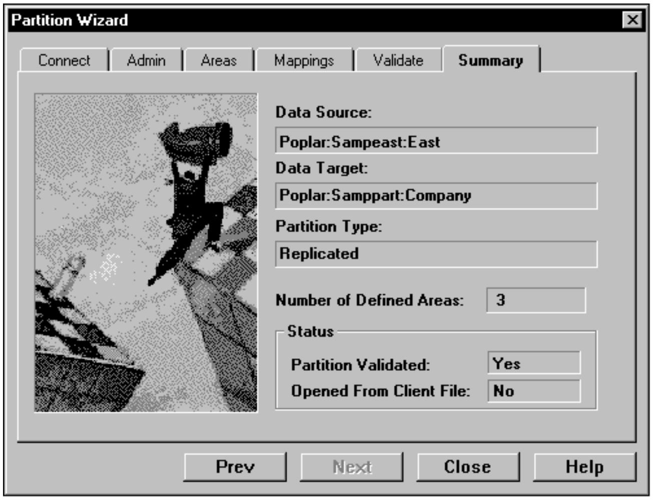
- The area definition is valid (contains no syntax errors).
- The specified data source members are valid and map to valid members in the data target.
- All connection information is correct; that is, the server names, database names, application names, user names, and password information.
- For linked 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.

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.



Data Source: Poplar:Sampeast:East

Data Target: Poplar:Samppart:Company

Partition Type: Replicated

Number of Defined Areas: 3

Status

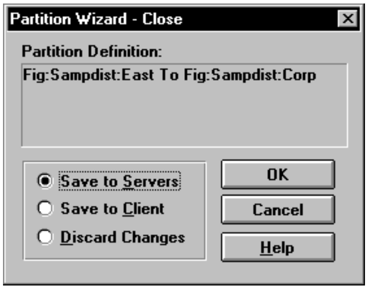
Partition Validated: Yes

Opened From Client File: No

Figure 15-18, Summary Page of the Partition Wizard

Check the information in the **Summary** page to make sure that the partition is defined correctly. If it's not, click the appropriate page in the Partition Wizard to change the settings.

2. Click **Close** to open the **Close** dialog box.



Partition Definition: Fig:Sampdist:East To Fig:Sampdist:Corp

● Save to Servers

○ Save to Client

○ Discard Changes

Figure 15-19, Partition Wizard Close Dialog Box

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 15-47.

Testing Partitions

To test a partition:

- View the data target(s) 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 don't need to perform advanced area-specific mapping. Skip to "Validating the Partition" on page 15-29.

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's 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

For example, if your data source contains:

```
Product
       Cola

Market
       East
Year
       1998
       1999
```

And your data target contains:

```
Product
       Cola

Market
       East
Year
       1998
       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.

Since the data source does not have Actual and Budget members, you must also map these members to `void` in the data target.

Area-Specific Mapping Example

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 don't know how to use the **Areas** tab of the Partition Wizard, see "Defining the Areas in a Partition" on page 15-7.

   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 (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 15-10.

   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 your data source and data target are structured very differently but contain the same kind of information. For example, if your data source contains:

```
Market
       NY
       CA
Scenario
       Actual
       Budget
```

And your data target contains:

```
Customer_Planning
       NY_Actual
       NY_Budget
       CA_Actual
       CA_Budget
```

You know that 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 15-10.

Area-Specific Mapping Example

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 15-7.

   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.

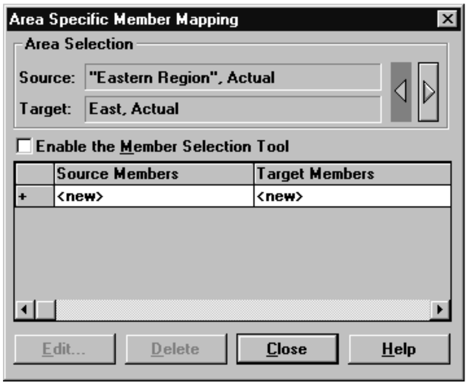
   i. Map Budget in the data source to CA_Budget in the data target.

   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.



Area Specific Member Mapping

Area Selection

Source: "Eastern Region", Actual

Target: East, Actual

☐ Enable the Member Selection Tool

	Source Members	Target Members
+	<new>	<new>

Figure 15-20, Area Specific Member Mapping Dialog Box

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.

What's the solution? Resynchronize your outlines. 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 attributes) 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 15-4.

If you make changes to the:

- 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 8, Creating and Changing Database Outlines.

Note: 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 file 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:

When You...	Essbase...
Make changes to the source outline	1. Records the changes in a change log file named `ESSxxxxx.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 file 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 file and places them in the target outline’s change log file. The change log files may have different names on the source outline and the target outline.

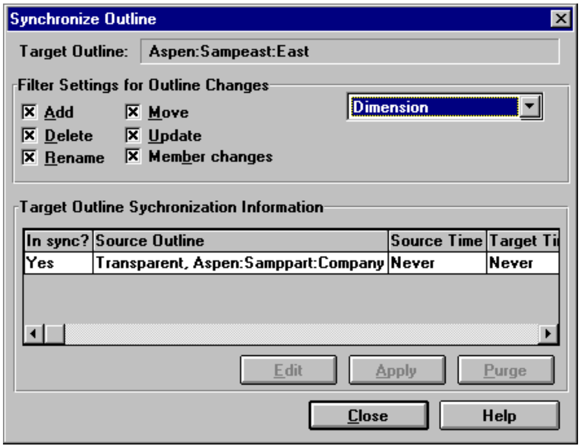
When You...	Essbase...
Select the changes to apply to the target outline	1. Applies the changes to the target outline. 2. Updates the timestamp in the target outline's `.DDB` file, using the time from the source outline.

Warning: 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.



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

Figure 15-21, Synchronize Outline Dialog Box

3. Choose the types of changes to retrieve from the source outline. These changes are not applied until you click **Apply**.

   a. Choose **Dimension** from the list box. Select the types of changes to retrieve by checking their boxes.

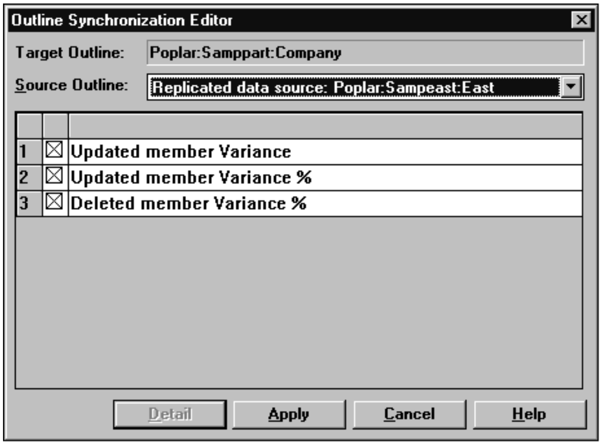
   b. Choose **Member** from the list box. Select the types of changes to retrieve by checking their boxes.

   c. Choose **Member Attributes** from the list box. Select the types of changes to retrieve by checking their boxes.

Warning: Deselecting 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`.

To accept or reject all changes, proceed to step 5. To choose which changes to accept or reject, proceed to step 4.

4. To choose which changes to accept or reject, click **Edit** to open the **Outline 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 %

Figure 15-22, Synchronization Editor Dialog Box

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 don't 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.

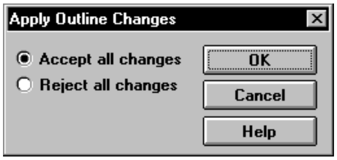
Warning: Rejecting specific changes in the Outline Synchronization Editor may result in losing additional changes. For example, if the database administrator creates a parent and children on the source outline and you clear the parent, the children may not get added to the target outline. If the children are not added to the target outline, a warning message prompts you to see the Application Server log file 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 7.

5. Click **Apply** to open the **Apply Outline Changes** dialog box.



● Accept all changes
○ Reject all changes

Figure 15-23, Apply Outline Changes Dialog Box

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 log files on both the target outline and the source outline, click **Purge**.

   Essbase deletes all records from the change log file that have been applied or rejected. If all records have been applied or rejected, Essbase deletes the change log file as well. Essbase does *not* purge records that have not yet been applied to the target outline.

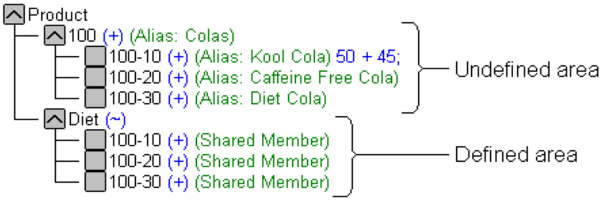
8. Click **Close** to close the **Synchronize Outline** dialog box.

Notes:

- Essbase keeps all change log files until you purge them. You should purge change log files 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 15-29.

How Shared Members are Treated 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 following example, 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.



```
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)
```

Figure 15-24, Shared Members and Outline Synchronization

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 15-24, 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 following example, 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 are 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.

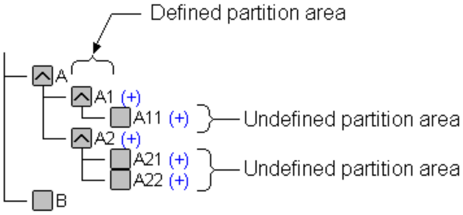


Figure 15-25, Implied Shared Members and Outline Synchronization

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 your outlines using ESSCMD. See the online *Technical Reference* in your DOCS directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

To Perform This Task:	Use This ESSCMD:
Retrieve the outline change log from the source outline. You can choose which 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 attribute changes. If you say Yes, Essbase retrieves all changes and proceeds to the next step. If you say No, Essbase asks you which member attribute changes to retrieve. You can retrieve changed member: statuses, aliases, unary calc symbols, account types, currency conversion information, user defined attributes, calc formulas, level numbers, and generation numbers.	`GETPARTITIONOTLCHANGES`
Apply the changes in the change log file to the target outline.	`APPLYOTLCHANGEFILE`
Reset the timestamp on both the source and the target outline change log files.	`RESETOTLCHANGETIME`
Purge entries that have been applied to the target outline from the change log files.	`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 below displays the replicated partition that you created between the Sampeast East and Samppart Company databases earlier in this chapter.

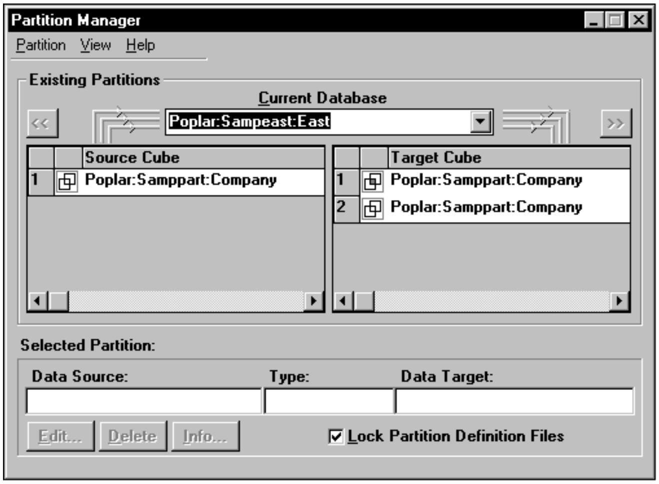


Figure 15-26, Partition Manager

3. Select the partition to edit and select the operation to perform on that partition:

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 “Building and Maintaining Partitions” on page 15-1.
Delete the partition	**Delete.** Essbase deletes the partition definition from the partition’s .DDB file on the data source and data target servers.

To...	Click...
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 online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, 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 `CLEARDATA` command in a calc script; using **Clear combinations** in your rules file during a data load; issuing `CLEAR UPPER`, `CLEAR INPUT` or `RESETDB` commands in the Hyperion Essbase 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.

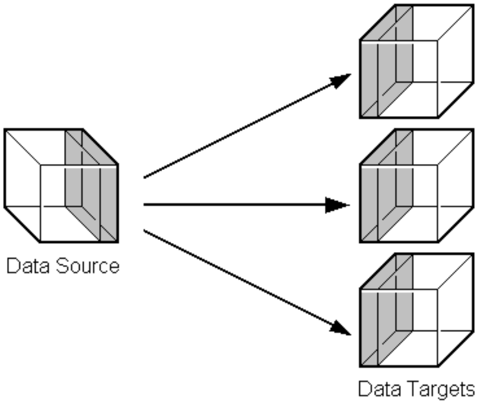


Figure 15-27, Put to Targets

- From 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.

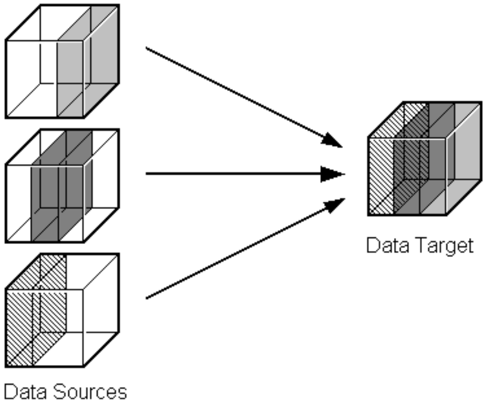


Figure 15-28, Get from Sources

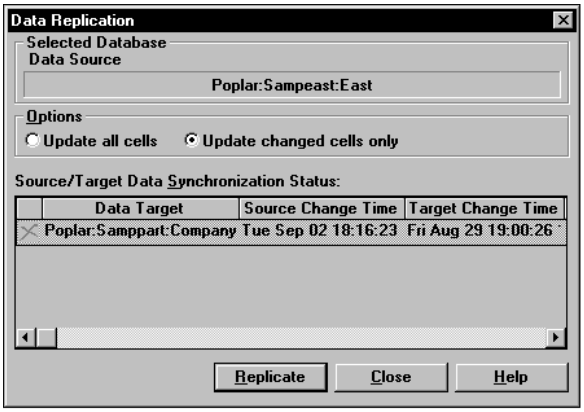
Updating Replicated Partitions

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.

   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

   Figure 15-29, Data Replication Dialog Box

   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's fastest to update just the changed cells. If you're not sure, see "Introducing the Updating of Replicated Partitions" on page 15-47.

3. Click **Replicate**. The replication may take some time to complete.

Note: 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 ESSCMD

You can update replicated partitions using ESSCMD. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

To Perform This Task:	Use This ESSCMD:
Retrieve only the cells that have changed from the data source and update them on the selected data target.	`GETUPDATEDREPLCELLS`
Retrieve all replicated cells from the data source and update them on the selected data target.	`GETALLREPLCELLS`
Retrieve only the cells that have changed from the selected data source and update them on the data target.	`PUTUPDATEDREPLCELLS`
Retrieve all replicated cells from the selected data source and update them on 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.
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.

Part III Designing and Building a Security System

Part III describes how to control access to data in Hyperion Essbase OLAP Server applications by designing and building a security system. Part III contains the following chapters:

- Chapter 16, Managing Security at Global and User Levels, 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 17, Controlling Access to Database Cells, describes how to define security filters and assign them to users.
- Chapter 18, Security Examples, contains detailed examples that illustrate how to implement a security system for Hyperion Essbase OLAP Server applications.

Part III-2

Chapter 16 Managing Security at Global and User Levels

Hyperion Essbase provides a comprehensive, multi-layered 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 security system. The next three sections explain how to create a security plan with the Essbase Security System.

This chapter contains the following sections:

Multi-Layered Security and Privileges

The Essbase Security System addresses a wide variety of database security needs with a multi-layered approach to let you develop the best plan for your environment. You can choose from the following security layers:

- **User and Group layer**, which lets you define privileges and access levels for individual users and groups of users. These settings take precedence over global access settings (general settings applied to applications and databases).
- **Global Access layer**, which lets you define privileges and access levels at the level of the server, the application, or the database. By default, all users have access (or no access) according to these global settings, unless they are granted higher privileges at the user and group layer.
- **Database Filter layer**, which lets you define specific database access levels that individual users can have for particular database members, down to the individual data value (cell). To learn more about the Database Filter layer, see Chapter 17, Controlling Access to Database Cells.

Privileges Available at Global and User Levels

Various types of management privileges can be assigned to the global or user levels.

- Ordinary users have no management privileges.
- At the global level, you assign management privileges on an application or database basis.
- At the user level, you assign management privileges to a specific user or group, and their privileges are constant for all applications and databases. Privileges assigned at the user level override the global settings defined at the application or database level.

The following table lists several tasks users with various management privileges can perform.

Table 16-1, Global and User Management Privileges

	Supervisor Privilege (assigned at user level)	Create/ Delete Users Privilege (assigned at user level)	Create/ Delete Applications Privilege (assigned at user level)	Application Designer Privilege (assigned at global level)	Database Designer Privilege (assigned at global level)
Create Applications	✓		✓		
Modify Applications	✓		✓[1]	✓[2]	
Delete Applications	✓		✓[1]		
Create Databases	✓		✓[1]	✓[2]	
Modify Databases	✓		✓[1]	✓[2]	✓[3]
Delete Databases	✓		✓[1]	✓[2]	
Define Global User Access at the Application Level	✓		✓[1]	✓[2]	
Define Global User Access at the Database Level	✓		✓[1]	✓[2]	✓[3]
Create Users	✓	✓			
Delete Users	✓	✓			
Log Out Users	✓		✓[1, 5]	✓[5]	
Reset User Passwords	✓	✓[4]			
Define Filter Objects	✓		✓[1]	✓[2]	✓[3]
Assign Filter Objects to Users or Groups	✓		✓[1]	✓[2]	✓[3]
Remove Data Locks	✓		✓[1]	✓[2]	✓[3]

[1] Users with Create/Delete Applications privilege can affect only applications they have created.

[2] Application Designer privileges are assigned per application; a user with Application Designer privilege in one application doesn't necessarily have that privilege in another.

[3] Database Designer privileges are assigned per database; a user with Database Designer privilege in one database doesn't necessarily have that privilege in another.

[4] Users with Create/Delete Users, Groups privilege can reset passwords of other users only if the other users have equal or lower privileges.

[5] You can log out only those users who are connected to an application for which you have Application Designer privilege. If you have Create/Delete Applications privilege, you automatically have Application Designer privilege for any application you create.

Managing Security at the User and Group Layer

The User and Group Access layer lets you define security settings for individual users and groups of users. Groups are collections of users that share the same minimum privileges. Users inherit all privileges of the group and can have access to privileges that exceed those of other group members. Manage user and group administration on a server–by–server basis.

User and Group Types

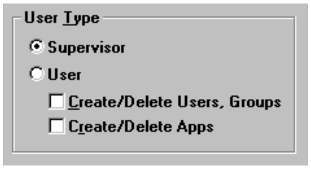
One major way to assign privileges to users and groups is to define user and group classes or "types" when you create or *edit* (modify the privileges of) the users and groups. You define these types in the **New** or **Edit User** dialog boxes (see Figure 16-6 and Figure 16-8, respectively).

In the Application Manager, users and groups can have one of four classes of privileges. A description of these user types follows. To learn how to define a type, see "Creating, Editing, and Copying Users and Groups" on page 16-8.

A user with Supervisor privilege has:

- Full access to all applications, databases, related files, and security mechanisms for a server.
- Privileges to create, delete, edit, and rename all users, including other supervisors.
- Privileges to create and delete applications.

Note: 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 privilege. Therefore, you cannot delete or downgrade the access level of the last supervisor on the server.



User Type

● Supervisor

○ User

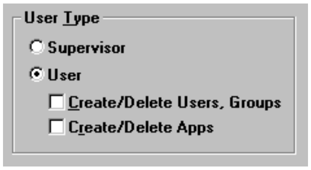
□ Create/Delete Users, Groups

□ Create/Delete Apps

Figure 16-1, Supervisor Privilege

A user with ordinary user privilege can:

- Access only those objects to which he or she has been granted privileges.
- Change his or her own password.



User Type

○ Supervisor

● User

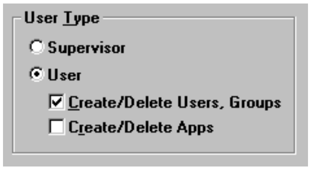
□ Create/Delete Users, Groups

□ Create/Delete Apps

Figure 16-2, Ordinary User Privilege

A user with Create/Delete Users/Groups privilege can:

- Create and edit users with equal or lower privileges only.
- Delete or rename users regardless of their privileges.
- Rename users regardless of their privileges.
- Create, edit, delete, or rename groups with equal or lower privileges only.



User Type

○ Supervisor

● User

☑ Create/Delete Users, Groups

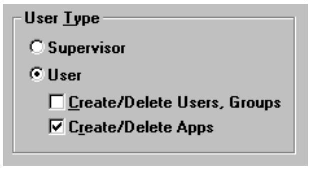
□ Create/Delete Apps

Figure 16-3, Create/Delete Users/Groups Privilege

A user with Create/Delete Applications privilege can:

- Create, modify, and delete applications and databases.
- Assign user access privileges to applications and databases at the Global Access layer.
- Define and assign filter objects.
- Remove data locks.

Users with Create/Delete Applications privilege cannot create or delete users, but they can manage global access to those applications which they have created. For more information on global access privileges, see "Managing Security at the Global Access Layer" on page 16-26.



User Type

○ Supervisor

● User

□ Create/Delete Users, Groups

☑ Create/Delete Apps

Figure 16-4, Create/Delete Applications Privilege

Users and groups can also have Application Designer or Database Designer privilege on an application or database basis. For more information about those settings, see "Application Designer Privilege" on page 16-31 or "Database Designer Privilege" on page 16-32.

Managing Users and Groups

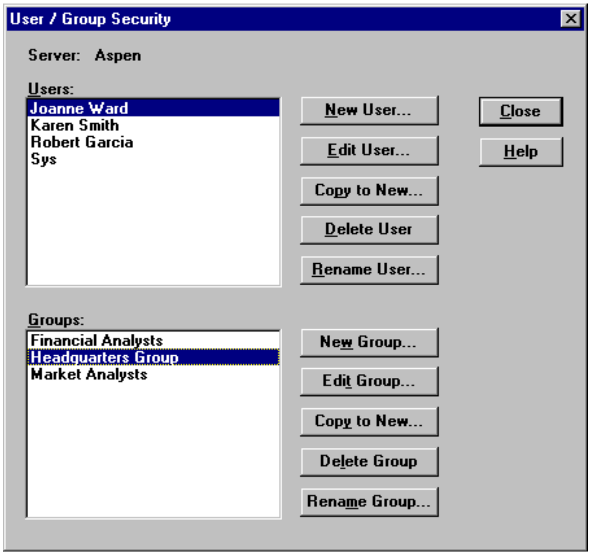
To help you manage security between users and groups, the following user-management tasks are available at varying degrees to differently privileged users:

- Create new users and groups
- Edit (modify the privileges of) existing users and groups
- Copy the security profiles of existing users and groups
- Delete users and groups
- Rename users and groups

To begin managing users and groups:

1. Connect to the server that houses the users or groups.
2. Choose Security | Users/Groups.

   Essbase displays the following dialog box:



User / Group Security

Server: Aspen

Users:

Joanne Ward
Karen Smith
Robert Garcia
Sys

Groups:

Financial Analysts
Headquarters Group
Market Analysts

Figure 16-5, User/Group Security Dialog Box

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 appearing at the right of each list box let you 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 16-8, “Copying an Existing Security Profile” on page 16-14, “Deleting Users and Groups” on page 16-17, or “Renaming Users and Groups” on page 16-18.

For information about lock management and password/user name management, see “Managing User Activity at the Server Level” on page 16-32.

You can also use the LISTUSERS and LISTGROUPS commands in ESSCMD to view a list of users and groups on the server. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Creating, Editing, and Copying Users and Groups

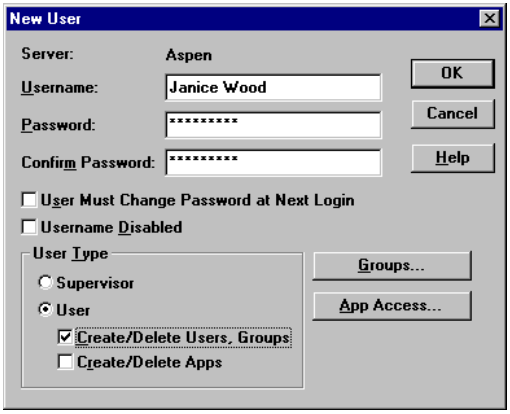
When you create, edit, or copy a user or a group, you are actually creating and modifying a security profile. This is where you define the extent of the privileges users and groups have in dealing with each other and in accessing applications and databases. For even more specific data–level security, see Chapter 17, Controlling Access to Database Cells.

Creating a New User

To create a user means to define the user's name, password, and privilege and access specifications. You can also specify group membership for the user, and you can specify that the user be required to change the password at the next login attempt, or that the user name be disabled for any reason.

To create a new user:

1. Choose Security | Users/Groups.
2. Click **New User** in the **User/Group Security** dialog box. Essbase displays the following dialog box:



New User

Server: Aspen

Username: Janice Wood

Password: **********

Confirm Password: **********

☐ User Must Change Password at Next Login

☐ Username Disabled

User Type

○ Supervisor

● User

☑ Create/Delete Users, Groups

☐ Create/Delete Apps

Figure 16-6, New User Dialog Box

3. Type the name of the new user in the **Username** text box.
4. Type the user's password in the **Password** text box. The password must be at least six characters long.

   As you type, Essbase masks your typing with asterisks.

5. 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 aren’t case-sensitive.

6. To force the user to change his or her password at the next login attempt, check **User Must Change Password at Next Login**. (This option lets you assign a generic password to all new users, which will then change when they begin using the system.)

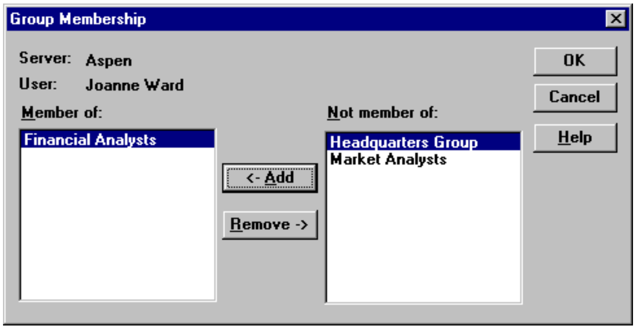
   When this user tries to log in, he or she will be prompted to first change the password in the **Change Password** dialog box, shown in Figure 16-9.

7. To lock the user out from the system for any reason, check **Username Disabled**. Only a system administrator (a user with Supervisor privilege) can re-enable the user name.

8. Choose the class of user to create from the **User Type** group. If you aren’t sure which type of user to create, see “User and Group Types” on page 16-4. If you don't have sufficient privileges to create a class of user, those options are disabled.

9. To assign the user to a group, click **Groups**.

   Essbase displays the following dialog box:



Group Membership

Server: Aspen

User: Joanne Ward

Member of:

Financial Analysts

Not member of:

Headquarters Group

Market Analysts

Figure 16-7, Group Membership Dialog Box

The **Not member of** list box contains the names of all groups on the server to which this user does not belong.

- Choose 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.
- Click OK to finalize the addition or removal.

10. To assign application and database access to the user, click **App Access** from the **New User** dialog box. See “Defining Global Application Access” on page 16-27 and “Defining Global Database Access” on page 16-30 for more information.
11. 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.

To create new users, you can also use the CREATEUSER command in ESSCMD. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

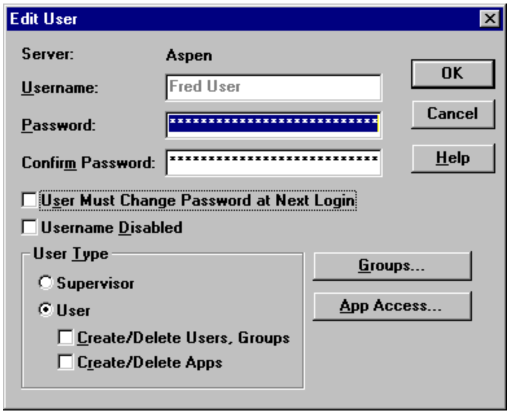
To add and remove users from groups, you can also use the ADDUSER and REMOVEUSER commands in ESSCMD. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Editing a User

To edit a user means to modify the security profile established when the user was created. Any privilege or limitation that you do not assign when creating a new user can be specified later, using the “Edit User” capability. The dialog boxes for editing a user and for creating a new one are exactly the same (except for their titles).

1. 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:



Edit User

Server: Aspen

Username: Fred User

Password: ****************************

Confirm Password: ****************************

☐ User Must Change Password at Next Login

☐ Username Disabled

User Type

○ Supervisor

● User

☐ Create/Delete Users, Groups

☐ Create/Delete Apps

Figure 16-8, Edit User Dialog Box

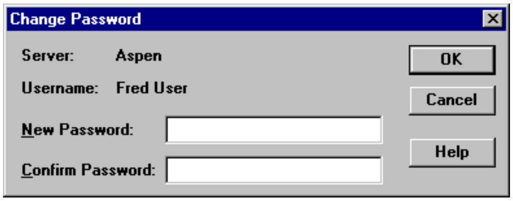
2. To change the user’s password, enter the new password in the **Password** text box. Essbase masks your typing with asterisks.

3. 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 aren’t case-sensitive.

4. To force the user to change his or her password at the next login attempt, check **User Must Change Password at Next Login**.

   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:



Change Password

Server: Aspen

Username: Fred User

New Password:

Confirm Password:

Figure 16-9, Change Password Dialog Box

5. To lock the user out from the server for any reason, check **Username Disabled** (in the **Edit User** dialog box). Only a system administrator (a user with Supervisor privilege) can re-enable the username.

6. To change the user’s group membership, click **Groups**.

7. To change the user’s application access, click **App Access**.

8. Click OK to save the user’s new security profile on the server. Essbase updates the server security file with your changes.

Note: 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 16-18.

Essbase EssCmd

To change a user’s password, you can also use the SETPASSWORD command in ESSCMD. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

What are “Groups” and How Do You Create Them?

A group is a collection of users who share the same minimum access privileges. It is helpful to place users in groups because it saves you the time of assigning identical privileges to users again and again.

A member of a group may have privileges beyond those assigned to the group, if they are 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, privileges, and access specifications 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.

Creating or Editing a Group

To create a new group or edit an existing group:

1. Choose Security | Users/Groups.

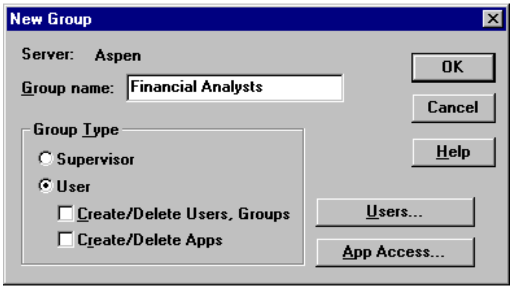
   Essbase displays the **User/Group Security** dialog box (see Figure 16-5).

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 16-18.

Essbase displays the following dialog box:



Server: Aspen

Group name: Financial Analysts

Group Type

○ Supervisor

● User

□ Create/Delete Users, Groups

□ Create/Delete Apps

Figure 16-10, New Group Dialog Box

3. Type the name of the group in the **Group name** text box.
4. Choose the type of group to create from the **Group Type** group. If you don't have sufficient privileges to assign a group to a certain type, those options are disabled.
5. To assign users to the group:
   - Click **Users**. The **Group Membership** dialog box appears, with the names of all users on the server listed either as members or non-members.
   - Choose which users should be in the group.
   - For more information on using the **Group Membership** dialog box to assign users to groups, click Help or see the instructions accompanying Figure 16-7.
   - 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 "Defining Global Application Access" on page 16-27 and "Defining Global Database Access" on page 16-30 for more information.
7. Click OK to add the new group to the server. Essbase updates the **Groups** list box (in the **User/Group Security** dialog box) with the new (or edited) group.

To simply view a list of users in a selected group, click **Edit Group** and then click **Users**. The **Members** list box of this dialog box contains a list of the group's users.

You can also use the LISTGROUPUSERS command in ESSCMD to view a list of users in a group. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

To create a new group, you can also use the CREATEGROUP command in ESSCMD. See the online *Technical Reference* in your `DOCS` directory for information about this command. Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Copying an Existing Security Profile

An easy way to create a new user with the same privileges 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 privileges the creator does not have from the copy. For example, a user with Create/Delete Users privilege cannot create a new supervisor by copying the profile of an existing supervisor.

Copying a User or Group 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 privileges in cases where you want them to be identical.

To create a new user by copying the security profile of an existing user:

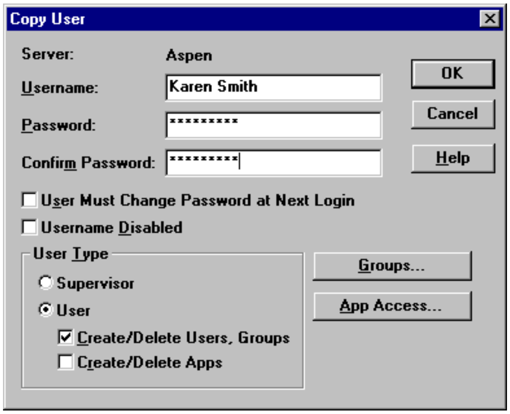
1. Choose Security | Users/Groups.

   Essbase displays the **User/Group Security** dialog box (see Figure 16-5).

2. Select the name of the user whose profile you want to copy.

3. Click **Copy to New**.

   Essbase displays the following dialog box:



Copy User

Server: Aspen

Username: Karen Smith

Password: **********

Confirm Password: **********

□ User Must Change Password at Next Login

□ Username Disabled

User Type

○ Supervisor

● User

☑ Create/Delete Users, Groups

□ Create/Delete Apps

Figure 16-11, Copy User Dialog Box

4. Type the name of the new user in the **Username** text box.
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 aren't case-sensitive.

7. To force the new user to change his or her password at the next login attempt, check **User Must Change Password at Next Login**.
8. To lock the new user out from the system for any reason, check **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 privilege can reactivate the user name.
9. To change the new user's security class, choose the new class from the **User Type** group. If you don't have sufficient privileges to assign certain classes to the user, those options are disabled.
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 your changes.

Copying a Group Profile

To create a new group by copying the security profile of an existing group:

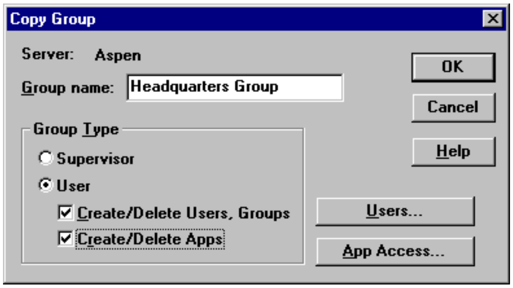
1. Choose Security | Users/Groups.

   Essbase displays the **User/Group Security** dialog box (see Figure 16-5).

2. Select the name of the group you want to copy.

3. Click **Copy to New**.

   Essbase displays the following dialog box:



Server: Aspen

Group name: Headquarters Group

Group Type

○ Supervisor

● User

☑ Create/Delete Users, Groups

☑ Create/Delete Apps

Figure 16-12, Copy Group Dialog Box

4. Type the name of the new group in the **Group name** text box.

5. To change the new group's security class, choose the new class from the **Group Type** group. If you don't have sufficient privileges to assign certain classes to the group, those options are disabled.

6. To change the new group's membership list, click **Users**.

   The **Group Membership** dialog box appears.

   For more information on using the **Group Membership** dialog box to assign users to groups, click Help, or see the instructions accompanying Figure 16-7.

7. To change the group's application access, click **App Access**.

8. Click OK to save the new group's security profile on the server. Essbase updates the server security file with your changes.

Deleting Users and Groups

To delete a user:

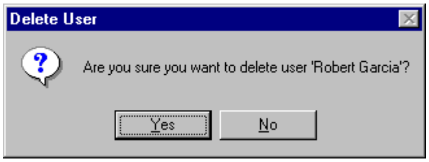
1. Choose Security | Users/Groups.

   Essbase displays the **User/Group Security** dialog box (see Figure 16-5).

2. Select the name of the user you want to delete in the **Users** list box.

3. Click **Delete User**.

   Essbase displays the following confirmation box:



Delete User

Are you sure you want to delete user 'Robert Garcia'?

Figure 16-13, Delete User Confirmation Box

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:

1. Choose Security | Users/Groups.

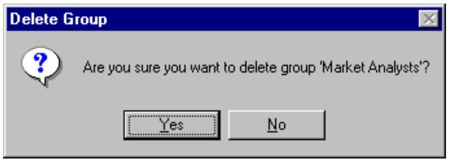
   Essbase displays the **User/Group Security** dialog box (see Figure 16-5).

2. Select the name of the group you want to delete in the **Groups** list box.

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 following confirmation box:



Delete Group

Are you sure you want to delete group 'Market Analysts'?

Figure 16-14, Delete Group Confirmation Box

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.

You can also use the DELETEUSER and DELETEGROUP commands in ESSCMD to perform these tasks. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Renaming Users and Groups

To rename a user:

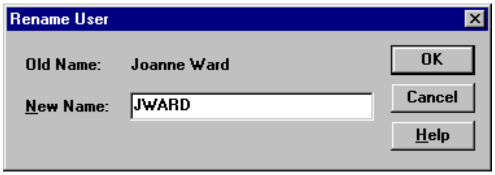
1. Choose Security | Users/Groups.

   Essbase displays the **User/Group Security** dialog box (see Figure 16-5).

2. Select the name of the user in the **Users** list box.

3. Click **Rename User**.

   Essbase displays the following dialog box:


Rename User

Old Name: Joanne Ward

New Name: JWARD

Figure 16-15, Rename User Dialog Box

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:

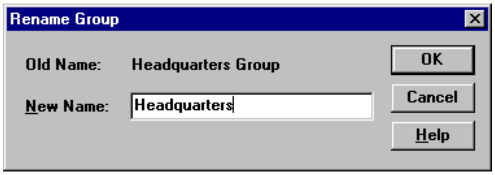
1. Choose Security | Users/Groups.

   Essbase displays the **User/Group Security** dialog box (see Figure 16-5).

2. Select the name of the group in the **Groups** list box.

3. Click **Rename Group**.

   Essbase displays the following dialog box:



Rename Group

Old Name: Headquarters Group

New Name: Headquarters

Figure 16-16, Rename Group Dialog Box

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.

To rename a user, you can also use the RENAMEUSER command in ESSCMD. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Modifying User Application and Database Access Settings

By default, users and groups inherit the global application and database settings, which become their security privileges. A user can, however, have application and database privileges that go beyond the global defaults. These settings can be defined by a system administrator when creating a new user or editing an existing user. There is no need to define the settings for Supervisors—they are automatically granted App Designer access (full privileges) to every application on the server.

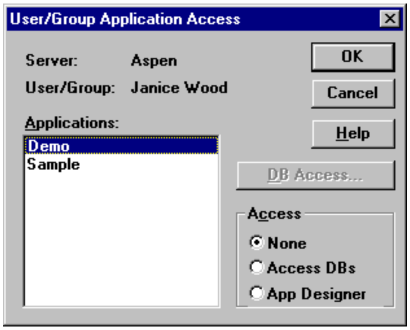
To modify application or database access settings for a group, follow the instructions below pertaining to a user, substituting the word “group” where you see “user.”

To modify application or database access settings for a user:

1. Choose Security | Users/Groups.
2. Select a user name, then click **Edit User**.

3. Click **App Access** in the **Edit User** dialog box. (This button is disabled if the user being edited is a Supervisor.)

   Essbase displays the following dialog box:



Server: Aspen
User/Group: Janice Wood
Applications:
Demo
Sample

Access
● None
○ Access DBs
○ App Designer

Figure 16-17, User/Group Application Access Dialog Box

   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. If you have not yet assigned privileges to this user, the default access setting is **None**.

4. Select the appropriate application and access option.

   - Choose **None** to give the user no access to the selected application or any of its databases.
   - Choose **Access DBs** to define specific database access within the selected application.
   - Choose **App Designer** to give the user most of the privileges available to a Supervisor, but for the selected application only. The user with Application Designer privilege has full access to the application, plus Create/Delete privileges for databases within the application, as well as the ability to log users out from that application, define and assign filters, and remove data locks.

5. Click OK to save the settings, unless you need to define database-level access settings. If you want to define database access settings, see "Assigning Database Access to a User" on page 16-21.

Assigning Database Access to a User

There is no need to assign database access for Supervisors, or for those with Application Designer privilege for the application or Database Designer privilege for the database. These users already have full database access.

You need to assign database access to other users if:

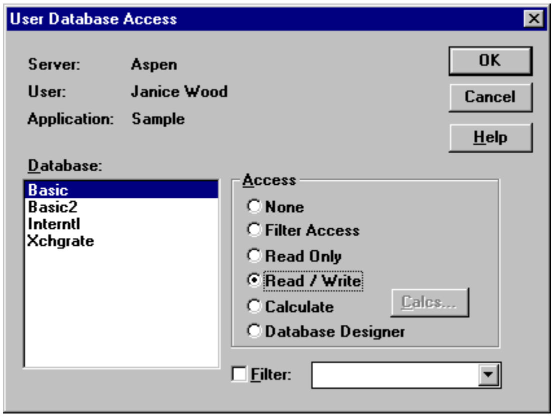
- The users have not been granted sufficient user privileges to access databases (see the privileges shown in Table 16-1).
- The database in question does not allow users sufficient access through its global settings (see “Minimum Database Access” on page 16-28).
- The users have no access granted to them through filters (see Chapter 17, Controlling Access to Database Cells).

If you need to assign access to databases within the selected application, proceed as follows:

1. Select **Access DBs** from the **User/Group Application Access** dialog box (available by clicking **App Access** in the **Edit User** dialog box—see Figure 16-17). The **DB Access** button then 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 are automatically given Database Designer access to every database within the application.

2. Click **DB Access**. Essbase displays the **User Database Access** dialog box:



User Database Access

Server: Aspen
User: Janice Wood
Application: Sample

Database:
Basic
Basic2
Interntl
Xchgrate

Access
○ None
○ Filter Access
○ Read Only
● Read / Write
○ Calculate
○ Database Designer

□ Filter:

Figure 16-18, User Database Access Dialog Box

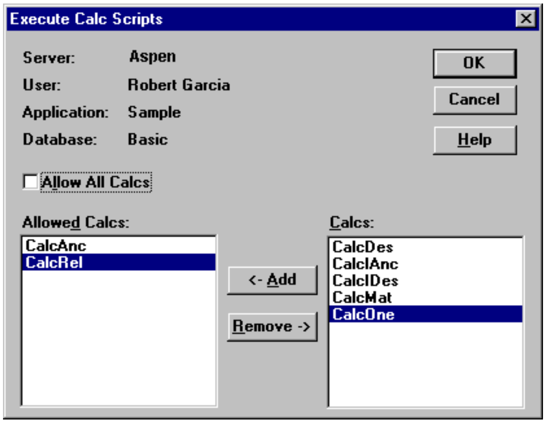
The **Database** list box shows all databases defined within the application to which you have access. When you select a database, the access the user has for the selected database appears in the **Access** group.

If the user is not a Supervisor, you can give the user one of the following access levels:

User Access Level	Privilege 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 17, Controlling Access to Database Cells.)
Read Only	Indicates read access 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 outline or any calc script to which the user has access.
Database Designer	Indicates that all data values can be retrieved, updated, and calculated. In addition, all database-related files can be modified.
Filter	Associates a filter object with a user name. A user can have one filter per database. (For information about filters, see Chapter 17, Controlling Access to Database Cells.) Checking this option or any other option except **None** enables the selection of a filter object from the list box.

3. Select the appropriate database and the access privileges you want to apply.

   If you choose the Calculate access level, the **Calcs** button lets you define calc script execution access. Users can run any server-based calc script (provided they have sufficient security privileges) from the Application Manager or a Hyperion Essbase Spreadsheet Add-in. When you click **Calcs**, Essbase displays the following dialog box:



Server: Aspen

User: Robert Garcia

Application: Sample

Database: Basic

□ Allow All Calcs

Allowed Calcs:
CalcAnc
CalcRel

Calcs:
CalcDes
CalcIAnc
CalcIDes
CalcMat
CalcOne

Figure 16-19, Execute Calc Scripts Dialog Box

The **Allow All Calcs** check box lets you give the user access to all calc scripts on the server. Any scripts defined afterward are automatically added to the user's calculate privileges. Individual calc script privileges can be added or removed by selecting the script and clicking **<-Add** or **Remove->**.

Note: By default, a Supervisor, Application Designer, or Database Designer can run all calc scripts.

4. Click OK to close the dialog box and save the settings.

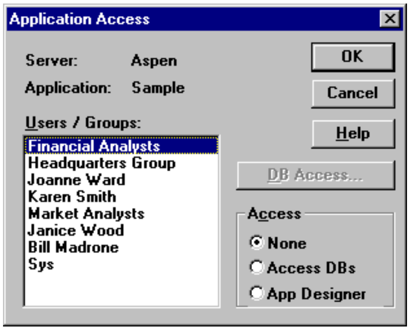
Viewing and Modifying User Access Privileges

The security system lets you view all users and groups on the server from a list. You can easily make changes to their application and database access levels from this same list. This enables you to effectively maintain a security plan for a large number of users.

Viewing and Modifying User Application Access

1. From the Application Desktop window, select the application name.
2. Choose Security | Application.

   Essbase displays the following dialog box, showing all users and groups on the server:



Application Access

Server: Aspen

Application: Sample

Users / Groups:

Financial Analysts
Headquarters Group
Joanne Ward
Karen Smith
Market Analysts
Janice Wood
Bill Madrone
Sys

Access

● None
○ Access DBs
○ App Designer

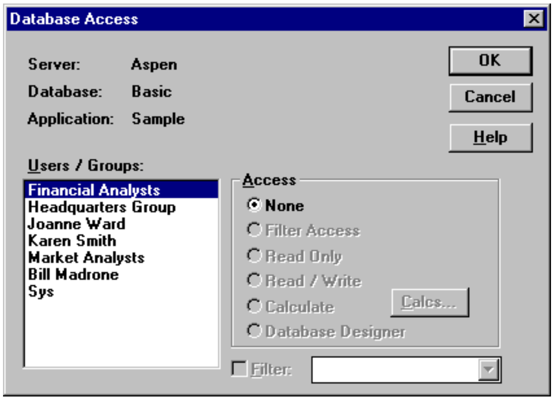
Figure 16-20, Application Access Dialog Box

3. To view the current access settings for a user or group, select the user or group name from the **Users/Groups** list box.

   The **Access** group shows the current access level for the selected user or group. Click Help for information on each setting.
4. To modify the current application access settings for the selected user or group, choose the appropriate level from the **Access** group.
5. To define access to databases within the application, choose **DB Access** from the **Application Access** dialog box, or see “Viewing and Modifying User Database Access” on page 16-25.
6. Click OK to save the settings.

Viewing and Modifying User Database Access

1. From the Application Desktop window, select the application and database name.
2. Choose Security | Database. Essbase displays the **Database Access** dialog box:



Server: Aspen

Database: Basic

Application: Sample

Users / Groups:

- Financial Analysts
- Headquarters Group
- Joanne Ward
- Karen Smith
- Market Analysts
- Bill Madrone
- Sys

Access

● None

○ Filter Access

○ Read Only

○ Read / Write

○ Calculate

○ Database Designer

□ Filter:

Figure 16-21, 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 settings for a user or group, select the user or group name from the list box and select the access setting you want to apply.
4. Click OK to save the settings.

Note: If a user has insufficient privileges to access the data in a database, the value shows up in the spreadsheet as `#NOACCESS`.

Managing Security at the Global Access Layer

The Global Access layer pertains directly to the security-access settings for applications and databases and their related files. Application and database security settings are based on the minimum database access privilege granted to all users. 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 privileges do not specify Read access. Similarly, if a database has the privilege None assigned, only users with higher access privileges (granted at the user, group, or database filter layer) can gain access to the database.

Users with Supervisor privilege, Application Designer privilege for the application, or Database Designer privilege for the database are not affected by these settings. Supervisors automatically have full access, and Application Designers and Database Designers have full access only for their applications or databases.

By default, users and groups inherit the global access settings, which become their security privileges. A user can, however, have application and database privileges that go beyond the global defaults. For more information on application and database privileges defined at the user level, see “Modifying User Application and Database Access Settings” on page 16-19.

The following access privileges are available in the Global Access layer. These privileges apply to applications and databases.

Access Level	Privilege Description
None	Specifies no access to any file or data value in the application or database. This is the default access privilege assigned when an application or database is created.
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 application or database. Users can view Essbase files, retrieve and update data values, and run report scripts. Write access does not permit calculations or outline modifications.

Access Level	Privilege Description
Calculate	Specifies Calculate and update access to any data value in the application or database. Users can view files, retrieve, update, and perform calculations based on data values, and run report and calc scripts. Calculate access does not permit outline modifications.
DB Designer	Specifies Calculate and update access to any data value in the application or database. In addition, DB Designer access lets users view and modify the outline and files, retrieve, update, and perform calculations based on data values, and run report and calc scripts.

Databases within applications inherit the privileges of the applications whenever the application's access settings are higher than those of the database.

Defining Global Application Access

You can define access settings and other settings that apply to applications on a global level. The settings you define for the application affect all users, unless they have higher privileges granted to them at the user level. The following application settings are available:

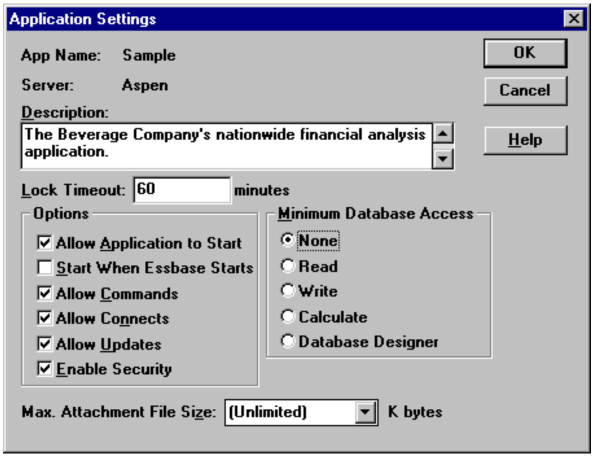
- A brief description of the application
- A time-out setting for locks
- Options that control application loading, command processing, connections, updates, and security
- Settings that define the minimum access level to databases within the application

Only users with Supervisor privilege (or Application Designer privilege for the application) can change Global Access settings for applications.

To define settings for an application:

1. Connect to the appropriate server and select the name of the application you want to secure.
2. Choose Application | Settings. (This command is unavailable to users with insufficient privileges to define application settings.)

   Essbase displays the following dialog box:



Application Settings

App Name: Sample

Server: Aspen

Description: The Beverage Company's nationwide financial analysis application.

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

Figure 16-22, Application Settings Dialog Box

3. Define the settings that you want to apply to the application. Click Help for information on each setting.

Minimum Database Access

The Global Access privileges are listed in the **Minimum Database Access** group. All databases within the application (as well as any databases created after the settings are defined) inherit the settings specified in the **Application Settings** dialog box (see Figure 16-22), unless they are changed at the database level.

Changes to the **Minimum Database Access** settings for applications affect only those databases that have lower access privileges. Assigning privileges at one level never takes away privileges that have been granted at another, except in the case of filters (for more information about filters, see Chapter 17, Controlling Access to Database Cells).

For example, an application with a setting of Write contains two databases. The first database has had no higher access privileges granted, and so it inherits the application's Write setting. The second database has been assigned a minimum database access of Calculate. The application setting of Write does not affect the second database because Calculate is a higher privilege than Write.

If you were to change the application settings from Write to a minimum database access setting of Read, this would lower the first database's access level to Read. (The Write privilege is taken away only because the database was never assigned privileges at the database level—it has inherited the application's settings by default.) The second database, which has been defined with a higher privilege at the database level, would remain with the original setting of Calculate.

Allow Settings

In the **Application Settings** dialog box, all "Allow" settings (**Allow Application to Start**, **Allow Commands**, **Allow Connects**, and **Allow Updates**) override other security and access settings defined for users, with the exception of supervisors. When a supervisor clears any of the Allow check boxes, other supervisors are not affected by the change.

All Allow settings (**Allow Commands**, **Allow Connects**, and **Allow Updates**) are checked by default. If a supervisor unchecks a setting, it is not rechecked when the supervisor disconnects from an application or database.

When a supervisor clears **Allow Commands**, all other users (except supervisors) are immediately affected by the change. Changes to other application settings don't affect users currently connected to the application.

Warning: *Never* power down or reboot your client machine when you have cleared any of the Allow settings. (Always choose 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 the Essbase server to improperly disconnect from the Essbase client, and your application is no longer accessible, you will need to shut down and restart the application. See Chapter 44, Running Essbase, Applications, and Databases, for more information.

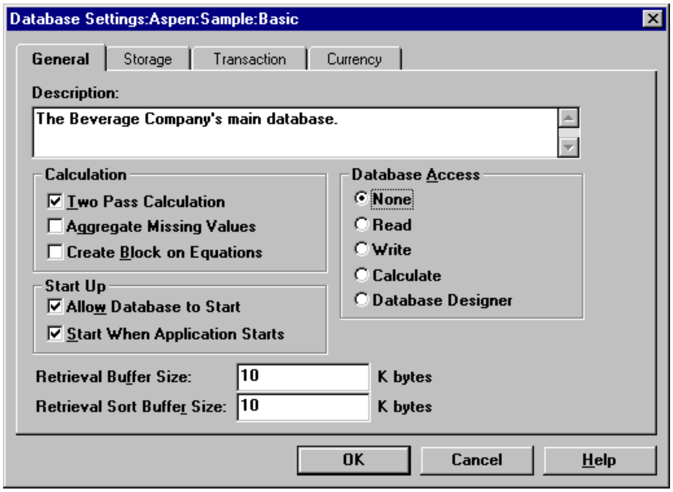
Defining Global Database Access

When you create a database, it inherits the Global Access settings defined for the application (see “Minimum Database Access” on page 16-28). However, any database within an application can be defined with its own Global Access settings, which will override the application’s Global Access settings.

To define settings 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. Choose Database | Settings. (This menu command is unavailable to users with insufficient privileges to define database settings.)

   Essbase displays the following dialog box:



Description:
The Beverage Company's main database.

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

Figure 16-23, Database Settings Dialog Box

4. In the **General** tab, define the settings that you want to apply to the database. Click Help for information on each setting.

   The Global Access privileges 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.

Assigning Global Access Settings Per User

Users and groups can be assigned Application Designer or Database Designer privilege on an application or database basis. These settings are useful for assigning supervisory privileges to users who need to be in charge of particular applications or databases, but who only need ordinary user privileges for other projects.

Application Designer Privilege

If you have Application Designer privilege for an application, you have complete access to all objects in that application. (You cannot create or delete an application unless you also have been granted that privilege on the user level.) If you have Application Designer privilege, you can do the following:

- Modify any object within the application
- Create, modify, and delete databases within the application
- Assign user access privileges at the application or database level for the application
- Define and assign filter objects anywhere in the application
- Remove data locks within the application
- Disconnect users who are connected to the application or any application for which you have Application Designer privilege

Application Designer privilege applies only to the assigned application. Outside of the application, you revert to the privileges of an ordinary user.

Note: If you are a user with Create/Delete Apps privilege, you are automatically given Application Designer privilege for any application you create. Therefore, you also have complete access to all objects in the application.

For a given database, users or groups can be assigned any one of the following privilege levels: None, Filter Access, Read Only, Read/Write, Calculate, and Database Designer.

Database Designer Privilege

If you have Database Designer privilege, you have complete access to all objects in the database. You cannot create or delete a database, but you can do the following:

- Modify any object within the database
- Restrict data access privileges at the database level for the database
- Define and assign filter objects anywhere in the database
- Remove data locks within the database

Database Designer privilege applies only to data access for the assigned database. Outside of the database, you revert to the privileges of an ordinary user.

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 lock management, connection management, and password/user name management. For information about managing security for partitioned databases, see Chapter 6, Designing Partitioned Applications.

Managing Locks

Spreadsheet Add-in users can interactively send data from a spreadsheet to the server. To maintain data integrity while providing multi-user concurrent access, Essbase lets users lock data for the purpose of updating it. If a user wants to update data, he or she must first lock the records to prevent other users from trying to change the same data.

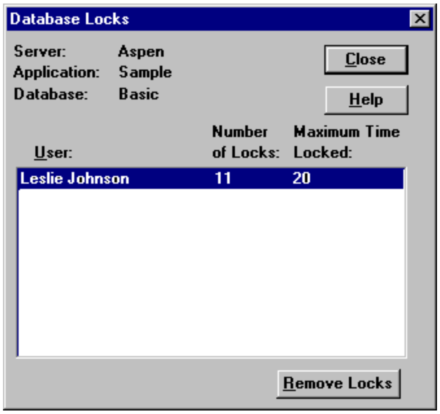
The default maximum lock time is 3600 seconds, or 60 minutes. To prevent data from becoming inaccessible for long periods, Essbase automatically unlocks data that remains locked beyond the allotted time. A user with Supervisor or Application Designer privilege can modify the maximum lock time setting.

Occasionally, you may need to force an unlock operation before the allotted time expires. 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.

The security system allows only Supervisors to view users holding locks and to remove the locks.

1. To view or remove locks, choose Security | Locks.

   Essbase displays the **Database Locks** dialog box:



Database Locks

Server: Aspen
Application: Sample
Database: Basic

User:	Number of Locks:	Maximum Time Locked:
Leslie Johnson	11	20

Figure 16-24, Database Locks Dialog Box

   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**.

Note: Removing a lock does not disconnect the user from his or her session.

You can also use the REMOVELOCKS command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Disconnecting Users

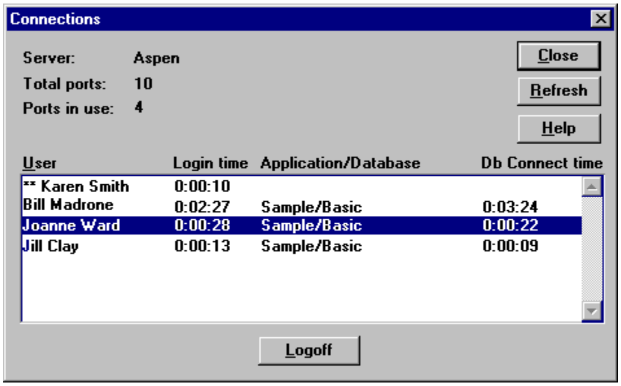
The security system lets you disconnect a user from the Essbase server when you want to restructure an outline or load data.

A user with Supervisor or Application Designer privilege can disconnect a user connected to a particular application and database.

To disconnect a user:

1. Choose Security | Connections.

   Essbase displays the **Connections** dialog box:



Server: Aspen
Total ports: 10
Ports in use: 4

User	Login time	Application/Database	Db Connect time
** Karen Smith	0:00:10		
Bill Madrone	0:02:27	Sample/Basic	0:03:24
Joanne Ward	0:00:28	Sample/Basic	0:00:22
Jill Clay	0:00:13	Sample/Basic	0:00:09

Figure 16-25, Connections Dialog Box

   If you have Supervisor privilege, this dialog box lists the following:

   - All users connected to the same server you are connected to.
   - The application and database each user is using. If no application and database name appear next to a user name, then the user is logged in but not connected to a specific application.
   - Your user name, which is preceded by two asterisks (**). You cannot disconnect yourself using this dialog box.

   If you have Application Designer privilege, this dialog box lists the following:

   - All users, including yourself, who are connected to any application for which you have Application Designer privilege. Your user name is preceded by two asterisks (**). You cannot disconnect yourself using this dialog box.
   - The application and database each user is using.

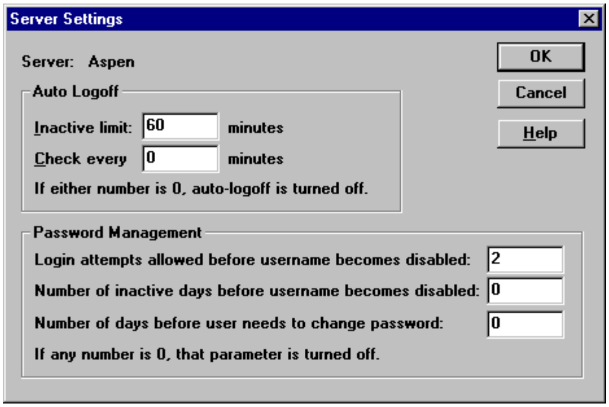
2. Select a user name from the list and click **Logoff** to disconnect the user.

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.

1. To place these settings, choose Server | Settings.

   Essbase displays the **Server Settings** dialog box.



Server Settings

Server: Aspen

Auto Logoff

Inactive limit: 60 minutes

Check every 0 minutes

If either number is 0, auto-logoff is turned off.

Password Management

Login attempts allowed before username becomes disabled: 2

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.

Figure 16-26, Server Settings Dialog Box

The **Password Management** option group contains the settings for user management. A setting of 0 for any option means that 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 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, as soon as you clicked OK, 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 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 same password, enter the number in the third text box.

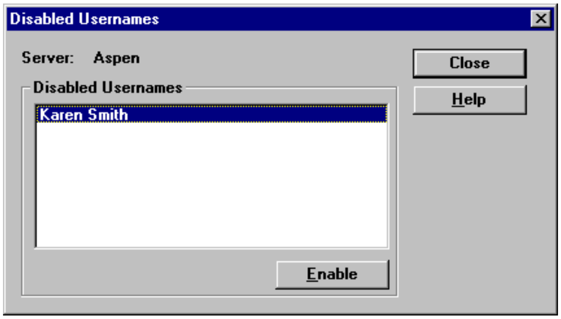
   The timer starts for all users as soon as you click OK, and it is reset for particular users each time they change their passwords or are reactivated or edited by Supervisors.

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 16-35), 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 16-10.

1. To view or activate currently disabled user names, choose Security | Disabled Usernames.

   Essbase displays the **Disabled Usernames** dialog box, which lists all disabled user names:



Disabled Usernames

Server: Aspen

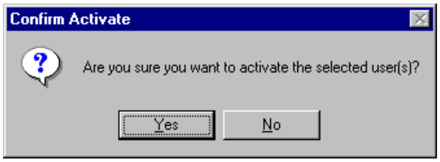
Disabled Usernames

Karen Smith

Figure 16-27, Disabled Usernames Dialog Box

2. To activate a user, select the user name from the list box and click **Enable**.

   Essbase displays a confirmation box.



Confirm Activate

Are you sure you want to activate the selected user(s)?

Figure 16-28, Confirm Activate Confirmation Box

3. Click Yes to confirm that you want to activate the selected user name.

Note: Only a system administrator (a user with Supervisor privilege) can view or reactivate disabled user names.

Chapter 17 Controlling Access to Database Cells

When the security levels defined for applications, databases, users, and groups are not enough, Hyperion Essbase's Database Filter layer gives 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 Supervisor privilege, 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 Application or Database Designer privilege, you can define and assign filters within your application(s) or database(s). For more information about user privileges, see Chapter 16, Managing Security at Global and User Levels.

This chapter contains the following sections:

Privileges at the Database Filter Layer

Filters control security access to data values, or cells. You create filters to accommodate your 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 privileges and apply them to data ranging from a list of members to an individual cell.

Access Level	Privilege 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 privileges assigned at the database access level. This occurs 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 16, Managing Security at Global and User Levels.

Calculation access is controlled by the database's global access privileges or the user's individually assigned privileges. Calculate privileges assigned globally or on a user-specific level take precedence over any filter access settings. 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.

You can also use the LISTFILTERS command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Creating a New Filter

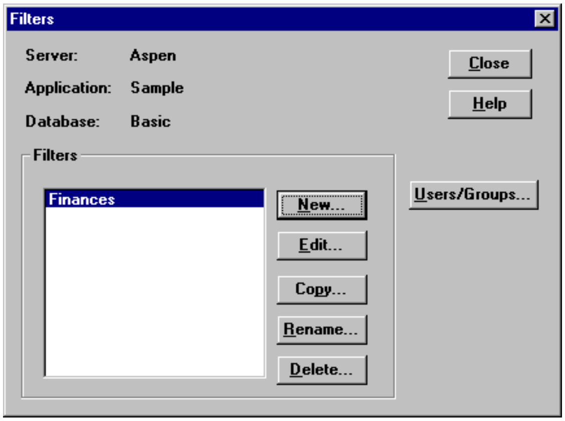
You can create a new filter for each set of access restrictions you need to place on database values. There is no need to create separate filters for users with the same access needs—once you have created a filter, you can assign it to multiple users or groups of users. However, only one filter per database can be assigned to a user or group.

1. To create a filter, 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 "Try This Filter Example" on page 17-6.

2. Choose Security | Filters.

   Essbase displays the **Filters** dialog box.

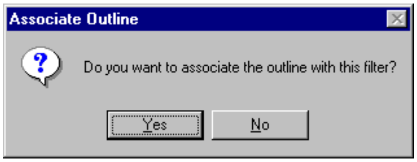


Server: Aspen
Application: Sample
Database: Basic
Filters: Finances

Figure 17-1, Filters Dialog Box

3. Click **New**.

   Essbase displays the following confirmation box.

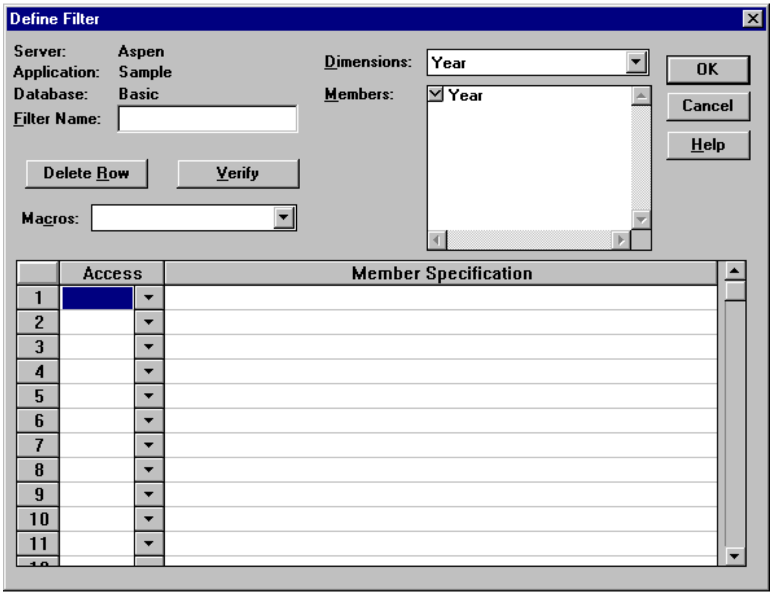


Do you want to associate the outline with this filter?

Figure 17-2, Associate Outline Confirmation Box

4. Click Yes to confirm that you want to associate the current outline with the filter.

   Essbase displays the **Define Filter** dialog box.



Server: Aspen
Application: Sample
Database: Basic
Filter Name:

Dimensions: Year
Members: ☑ Year

	Access	Member Specification
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		

Figure 17-3, Define Filter Dialog Box

   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 17-3.

   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.
8. To apply a macro to dimensions and members:
    a. Position the cursor in the **Member Specification** row and select the appropriate macro from the **Macros** list box.
    b. Specify a member for the macro 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**.

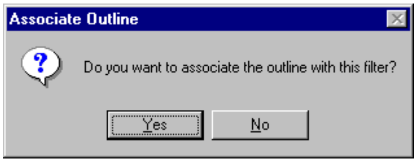
Note: For more information about macros and syntax, consult the online *Technical Reference* in your `DOCS` directory.

11. Click OK to save the filter.

Try This Filter Example

1. To create an example filter, 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 17-1).
3. Click **New**.

    Essbase displays the following confirmation box.



Associate Outline

Do you want to associate the outline with this filter?

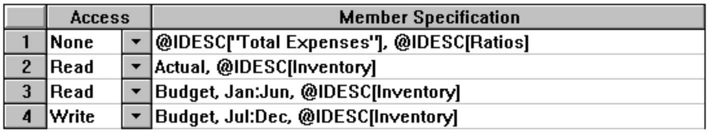
Figure 17-4, Associate Outline Confirmation Box

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 17-3).

5. Type the name `Finances` in the **Filter Name** text box.

6. Fill in the filter sheet for the sample filter, `Finances`, to match the following example:



	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]

Figure 17-5, Sample Member Specifications in the Define Filter Dialog Box

See the instructions for creating a filter (see "Creating a New Filter" on page 17-4).

This filter defines the following access plan:

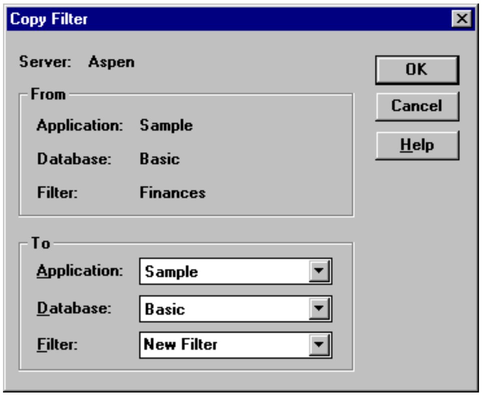
- No retrieval or update access to any data for members of the Total Expenses branch or the Ratios branch.
- Read-only access to all members of the Inventory branch below Actual.
- Read-only access to all members of the Inventory branch below Budget for the months of Jan through Jun.
- Write access to all members of the Inventory branch below Budget data for months of Jul through Dec.

Editing a Filter

1. To edit an existing filter, choose Security | Filters. Essbase displays the **Filters** dialog box (see Figure 17-1).
2. Select the filter you want to edit and click **Edit**. Essbase displays the **Associate Outline Confirmation** box, as in Figure 17-2.
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 17-3.
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 17-4.
5. Click OK to save the filter.

Copying a Filter

1. To copy an existing filter, choose Security | Filters. The **Filters** dialog box appears (see Figure 17-1).
2. Select the filter you want to copy and click **Copy**. Essbase displays the **Copy Filter** dialog box.
   - 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.


Copy Filter

Server: Aspen

From

Application: Sample

Database: Basic

Filter: Finances

To

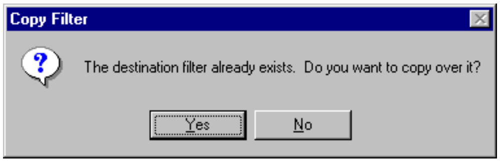
Application: Sample

Database: Basic

Filter: New Filter

Figure 17-6, Copy Filter Dialog Box

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 the following confirmation box:


Copy Filter

The destination filter already exists. Do you want to copy over it?

Figure 17-7, Copy Filter Confirmation Box

5. Click Yes to confirm that you want to replace the existing filter with the copy.
6. Click OK to save the filter.

You can also use the COPYFILTER command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

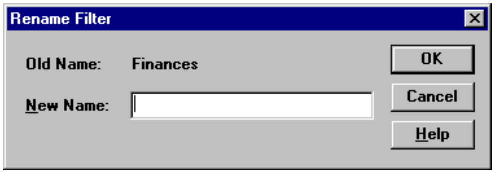
Renaming a Filter

1. Choose Security | Filters.

   Essbase displays the **Filters** dialog box (see Figure 17-1).

2. Select the filter you want to rename and click **Rename**.

   Essbase displays the **Rename Filter** dialog box.



Old Name: Finances

New Name:

Figure 17-8, Rename Filter Dialog Box

3. Type in the new name and click OK to save.

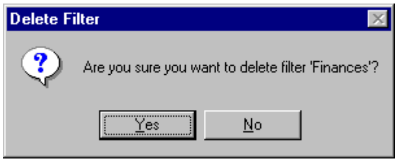
You can also use the RENAMEFILTER command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Deleting a Filter

1. Choose Security | Filters.

   Essbase displays the **Filters** dialog box (see Figure 17-1).

2. Select the filter you want to delete and click **Delete**. Essbase displays the following confirmation box:



Delete Filter

Are you sure you want to delete filter 'Finances'?

Figure 17-9, Delete Filter Confirmation Box

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 Supervisor privileges. Only one filter per database can be assigned to a user or group.

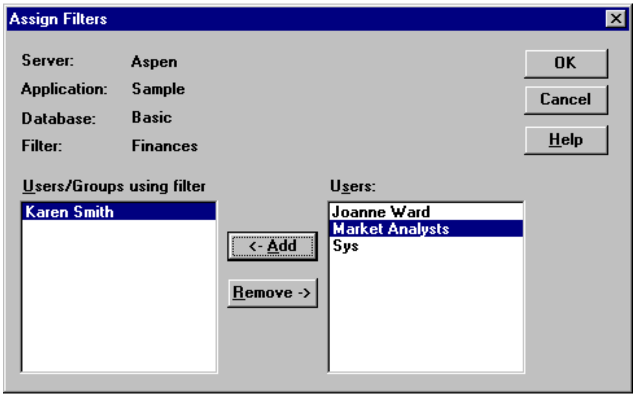
To assign a filter to a user or group:

1. Choose Security | Filters.

   The **Filters** dialog box appears as in Figure 17-1.

2. Select the filter name you want to assign and click **Users/Groups**.

   Essbase displays the **Assign Filters** dialog box:



Assign Filters

Server: Aspen
Application: Sample
Database: Basic
Filter: Finances

Users/Groups using filter:
Karen Smith

Users:
Joanne Ward
Market Analysts
Sys

Figure 17-10, Assign Filters Dialog Box

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.

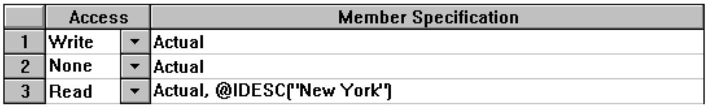
Overlapping Filter Definitions

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, the following filter contains overlap conflicts:



	Access	Member Specification
1	Write	Actual
2	None	Actual
3	Read	Actual, @IDESC("New York")

Figure 17-11, Filter with Overlap Conflicts

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 privilege than an access level of None, the remaining data values in Actual are granted Write access.

All other data points, such as Budget, inherit the database access privileges.

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

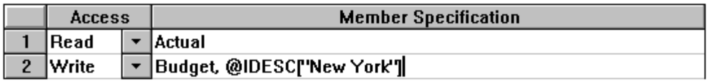
She is assigned to Group Marketing which has the following database access:

FINPLAN	N
PRODPLAN	W

Her effective rights become:

FINPLAN	R
PRODPLAN	W

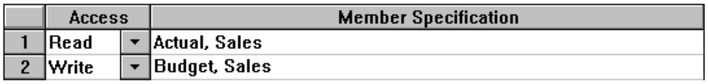
In addition, Mary uses the filter object RED (for the database FINPLAN), which has the following filter rows:



	Access	Member Specification
1	Read ▾	Actual
2	Write ▾	Budget, @IDESC["New York"]

Figure 17-12, RED Filter for Database FINPLAN

The Group Marketing also uses a filter object BLUE (for the database FINPLAN) which has the following filter rows:



	Access	Member Specification
1	Read ▾	Actual, Sales
2	Write ▾	Budget, Sales

Figure 17-13, BLUE Filter for Database FINPLAN

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 18, Security Examples.

Chapter 18 Security Examples

This chapter describes some sample security problems and solutions, which are based on the Sample application on the Hyperion Essbase OLAP Server. These examples use security procedures for which the basic instructions are found in Chapter 16, Managing Security at Global and User Levels.

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 Hyperion Essbase 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 16-6 in Chapter 16, Managing Security at Global and User Levels) 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 16-6 in Chapter 16, Managing Security at Global and User Levels) 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 16-22 in Chapter 16, Managing Security at Global and User Levels). 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 16-6 in Chapter 16, Managing Security at Global and User Levels) 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 16-22 in Chapter 16, Managing Security at Global and User Levels). 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. (Since 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 16-6 in Chapter 16, Managing Security at Global and User Levels) 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 16-22 in Chapter 16, Managing Security at Global and User Levels) 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 should allow the users to full 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.

Part IV Loading Data

Part IV describes how to load data from an external data source into existing Hyperion Essbase OLAP Server databases by describing the concepts behind data loading, how to create rules files to manipulate the records and fields in a data source, how to perform a data load, and how to debug and optimize existing data loads. Part IV contains the following chapters:

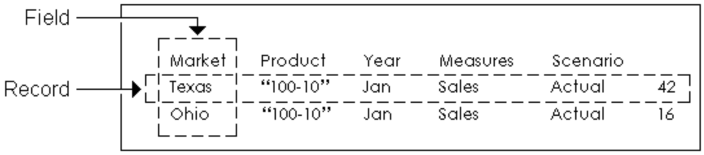
- Chapter 19, Introducing Data Loading, introduces you to the parts of data loading, including external data sources, rules files, free-form data loading, and describes how Essbase handles security and multi-user issues while loading data.
- Chapter 20, Setting up a Rules File to Manipulate Records, 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 21, 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 22, Performing a Data Load, describes how to load data using 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 provides tips for loading data.
- Chapter 23, Debugging and Optimizing Data Loads, describes how to debug problems with data loads and how to optimize your data loads so that Essbase can load the data more quickly.

Chapter 19 Introducing Data Loading

Data loading is the process of copying data from external data sources, such as spreadsheets or SQL databases, into a Hyperion Essbase OLAP Server 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

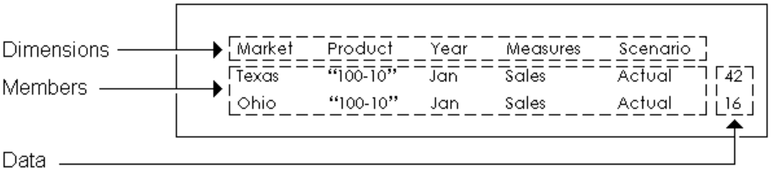
As illustrated in Figure 19-1, a data source is composed of records and fields. A *record* is a row of fields that is read as a unit. A *field* is a vertical list of values.



Market	Product	Year	Measures	Scenario	
Texas	"100-10"	Jan	Sales	Actual	42
Ohio	"100-10"	Jan	Sales	Actual	16

Figure 19-1, Records and Fields

As illustrated in Figure 19-2, 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.



Market	Product	Year	Measures	Scenario	
Texas	"100-10"	Jan	Sales	Actual	42
Ohio	"100-10"	Jan	Sales	Actual	16

Figure 19-2, 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 19-2, 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 19-8.
- 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 19-17.

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 19-2, 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:

Guidelines	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 21-14.

Valid Member Fields

A *member field* contains the name of a member or alias in a dimension. In Figure 19-2, for example, Texas and Ohio are members of the Market dimension. Member fields must be formatted as follows:

Rules	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 were a Dynamic Calc member, you could not 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: • Whitespace • Numeric characters (0-9) • Dashes (minus signs) • 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 information about using & (ampersand) in dimension or member names, press the Help button in the **Substitution Variables** dialog box.
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.

```
East Cola  Actual
Sales      Jan       $10
           Feb       $21
           Mar       $15-
           Apr       $16
```

Figure 19-3, Invalid Data Field

For information on continuing the load, see “Loading the Error Log File” on page 23-4.

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 19-4, for example, the fields are separated by spaces. Essbase ignores the extra spaces between East and Cola in the first record.

```
East     Cola     Actual     Jan     Sales    10
East     Cola     Actual     Feb     Sales    21
East     Cola     Actual     Mar     Sales    30
```

Figure 19-4, File Delimiters

For more information, see “Setting File Delimiters” on page 20-9.

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 19-5, but loads the other fields normally.

```
East Actual "100-10"
       Sales     Marketing
       =====     =========
Jan    10        8
Feb    21        16
```

Figure 19-5, Ignoring Formatting Characters During Loading

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.

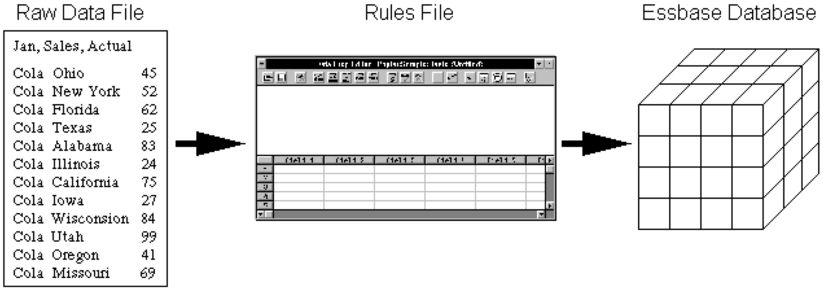


Figure 19-6, Loading Data Sources through Rules Files

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.

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 20, Setting up a Rules File to Manipulate Records, and Chapter 21, Manipulating Fields Using a Rules File, for information about manipulating fields and records. See Chapter 12, 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 could choose an SQL data source, a spreadsheet, or a text file. See “Selecting the Data Source” on page 20-1.
2. Set the file delimiter for your data source. For example, you could choose tabs, commas, or spaces. See “Setting File Delimiters” on page 20-9.
3. Perform operations on records. For example, you could set up header records, and define select and reject operations. See Chapter 20, Setting up a Rules File to Manipulate Records.
4. Perform operations on fields. For example, you could split them, join them, and create new ones. See Chapter 21, Manipulating Fields Using a Rules File.
5. Map fields in the data source to dimensions and members in the database. See “Mapping Fields to Member Names” on page 21-11.
6. Save and validate the data load rules. For more information, see “Validating and Saving Data Load Rules” on page 20-18.

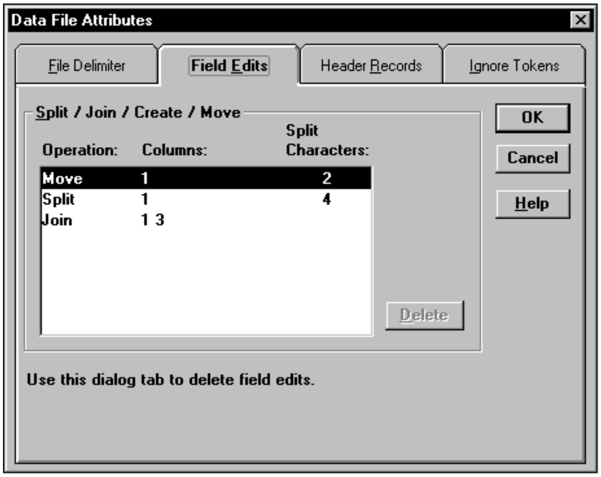
How Does Essbase Execute Operations in a Rules File?

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 20-9.
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 21-5.

   If you’re not sure in what order the field operations were defined, choose Options | Data File Attributes and click the **Field Edits** tab. The **Data File Attributes** dialog box appears, listing all the field operations.

The rules file in Figure 19-7, for example, contains move, split, and join operations.



Operation:	Columns:	Split Characters:
Move	1	2
Split	1	4
Join	1 3	

Use this dialog tab to delete field edits.

Figure 19-7, Field Operations

3. Essbase applies all attributes for each field, applying all of the attributes to field1 before proceeding to field2. Essbase applies field attributes 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, choose Field | Attributes and click the **Global Attributes** tab. The **Field Attributes** 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.

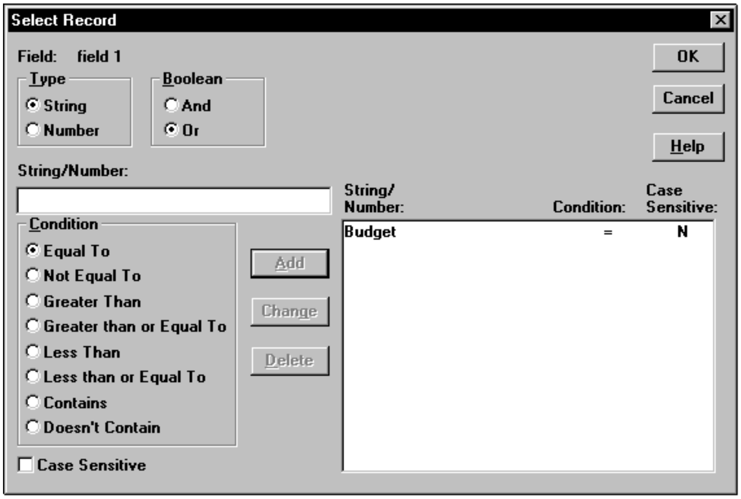
    For more information, see Chapter 21, Manipulating Fields Using a Rules File.

4. If you chose 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 20-10.

5. Essbase performs select or reject criteria in the order they are defined in the rules file. Select and reject criteria load or don't load individual records in the data source based on criteria specified in the rules file.

   If you're not sure in what order the select or reject criteria were defined, choose Record | Select or Record | Reject. The **Select Record** or **Reject Record** dialog box appears, listing all the select or reject operations.

   The rules file in Figure 19-8, for example, contains a selection criterion.



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

Figure 19-8, Selecting Records

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 20-10.

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 19-9 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.

```
Jan, Sales, Actual
Ohio       Cola          25
           "Root Beer"   50
           "Diet Cola"   19
```

Figure 19-9, Valid Missing Members

Essbase stops the data load if no prior records contain a value for the missing member field. If you tried to load Figure 19-10 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 19-9) is not specified.

```
Jan, Sales, Actual
           Cola          25
           "Root Beer"   50
           "Diet Cola"   19
```

Figure 19-10, Invalid Missing Members

For information on restarting the load, see “Loading the Error Log File” on page 23-4.

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.

```
Jan     Cola     East     Sales    Actual   100
Feb     Cola     East     Sales    Actual   200
```

Figure 19-11, All Dimensions Specified

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 19-12, for example, quotes New York, because it has a space in it:

```
Cola    Jan    "New York"Actual   Sales    50
Cola    Jan    Ohio      Actual   Sales    78
```

Figure 19-12, Quoted Member Names

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 19-13 into the Sample Basic database, Essbase rejects the record containing Ginger Ale because it's 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.

```
Jan, Sales, Actual
Ohio    Cola          2
        "Root Beer"   12
        "Ginger Ale"  15
        "Cream Soda"  11
```

Figure 19-13, Unknown Members

Note: Instead of rejecting the record, you can add the new members encountered to the database using the dimension build feature. See Chapter 12, Introducing Dynamic Dimension Building.

For information on restarting the load, see "Loading the Error Log File" on page 23-4.

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 19-2, 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 19-14 into the Sample Basic database, for example, identify the last column as a data field.

```
Jan     Cola     East     Sales    Actual   100
Feb     Cola     East     Sales    Actual   200
```

Figure 19-14, Setting Data Fields

To identify a column as a data field, see "Defining a Column as a Data Field" on page 21-19.

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.

```
             Jan, Actual
Cola         East      Sales     100
"Root Beer"  East      Sales     200
```

Figure 19-15, Assigning Data Fields

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 19-14, 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 21-14.

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 dimensions: 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 21-6.

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 19-16, is invalid, because there is no value under Apr. To fix the file, insert #MISSING or #MI into the missing field.

```
Actual Ohio Sales Cola
Jan     Feb      Mar      Apr
10      15       20
```

Figure 19-16, Missing Fields

Figure 19-17 is valid because #MI was inserted to replace the missing field.

```
Actual Ohio Sales Cola
Jan     Feb      Mar      Apr
10      15       20       #MI
```

Figure 19-17, Valid Missing Fields

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 19-18 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.

```
East,,Cola,Actual,Jan,Sales,10
East,Cola,Actual,Feb,Sales,21
East,Cola,Actual,Mar,Sales,30
```

Figure 19-18, File Delimiters

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 19-8.

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 21-14

Use the LOADDATA command in ESSCMD to load data free form. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, 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 19-19, for example, Jan and Feb form a range in the Year dimension and Sales and Profit form a range in the Measures dimension.

```
Texas               Sales             Profit
                    Jan      Feb      Jan      Feb
Actual  "100-10"    98       89       26       19
        "100-20"    87       78       23       32
```

Figure 19-19, Multiple Ranges of Member Names

In Figure 19-19, 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 19-20, 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.

```
Texas Sales
                        Jan      Feb      Mar
Actual     "100-10"     98       89       58
           "100-20"     87       78       115
```

Figure 19-20, Ranges of Member Names

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 19-21, 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.

```
Cola Actual East
        Jan      Feb
Sales   100      120      10
COGS    30       34       32
```

Figure 19-21, Extra Data Values

For information on restarting the load, see “Loading the Error Log File” on page 23-4.

Duplicate Members in a Range

If the same member appears more than once in a range, Essbase ignores the duplicate members.

The file in Figure 19-22 contains duplicate members.

```
Cola East
        Actual   Budget   Actual   Budget
        Sales    Sales    COGS     COGS
Jan     108      110      49       50
Feb     102      120      57       60
```

Figure 19-22, Duplicate Members in a Range

Essbase ignores the duplicate members. The members that Essbase ignores have a line through them in the following example:

```
Cola East
        Actual   Budget   ~~Actual~~ ~~Budget~~
        Sales    ~~Sales~~ COGS     ~~COGS~~
Jan     108      110      49       50
Feb     102      120      57       60
```

Figure 19-23, Ignored Duplicate Members

Because Essbase ignores duplicate members, it interprets the file as follows:

```
Cola East
                 Actual            Budget
                 Sales    COGS     Sales    COGS
Jan              108      110      49       50
Feb              102      120      57       60
```

Figure 19-24, How Essbase Interprets the File in Figure 19-22

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 19-24, 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 19-25, for example, each dimension column has one column of members under it. For example, Product has 100-10 under it.

```
Product  MeasuresMarket  Year   Scenario *data*
"100-10" Sales    Texas   Jan    Actual   112
"100-10" Sales    Ohio    Jan    Actual   145
```

Figure 19-25, Symmetric Columns

The columns in the following file are also symmetric, because Jan and Feb have the same number of members under them:

```
                              Jan            Feb
                        Actual   Budget  Actual  Budget
"100-10" Sales    Texas 112      110     243     215
"100-10" Sales    Ohio  145      120     81      102
```

Figure 19-26, Groups of Symmetric Columns

Asymmetric Columns

Columns can also be asymmetric. In Figure 19-27, 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):

```
                 Jan     Jan     Feb
                 Actual  Budget  Budget
"100-10" Sales   Texas   112     110     243
"100-10" Sales   Ohio    145     120     81
```

Figure 19-27, Valid Groups of Asymmetric Columns

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 19-28, for example, is not valid because the column labels are incomplete. The Jan label must appear over both the Actual and Budget columns.

```
                        Jan             Feb
                        Actual  Budget  Budget
"100-10" Sales  Texas   112     110     243
"100-10" Sales  Ohio    145     120     81
```

Figure 19-28, Invalid Asymmetric Columns

This file in Figure 19-29 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.

```
                        Jan     Jan     Feb
                        Actual  Budget  Budget
"100-10" Sales  Texas   112     110     243
"100-10" Sales  Ohio    145     120     81
```

Figure 19-29, Valid Asymmetric Columns

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 multi-user environment, Essbase protects your data using the security system described in Chapter 16, Managing Security at Global and User Levels.

- **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.
- **Multi-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 41, Ensuring Data Integrity, for more information.

      For information on how to see which user has a lock on a particular block, see Chapter 16, Managing Security at Global and User Levels.
    - 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 41, Ensuring Data Integrity, for more information.

Chapter 20 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 Hyperion Essbase Application Manager. For information about performing operations on fields within a record, see Chapter 21, Manipulating Fields Using a Rules File. For information about loading data using rules files, including prerequisites, see Chapter 22, Performing a Data Load.

This chapter contains the following sections:

- “Selecting the Data Source” on page 20-1
- “Setting File Delimiters” on page 20-9
- “Using Header Information” on page 20-10
- “Selecting Records” on page 20-13
- “Rejecting Records” on page 20-14
- “Defining Multiple Select and Reject Criteria” on page 20-16
- “Setting the Records Displayed” on page 20-17
- “Validating and Saving Data Load Rules” on page 20-18

Selecting the Data Source

This section describes how to select your data source, including:

- “Connecting to the Server” on page 20-2
- “Viewing the Application and Database” on page 20-2
- “Opening the Data Prep Editor” on page 20-3
- “Opening a Data Source” on page 20-4

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. You must open the Data Prep Editor and view the rules file. After you connect to the server, the Application Desktop window appears.

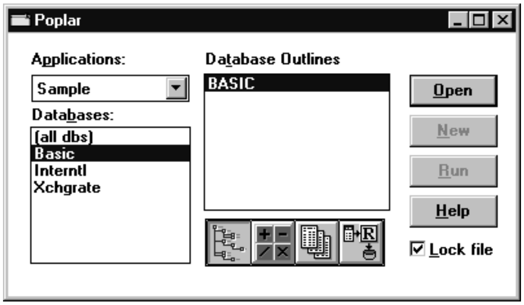


Figure 20-1, Application Desktop Window

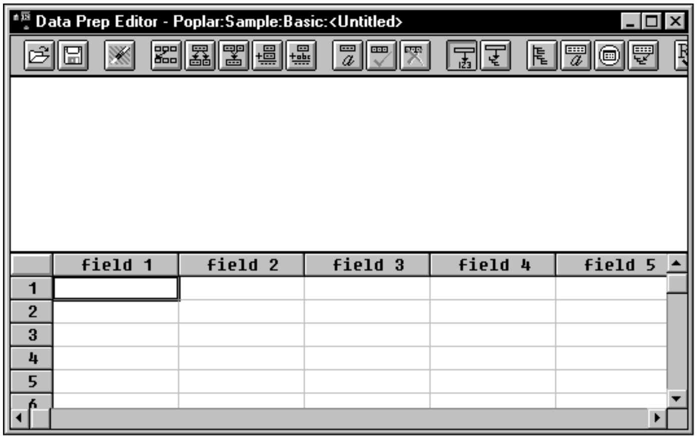
1. Select the application to view from the **Applications** list box; for example, the Sample application.
2. Select the database from the **Databases** list box; for example, the Basic database.

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.



Data Prep Editor - Poplar:Sample:Basic:<Untitled>

	field 1	field 2	field 3	field 4	field 5
1					
2					
3					
4					
5					
6					

Figure 20-2, Data Prep Editor

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.

- Choose 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.
- Choose 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 12, 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, choose 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, choose View | Gridlines.
- To hide the raw data, choose 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, choose View | Raw Data.
- To hide the toolbar, choose 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, choose View | Toolbar.

Opening a Data Source

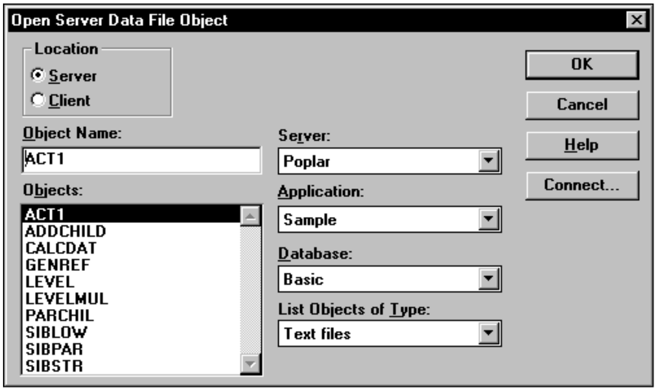
After you open the Data Prep Editor, you can open data sources, such as text files, spreadsheet files, and SQL data sources, to create rules files for. This section describes how to open the following kinds of data sources:

- “Opening a Text File” on page 20-5
- “Opening a Spreadsheet File” on page 20-6
- “Opening an SQL Data Source” on page 20-7

Opening a Text File

After you open the Data Prep Editor, you can open text files in it. To open a text file:

1. Choose 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.



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

Figure 20-3, Open Server Data File Object Dialog Box

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 don't, 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.

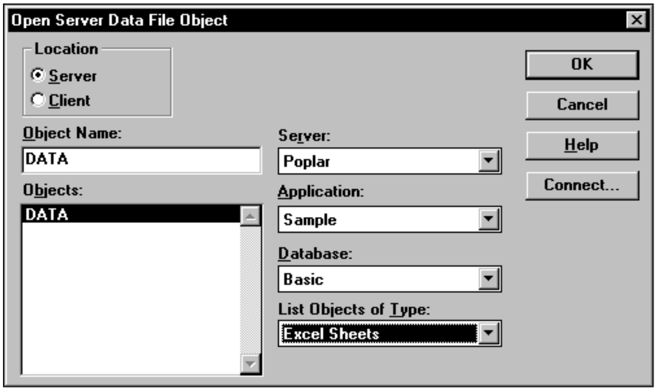
Note: The `\ESSBASE\APP` and `\ESSBASE\CLIENT` are the default directories specified during installation. You may have set these directories differently.

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. Choose File | Open Data File to open the **Open Server Data File Object** dialog box.



Location: ● Server ○ Client

Object Name: DATA

Objects: DATA

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Excel Sheets

Figure 20-4, Open Server Data File Object Dialog Box: Spreadsheet Files

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.

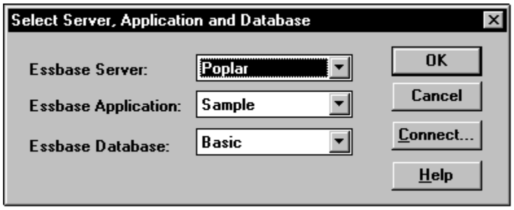
Opening an SQL Data Source

When you open the Data Prep Editor, you can open an SQL data source if you've licensed Hyperion Essbase SQL Interface. The *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. Choose 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.



Select Server, Application and Database

Essbase Server: Poplar

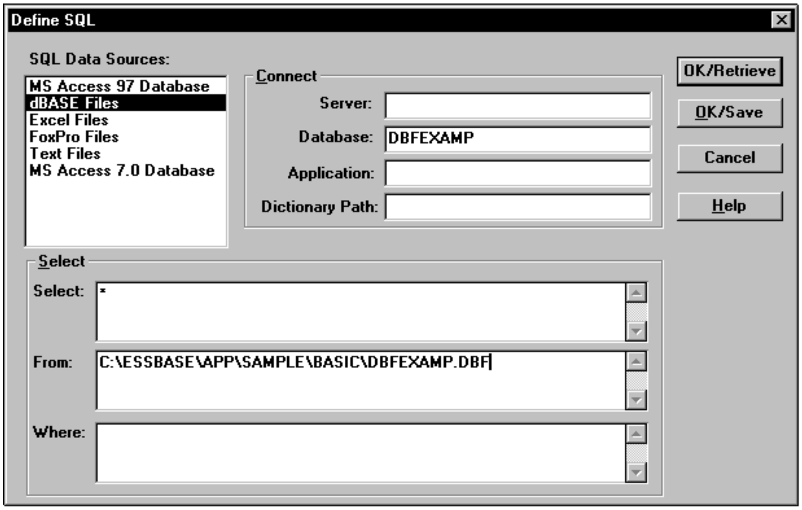
Essbase Application: Sample

Essbase Database: Basic

Figure 20-5, Select Server, Application and Database Dialog Box

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:



Define SQL

SQL Data Sources:
MS Access 97 Database
dBASE Files
Excel Files
FoxPro Files
Text Files
MS Access 7.0 Database

Connect
Server:
Database: DBFEXAMP
Application:
Dictionary Path:

Select
Select: *
From: C:\ESSBASE\APP\SAMPLE\BASIC\DBFEXAMP.DBF
Where:

Figure 20-6, Define SQL Dialog Box

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.

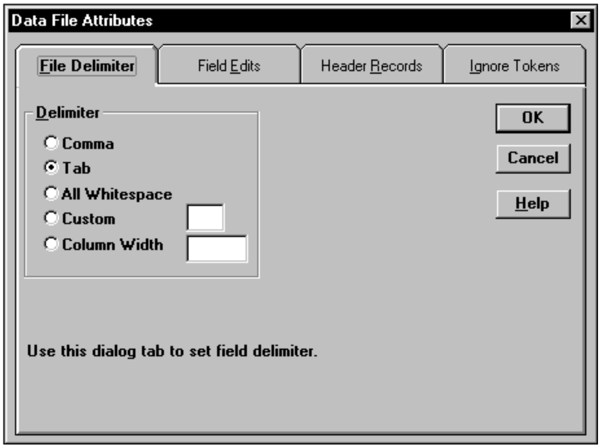
Setting File Delimiters

File delimiters are the character(s) 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. Choose Options | Data File Attributes or click the **Data File Attributes** button, , to open the **Data File Attributes** dialog box. Click the **File Delimiter** tab.



Delimiter

○ Comma
● Tab
○ All Whitespace
○ Custom
○ Column Width

Use this dialog tab to set field delimiter.

Figure 20-7, File Delimiter Page

2. Select the type of delimiter to use in your file:
   - **Comma**.
   - **Tab**.
   - **All Whitespace**.
   - **Custom:** enter the custom delimiter in the text box. Custom delimiters must be two characters or less.
   - **Column Width:** enter the width of the column in the text box. 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.

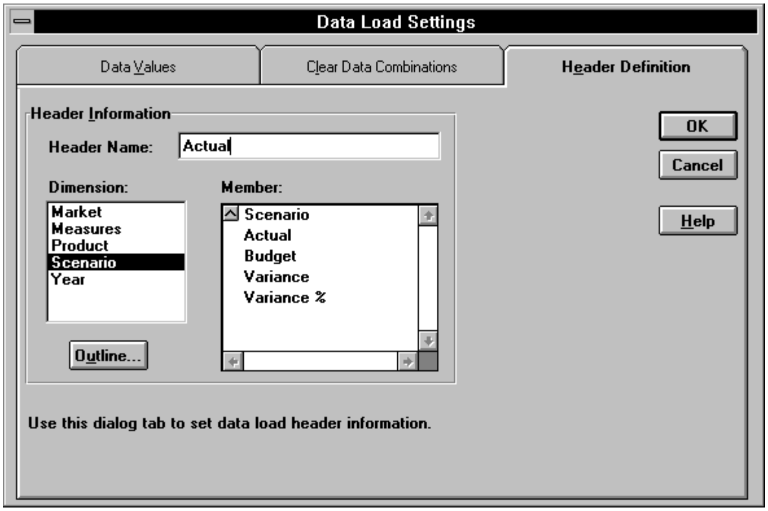
You can create header records using the following methods:

- 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. Choose Options | Data Load Settings or click the **Global Data Load Attributes** button, , to open the **Data Load Settings** dialog box. Click the **Header Definition** tab.



Header Name: Actual

Use this dialog tab to set data load header information.

Figure 20-8, Header Definition Page

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.

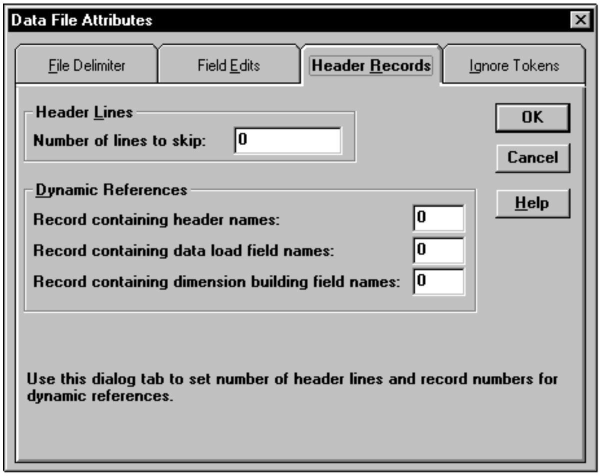
When you add one or more headers to the data source, you must also specify the location of the headers in the data source using a dynamic reference. *Dynamic references* are set in the rules file and tell Essbase to read the header information as a header record and not a data record. When setting dynamic 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 your data source using a text editor or spreadsheet. Member combinations must be separated by a comma.

2. Choose Options | Data File Attributes or click the **Data File Attributes** button, , to open the **Data File Attributes** dialog box. Click the **Header Records** tab.



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.

Figure 20-9, Header Records Page

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 12, Introducing Dynamic Dimension Building, for more information.

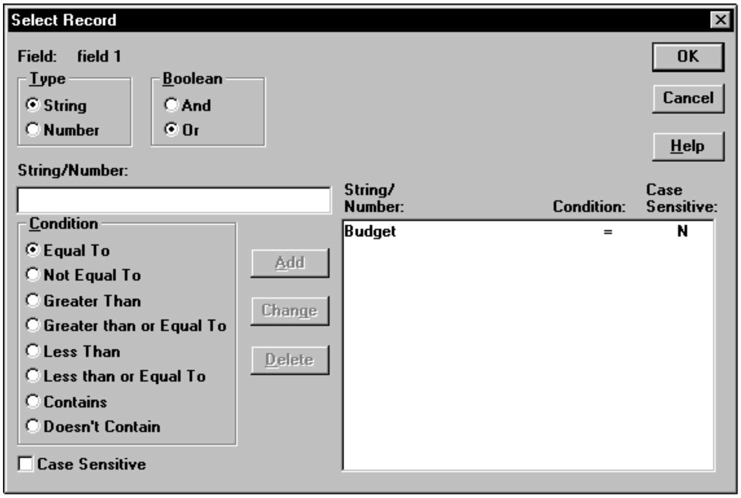
7. Click OK.

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. Choose Record | Select or click the **Record Selection** button, ✓, to open the **Select Record** dialog box.



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

Figure 20-10, Select Record Dialog Box

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 checking **Case Sensitive**.
4. Enter the string or number upon which the criterion is based in the **String/Number** text box; for example, `Budget`.
5. Choose 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 20-16.

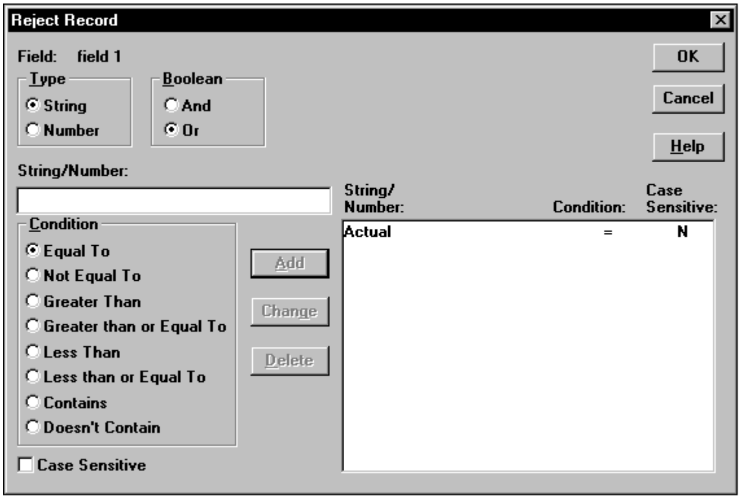
8. Click OK.

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. Choose Record | Reject or click the **Record Rejection** button, ☒, to open the **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

Figure 20-11, Reject Record Dialog Box

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 checking 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. Choose 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 20-16.

8. Click OK.

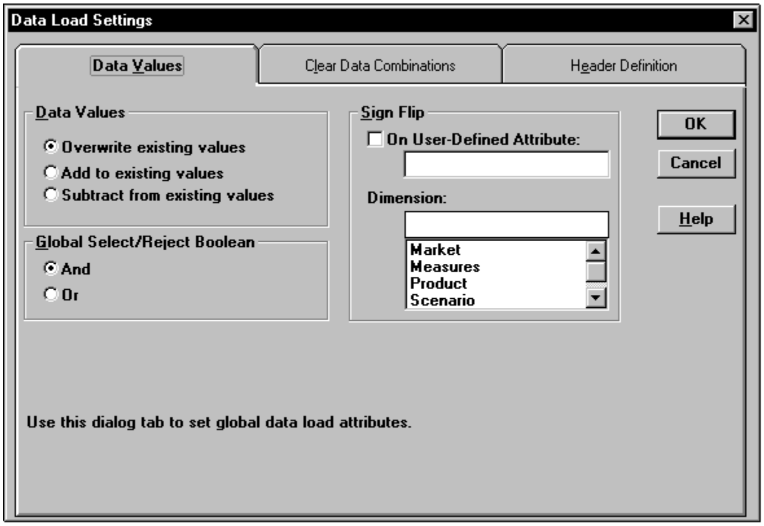
Defining Multiple Select and Reject Criteria

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. Choose Options | Data Load Settings or click the **Global Data Load Attributes** button, , to open the **Data Load Settings** dialog box. Click the **Data Values** tab.



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.

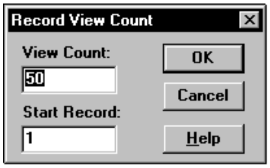
Figure 20-12, Data Values Page: Global Select/Reject Boolean

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. Choose Record | Record View Count to open the **Record View Count** dialog box.



View Count: 50

Start Record: 1

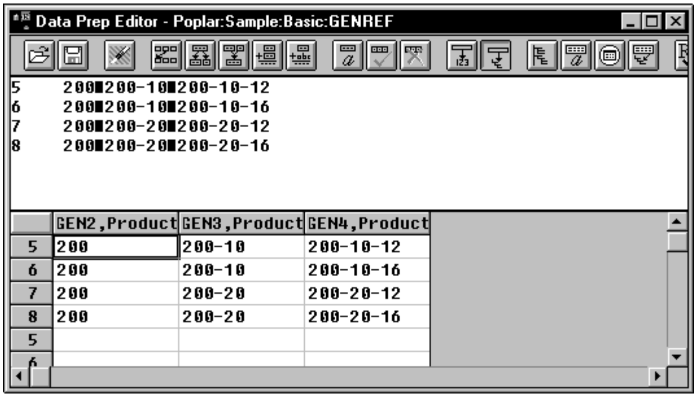
Figure 20-13, Record View Count Dialog Box

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.

Notes:

- 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 20-14, for example, displays records 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			

Figure 20-14, Displaying Record 5 through 8

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.

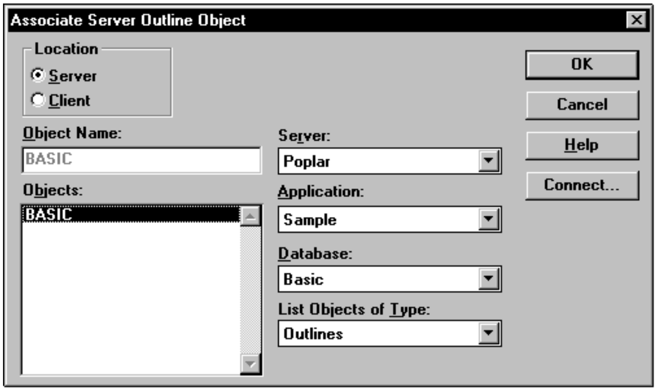
Before you validate a data load rules file, you must associate the rules file with an outline. This is usually the outline of the database into which to load the data. The rules file is not permanently associated with that outline, and you can associate it with any other outline in the future. See “Associating a Rules File with an Outline” on page 20-18.

Rules files are validated to make sure the member and dimension mapping defined in the rules file maps to the outline. See Chapter 21, 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. Choose Options | Associate Outline or click the **Outline** button, , to open the **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

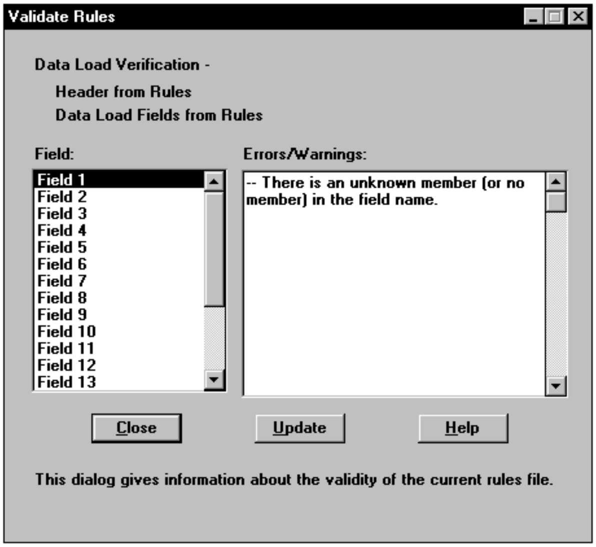
Figure 20-15, Associate Server Outline Object Dialog Box

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're building dimensions by altering the data source (using dynamic references), make sure the data source is open as well.

1. Choose 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.



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.

Figure 20-16, Validate Rules Dialog Box

2. If the rules file validates with no problems, you can use it to load data. See Chapter 19, 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 20-16, for example, **Field 1** was 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?

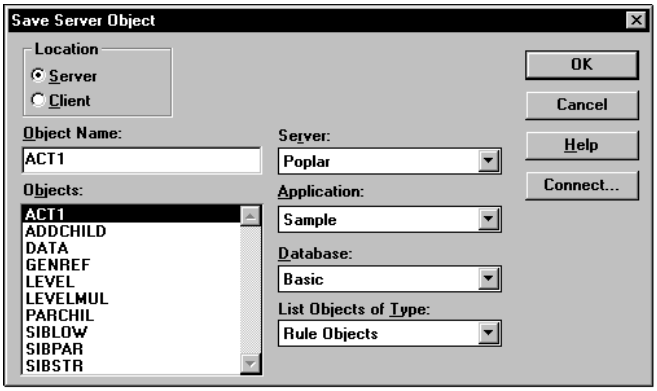
Note: Chapter 23, Debugging and Optimizing Data Loads, contains a complete list of data load rules file validation errors and possible causes.

5. Validate the file again. Return to step 1.

Saving Rules Files

To save a rules file:

1. Choose File | Save or click the **Save** button, 💾. If you haven't saved the rules file before, the **Save Server Object** dialog box appears.



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

Figure 20-17, Save Server Object Dialog Box

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, all rules file names 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, choose 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 21 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 Hyperion Essbase 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 20, Setting up a Rules File to Manipulate Records.

For more information about loading data using rules files, including prerequisites, see Chapter 22, Performing a Data Load.

This chapter contains the following sections:

Selecting Multiple Fields

You can select multiple fields and then set the attributes for them. Essbase grays out any menu items and controls you can't 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 don't 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.

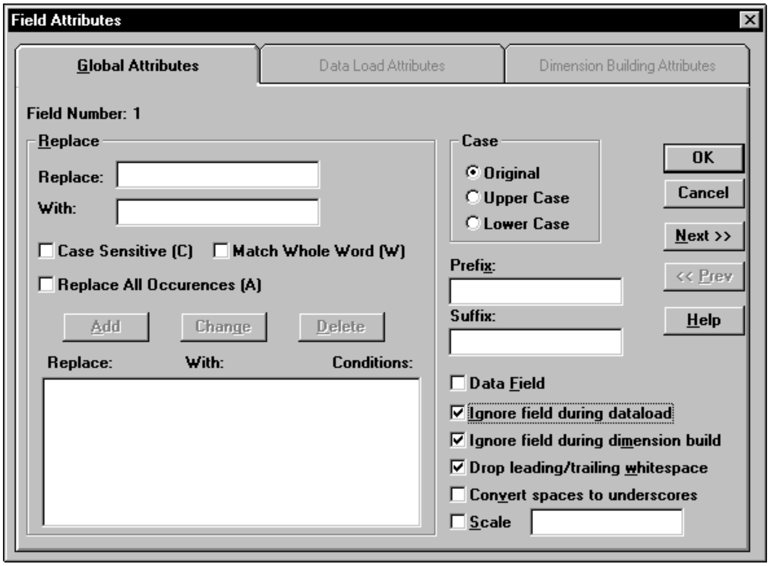
You can also ignore individual fields in your data source that match a string called a *token*. When you ignore fields based on string values, these fields are ignored everywhere they appear in the data source, not just in a single column.

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 column(s) to ignore.
2. Choose Field | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Global Attributes** tab.



Field Number: 1

Case: ● Original ○ Upper Case ○ Lower Case

□ Case Sensitive (C) □ Match Whole Word (W)

□ Replace All Occurences (A)

□ Data Field

☑ Ignore field during dataload

☑ Ignore field during dimension build

☑ Drop leading/trailing whitespace

□ Convert spaces to underscores

□ Scale

Figure 21-1, Global Attributes Page: Ignore Check Boxes

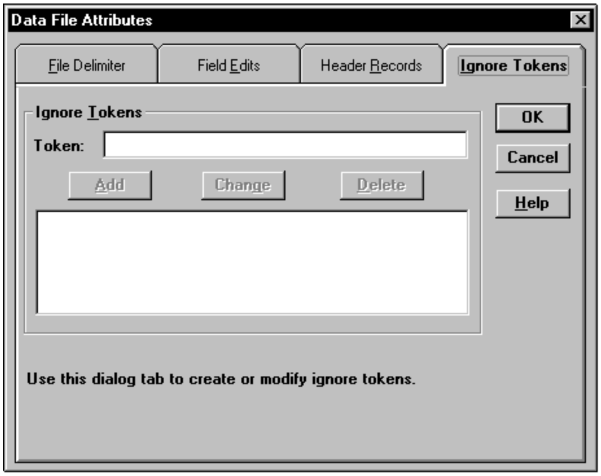
3. Check **Ignore field during data load** to ignore the field during a data load. Check **Ignore field during dimension build** to ignore the field during a dimension build. To ignore the field for both, check both boxes.
4. Click OK. The values in that field now appear grayed out in the Data Prep Editor 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 Attributes to bring up the **Data File Attributes** dialog box. Click the **Ignore Tokens** tab. Token is another word for string.



Use this dialog tab to create or modify ignore tokens.

Figure 21-2, Ignore Tokens Page

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 21-5
- “Joining Fields” on page 21-6
- “Creating a New Field While Leaving Existing Fields Intact” on page 21-7
- “Splitting Fields” on page 21-8
- “Creating Additional Text Fields” on page 21-9
- “Undoing Field Ordering” on page 21-10

Note: Whenever you want to undo a single operation, choose Edit | Undo. To undo multiple field operations, see “Undoing Field Ordering” on page 21-10.

Moving Fields

To move one or more fields:

1. Select the field to move. If you don’t 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.

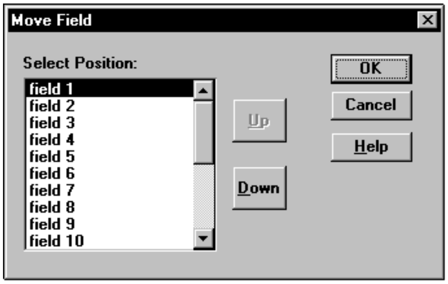


Figure 21-3, Move Field Dialog Box

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 21-10.

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 don't know how to move fields, see "Moving Fields" on page 21-5.

2. Select the fields to join, for example `100` and `-10`. If you don't know how to select multiple fields, see "Selecting Multiple Fields" on page 21-2.

3. Choose Field | Join or click the **Join** button, , to open the **Join Fields** dialog box.

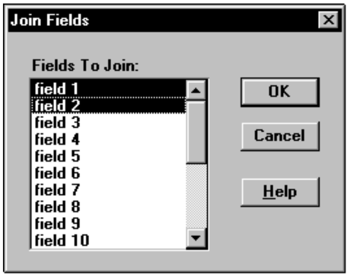


Figure 21-4, Join Fields Dialog Box

4. If you didn't 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 21-10.

Creating a New Field While Leaving Existing Fields Intact

You can join two or more fields by sticking the joined fields into a brand new field. This leaves the existing fields intact. 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 don’t know how to select multiple fields, see “Selecting Multiple Fields” on page 21-2.

2. Choose Field | Create Using Join or click the **Create Using Join** button, , to open the **Create Field Using Join** dialog box.

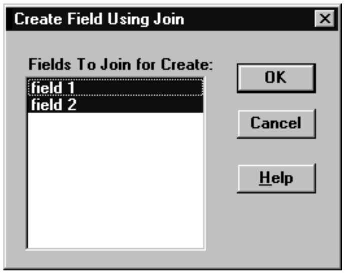
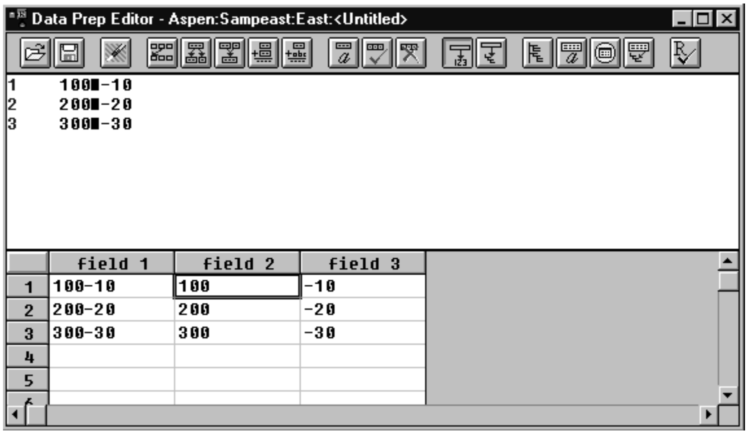


Figure 21-5, Create Field Using Join Dialog Box

3. If you didn’t select the fields to join, select the fields in the **Create Field Using Join** dialog box.

4. Click OK. The new field appears to the left of the first field containing joined information. For example, in Figure 21-6, a new 100-10 field appears to the left of the existing 100 and -10 fields.



Data Prep Editor - Aspen:Sampeast:East:<Untitled>

```
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			

Figure 21-6, New 100-10 Field

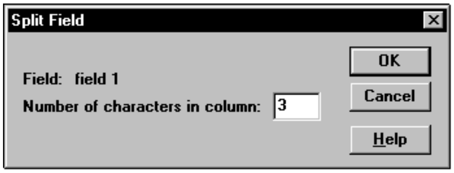
Splitting Fields

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 21-3. 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.



Split Field

Field: field 1

Number of characters in column: 3

Figure 21-7, Split Field Dialog Box

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 appears to the left of the original field.

Note: To undo a split, see "Undoing Field Ordering" on page 21-10.

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.

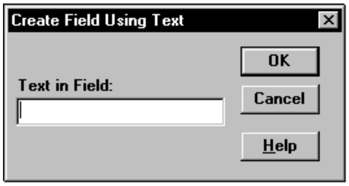


Figure 21-8, Create Field Using Text Dialog Box

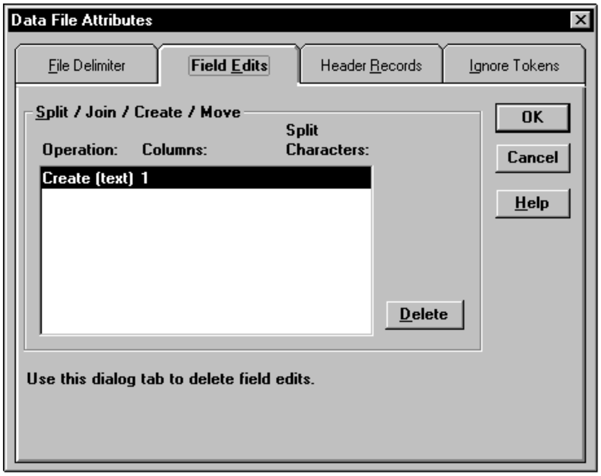
3. Enter the text to put into the new field, for example, a hyphen -.
4. Click OK. The new field appears to the left of the selected field.

Note: To undo a field you created using text, see "Undoing Field Ordering" on page 21-10.

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've 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 don't know how to select multiple fields, see "Selecting Multiple Fields" on page 21-2.
2. Choose Options | Data File Attributes or click the **Data File Attributes** button, , to open the **Data File Attributes** dialog box. Click the **Field Edits** tab.



Create (text) 1

Use this dialog tab to delete field edits.

Figure 21-9, Field Edits Page

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 21-11.
- Replace strings, or translate, the fields in the data source to match members in the database. See “Replacing Text Strings” on page 21-13.

Naming Fields

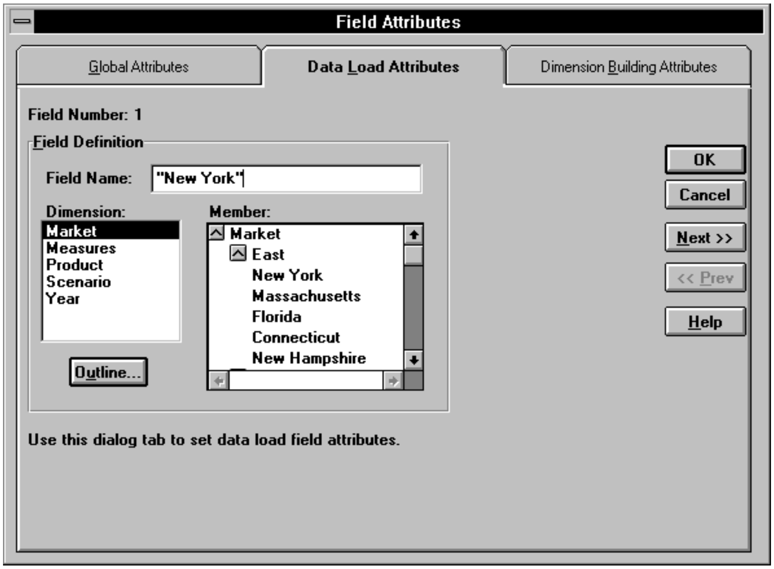
Use a rules file to name data source fields to match Essbase dimension names during a data load. The data source is not changed.

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 don’t have to perform any field mapping.

To name a field:

1. Select the field to name in the Data Prep Editor.

2. Choose Field | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Data Load Attributes** tab.



Field Attributes

Field Number: 1

Field Definition

Field Name: "New York"

Dimension:
Market
Measures
Product
Scenario
Year

Member:
Market
East
New York
Massachusetts
Florida
Connecticut
New Hampshire

Use this dialog tab to set data load field attributes.

Figure 21-10, Data Load Attributes Page

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 quotes around the member name. If you click the member name in the **Member** list box, Essbase automatically puts quotes 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.

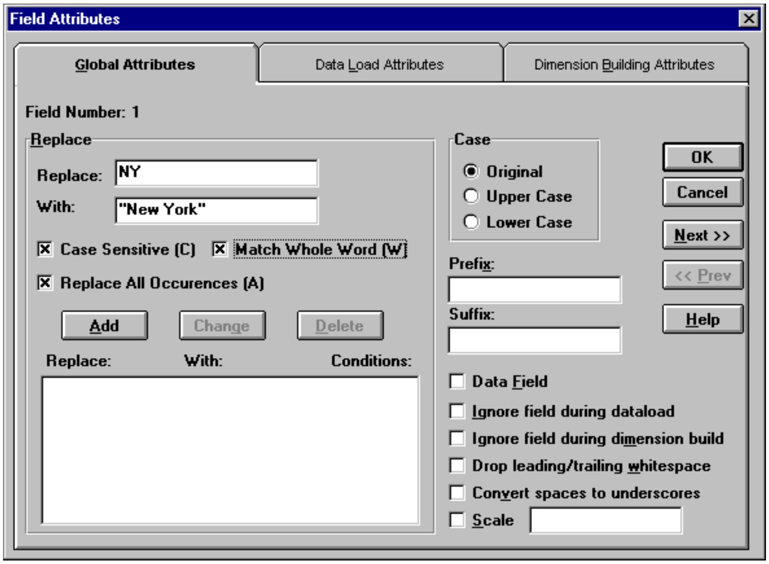
5. When you're finished, click OK.

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 | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Global Attributes** tab.



Field Number: 1

Replace

Replace: NY

With: "New York"

☒ Case Sensitive (C) ☒ Match Whole Word (W)

☒ Replace All Occurences (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

Figure 21-11, Global Attributes Page: Replace Box

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 21-14.

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:

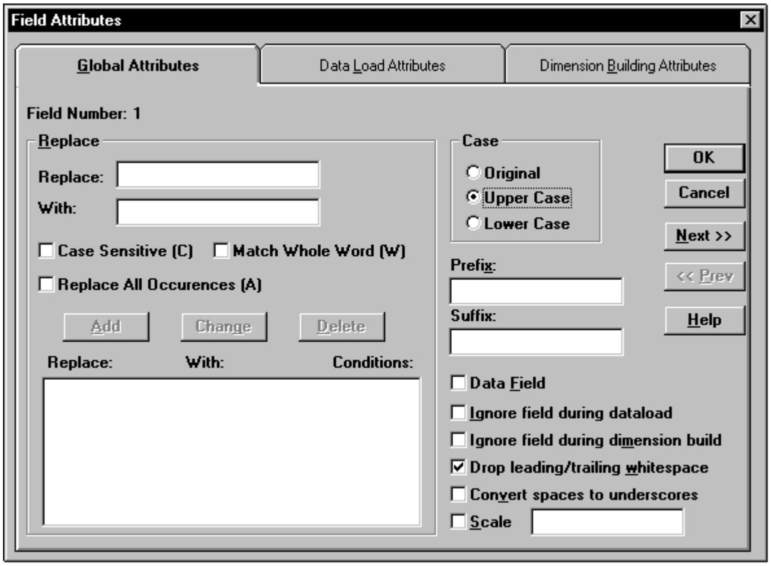
1. Select the column containing the empty fields you want to replace.
2. Choose Field | Create Using Text or click the **Create Using Text** button, .
3. Enter the text to put in the new field. Enter a dummy string that's not in the selected column, such as `temp`. Click OK.
4. Join the new field with 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 | Attributes or click the **Field Attributes** button, . Click the **Global Attributes** 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 | Attributes or click the **Field Attributes** button, . Click the **Global Attributes** 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 | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Global Attributes** tab.



Field Number: 1

Replace:

With:

☐ Case Sensitive (C) ☐ Match Whole Word (W)

☐ Replace All Occurences (A)

Replace: With: Conditions:

○ 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

Figure 21-12, Global Attributes Page: Case Box

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.

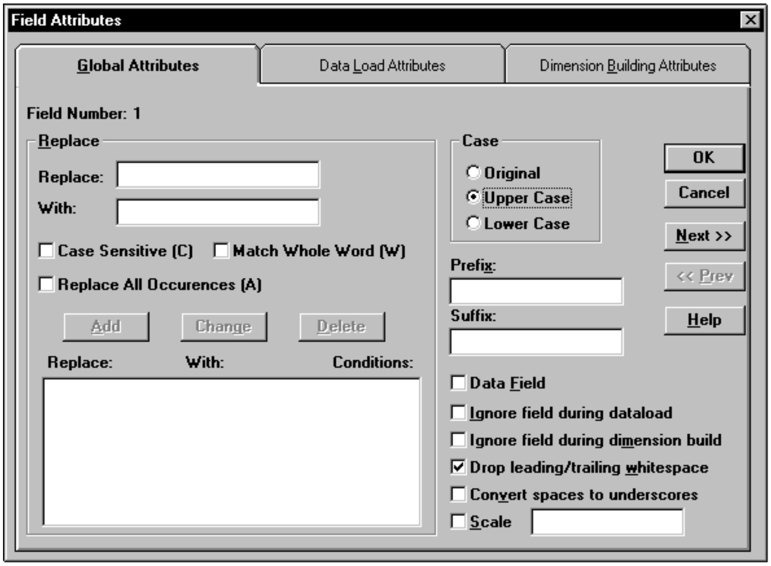
Dropping Leading/Trailing White Space

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 don't know how to select multiple fields, see "Selecting Multiple Fields" on page 21-2.
2. Choose Field | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Global Attributes** tab.



Field Attributes

Field Number: 1

Case: ○ Original ● Upper Case ○ Lower Case

□ Case Sensitive (C) □ Match Whole Word (W)

□ Replace All Occurences (A)

□ Data Field

□ Ignore field during dataload

□ Ignore field during dimension build

☑ Drop leading/trailing whitespace

□ Convert spaces to underscores

□ Scale

Figure 21-13, Global Attributes Page: Drop Leading/Trailing Whitespace Check Box

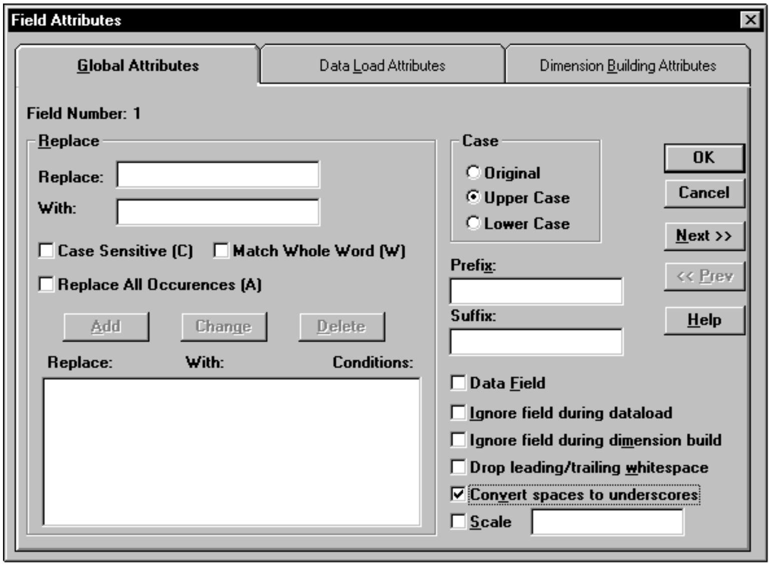
3. By default, the **Drop leading/trailing whitespace** box is checked. If it's not already checked, check it.
4. Click OK.

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 don't know how to select multiple fields, see "Selecting Multiple Fields" on page 21-2.
2. Choose Field | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Global Attributes** tab.



Field Attributes

Global Attributes | Data Load Attributes | Dimension Building Attributes

Field Number: 1

Replace

Replace:

With:

□ Case Sensitive (C) □ Match Whole Word (W)

□ Replace All Occurences (A)

Add Change Delete

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

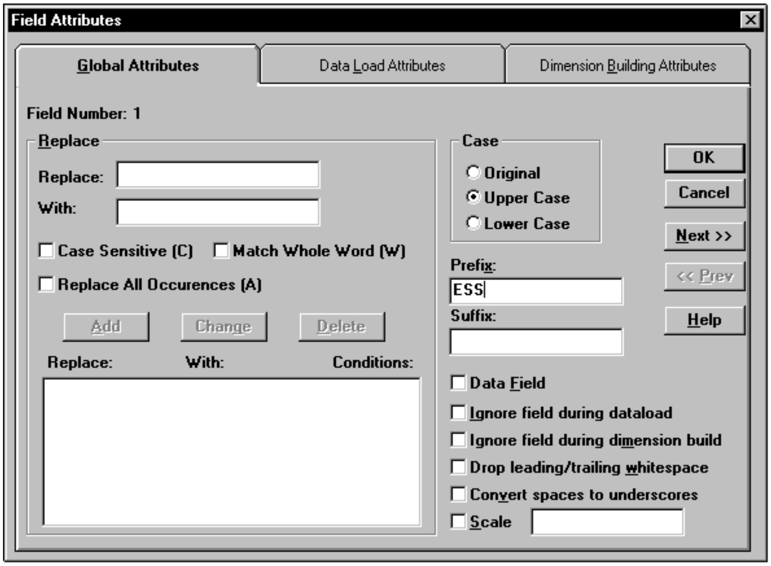
Figure 21-14, Global Attributes Page: Convert Spaces to Underscores Check Box

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 don’t know how to select multiple fields, see “Selecting Multiple Fields” on page 21-2.
2. Choose Field | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Global Attributes** tab.



Field Number: 1

Replace

Replace:

With:

□ Case Sensitive (C) □ Match Whole Word (W)

□ Replace All Occurences (A)

Case

○ Original

● Upper Case

○ Lower Case

Prefix: ESS

Suffix:

□ Data Field

□ Ignore field during dataload

□ Ignore field during dimension build

□ Drop leading/trailing whitespace

□ Convert spaces to underscores

□ Scale

Figure 21-15, Global Attributes Page: Prefix and Suffix Text Boxes

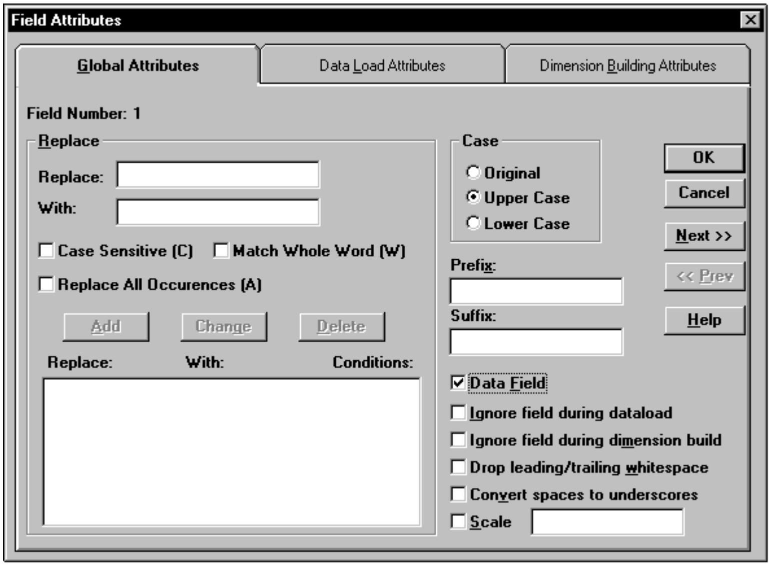
3. Enter the prefix or suffix in the **Prefix** or **Suffix** text box.
4. Click OK.

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 | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Global Attributes** tab.



Field Number: 1

Replace:

With:

☐ Case Sensitive (C) ☐ Match Whole Word (W)

☐ Replace All Occurences (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

Figure 21-16, Global Attributes Page: Data Field Check Box

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:

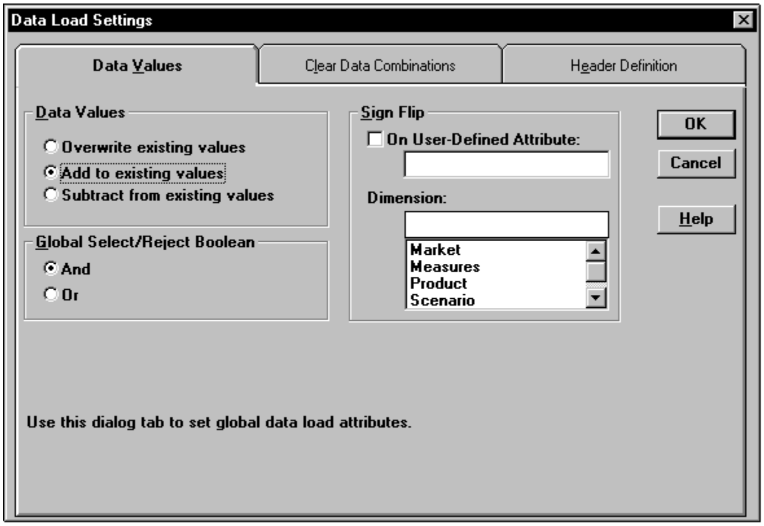
- "Adding to and Subtracting from Existing Values" on page 21-20
- "Clearing Existing Data Values" on page 21-21
- "Scaling Data Values" on page 21-23

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 Attributes** button, , to open the **Data Load Settings** dialog box. Click the **Data Values** tab.



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.

Figure 21-17, Data Values Page: Data Values Box

3. Choose **Add to existing values** to add the values or the **Subtract from existing values** to subtract the values.

Warning: 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 error log file. For more information, see Chapter 23, Debugging and Optimizing Data Loads.

To solve this problem, set the `Commit Rows` setting to 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 Chapter 40, Specifying Storage Manager Settings.

4. 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's assume that the Sales figures for January are calculated by adding together the values for each week in January. That means:

```
January Monthly 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:

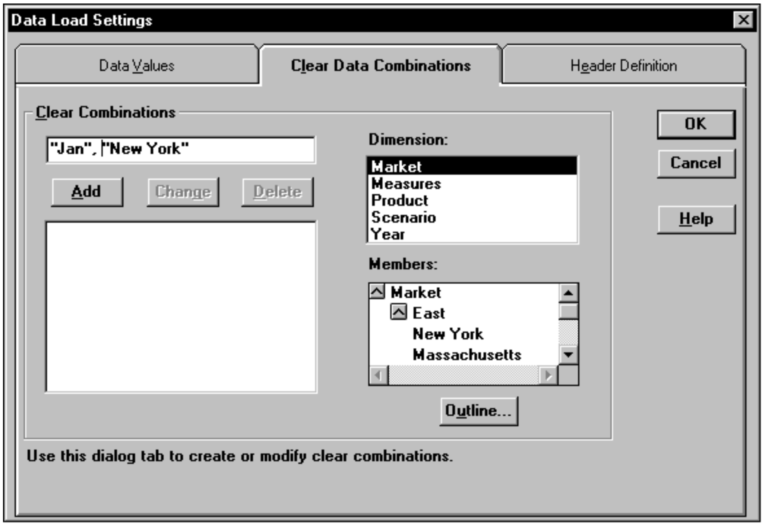
```
January Monthly Sales = Existing Value + Week 1 Sales + Week 2 Sales + Week 3
Sales + Week 4 Sales
```

You can also clear data from fields that aren't 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 Attributes** button, ▣, to bring up the **Data Load Settings** dialog box. Click the **Clear Data Combinations** tab.



Data Load Settings

Clear Combinations: "Jan", "New York"

Dimension: Market (selected), Measures, Product, Scenario, Year

Members: Market, East, New York, Massachusetts

Use this dialog tab to create or modify clear combinations.

Figure 21-18, Clear Data Combinations Page

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 20-18 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 `@ISDESCENDANTS(Massachusetts)`. For more information on Essbase functions, see the online *Technical Reference* in your `DOCS` directory.

Note: You must separate member combinations with a comma.

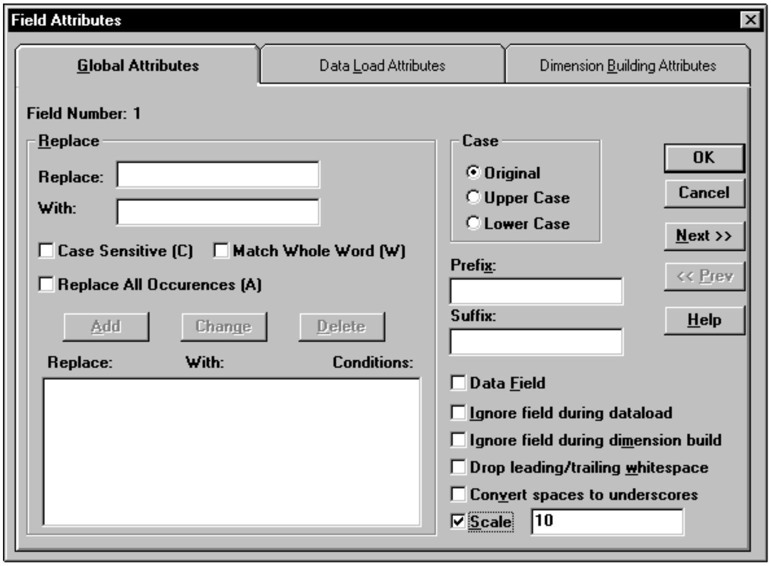
3. Click **Add** to add the member combination to the list. To change a member combination that’s in the list, select the item in the list and click **Change**. It appears in the **Clear Combinations** text box. To delete member combinations from the list, select them and click **Delete**.

Scaling Data Values

You can scale data values if the values in the data source aren't 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 | Attributes or click the **Field Attributes** button, , to open the **Field Attributes** dialog box. Click the **Global Attributes** tab.



Field Number: 1

Replace

Replace:

With:

□ Case Sensitive (C) □ Match Whole Word (W)

□ Replace All Occurences (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 10

Figure 21-19, Global Attributes Page: Scale Check Box

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.

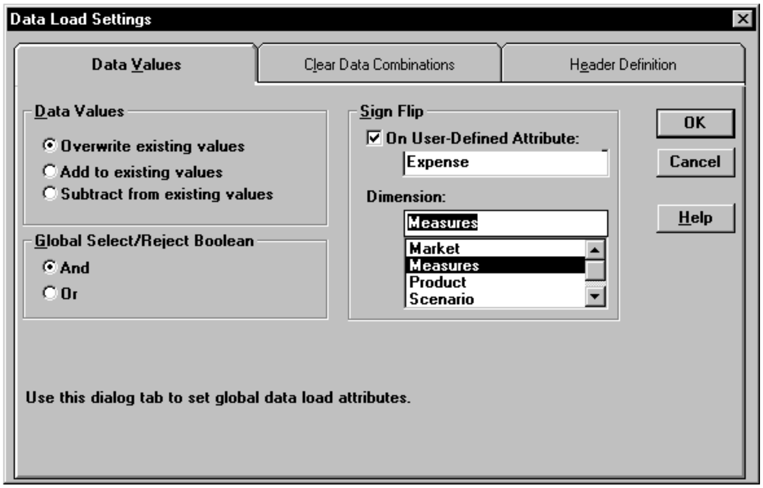
Note: The **Next** and **Prev** buttons only work if a single field is selected.

Flipping Field Signs

You can reverse or flip the value of a data field by flipping its sign. Sign flips are based on 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 user-defined attribute of Expense should change from a plus sign to a minus sign. You set user-defined attributes in the Outline Editor. See Chapter 8, 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 Attributes** button, , to open the **Data Load Settings** dialog box. Click the **Data Values** tab.



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: Expense
- Dimension: Measures
  - Market
  - Measures
  - Product
  - Scenario

Use this dialog tab to set global data load attributes.

Figure 21-20, Data Values Page

2. Check the **On User-Defined Attribute** box under **Sign Flip**.
3. Enter the user-defined attribute, for example, Expense. Use the Outline Editor to get a list of user-defined attributes. See Chapter 8, 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 20-18 for more information.
5. Click OK.

Chapter 22 Performing a Data Load

This chapter describes how to load data from external data sources to your Hyperion Essbase 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.

This chapter contains the following sections:

Use the LOADDATA or UPDATEFILE command in ESSCMD to load data without a rules file. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Prerequisites for Loading Data

To start loading data sources or building dimensions, you must have:

- A Hyperion Essbase database to load them into.
- 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
  - SQL data sources
- A correctly formatted data source, if you are not using a rules file. For information about free-form file formats, see "Rules for Free-Form Data Sources" on page 19-17.
- A valid rules file. For information about defining data load rules files, see Chapter 19, Introducing Data Loading.
- A dimension build rules file, if you are building dimensions dynamically. For more information about defining dimension build rules, see Chapter 12, Introducing Dynamic Dimension Building.

Note: You must use data load rules to load SQL data and to build dimensions and members dynamically.

Choosing the Data Sources Using the Application Manager

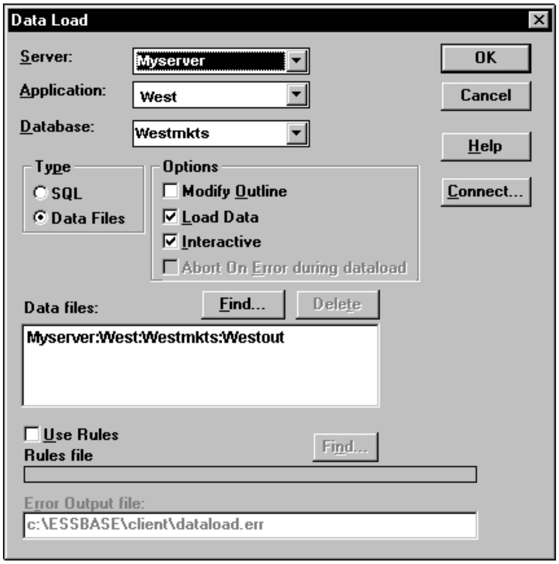
You can choose data sources using the Application Manager or Windows. For a list of valid data sources, see “Prerequisites for Loading Data” on page 22-2.

Make sure you are connected to the server before you specify the data sources.

Use the LOADDB command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

When you are connected to a server:

1. Click the Application Desktop window.
2. Choose Database | Load Data to specify how to load data or build dimensions. The **Data Load** dialog box appears.



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

Figure 22-1, Data Load Dialog Box

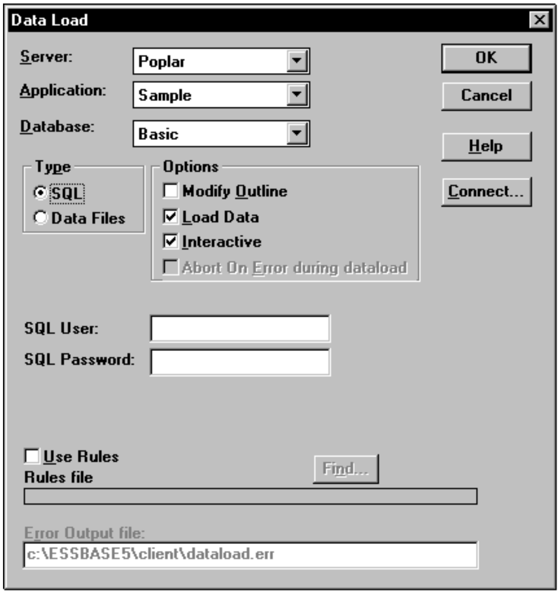
3. Click **Connect** to open the **Essbase System Login** dialog box. Enter the correct values and click OK. Now you are ready to start:
   - “Choosing SQL Data Sources” on page 22-4

- “Choosing Text or Spreadsheet Files” on page 22-5

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. Choose Database | Load Data. The **Data Load** dialog box appears.
4. To access SQL data, you must first connect to the SQL data source. Choose the **SQL** option. The **Data Load** dialog box changes to look like this:



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

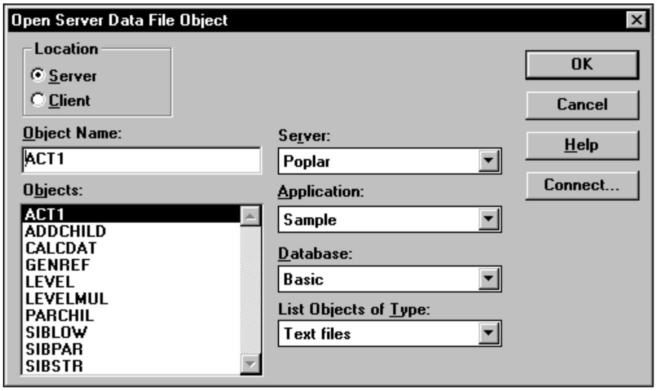
Figure 22-2, SQL Data Load Dialog Box

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 22-7 to finish, because you must use a rules file to load SQL data sources.

Choosing Text or Spreadsheet Files

To choose 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. Choose Database | Load Data. The **Data Load** dialog box appears.
4. Click the **Data File** option button if it's not already selected.
5. Click **Find** to select a text or spreadsheet file to load. The **Open Server Data File Object** dialog box appears:



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

Figure 22-3, Open Server Data File Object Dialog Box

6. Make sure the appropriate Essbase 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 22-3, for example, you could choose `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. Choose 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.

Notes:

- 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.

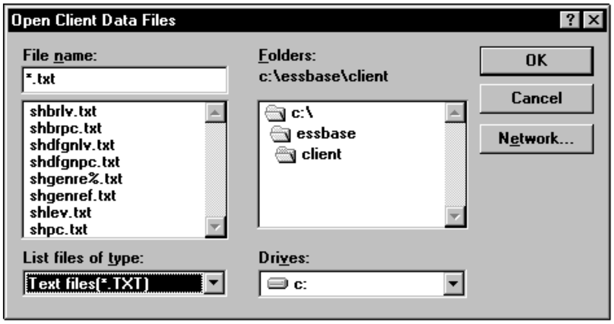


Figure 22-4, Open Client Data Files Dialog Box

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 choose the data sources using the Application Manager, the Windows File Manager or the Windows Explorer. For a list of valid data sources, see “Prerequisites for Loading Data” on page 22-2.

Make sure you are connected to the server before you choose the data sources.

To select a list of files to load:

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 source(s).

To select...	Do...
One file	Click the file name.
Several files	Hold the `Ctrl` key while clicking on the files.
A range of files	Click the first file, then hold down the `Shift` key and select the last file in the range.

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 appears. 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 21-14

Specifying How to Load Data or Build Dimensions

After you choose the data sources to load, you can specify how Essbase loads those data sources and whether to build dimensions dynamically using the **Data Load** dialog box. If you haven't chosen your data sources yet, see "Choosing the Data Sources Using the Application Manager" on page 22-3 or "Choosing the Data Sources Using Windows" on page 22-6.

You can set the following options:

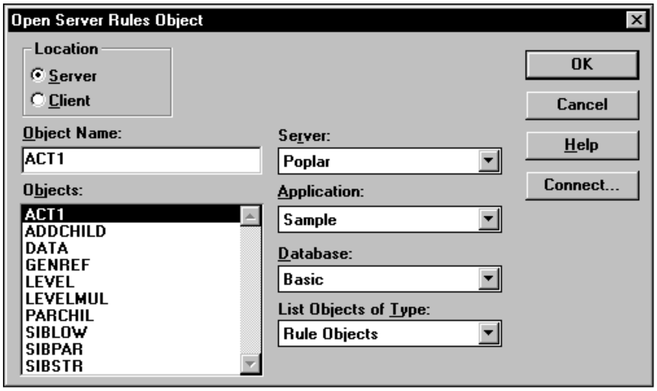
- Using a rules file with the data source
- Building dimensions by changing the outline

If you don't need to use a rules file or change the outline, skip to the "Setting the Error Log File" on page 22-10.

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 dynamically or load SQL data, you must use a rules file.

1. Choose your data sources. If you haven't chosen your data sources yet, see "Choosing the Data Sources Using the Application Manager" on page 22-3 or "Choosing the Data Sources Using Windows" on page 22-6.
2. From the **Data Load** dialog box, check **Use Rules**.
3. Click the **Find** button to open the **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

Figure 22-5, Open Server Rules Object Dialog Box

4. Make sure the appropriate Essbase 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.

Reasons to Stop	Reasons Not to Stop
To learn immediately that something is wrong with the data source.	To load as much data as possible and then look at errors in the error log.
To learn immediately that something is wrong with the rules file.	

8. To stop the data load if an error occurs, check **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.

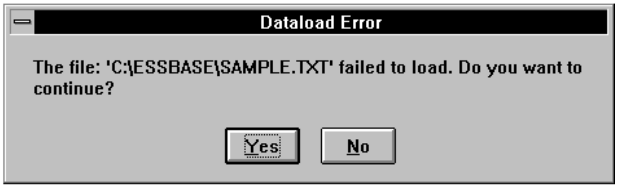
Warning: Modifying the outline using a data source and rules file restructures your database.

To change the outline:

1. Check **Modify Outline** in the **Data Load** dialog box. 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 12, Introducing Dynamic Dimension Building.

Note: If **Modify Outline** is not checked, Essbase rejects any records containing new members during the data load.

2. Click the **Interactive** check box to be prompted each time a data source fails. Essbase tells you which data source failed to load and asks you if you want to continue reading the remaining data sources.



Dataload Error

The file: 'C:\ESSBASE\SAMPLE.TXT' failed to load. Do you want to continue?

Figure 22-6, Dataload Error Dialog Box

3. To continue loading the remaining data sources, click Yes. To stop, click No. Any data sources loaded before you stop are in the database.
4. Check the error log to determine why the data source wouldn’t load or build dimensions. See “Finishing the Data Load or Dimension Build” on page 22-10 if you don’t 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 43, Performing Interactive and Batch Operations Using ESSCMD, for more information.

Setting the Error Log File

You can set a file to record errors during the data load or dimension build if you are using a rules file. An error file can be a valuable debugging tool if your data load or dimension build fails. By default, the error log is in the `\ESSBASE\CLIENT` directory and is named `DATALOAD.ERR`. To name the error log something else, enter the name in the **Error Output File** text box in the **Data Load** dialog box.

Warning: If the **Server Output File** text box is blank, Essbase doesn't capture errors.

For more information on errors during data loading or dimension building, see "Finishing the Data Load or Dimension Build" on page 22-10.

Now you can start loading data or building dimensions.

Starting the Data Load or Dimension Build

After you've set the data load options in the **Data Load** dialog box, you can start loading the data source(s) or building dimensions dynamically.

1. For data loads, make sure **Load Data** is checked and click OK.
2. For dimension builds, make sure **Modify Outline** is checked and click OK.

Note: You can load data or build dimensions at the same time or separately.

To speed up or optimize a data load, see the "Optimizing Data Loads" on page 23-6.

Finishing the Data Load or Dimension Build

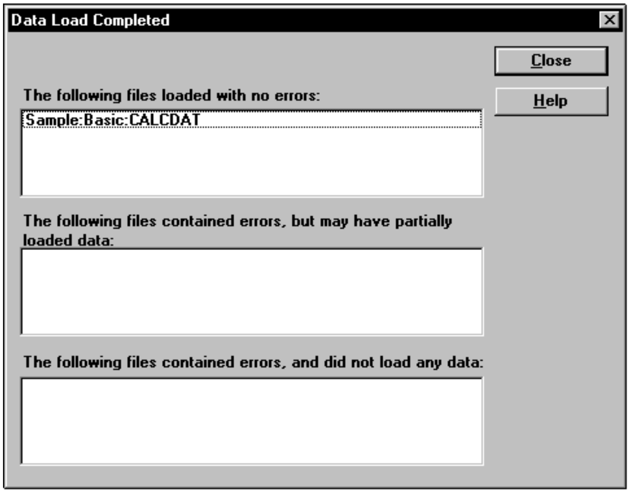
When the data load or dimension build finishes, Essbase displays a dialog box listing the results. Data loads and dimension builds end in one of the following:

- 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, both dialog boxes appear.

Complete Load

In a complete load, Essbase had no problems loading every specified record in each data source. When data source(s) load completely, Essbase lists the data sources successfully loaded in the following dialog box. In Figure 22-7, for example, the `Calcdat` file loaded successfully.



Data Load Completed

The following files loaded with no errors:

Sample:Basic:CALCDAT

The following files contained errors, but may have partially loaded data:

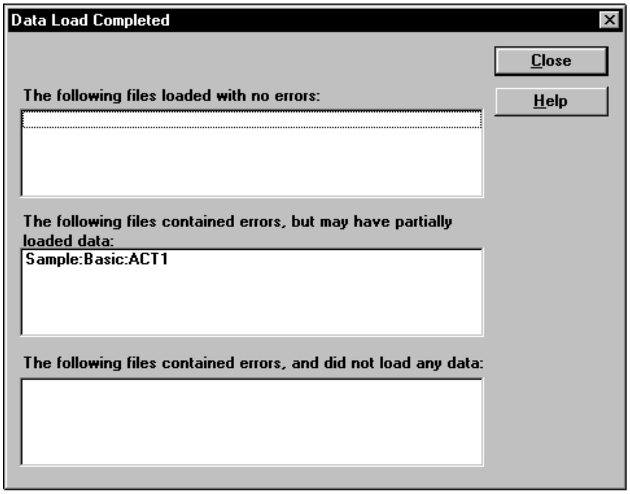
The following files contained errors, and did not load any data:

Figure 22-7, Data Load Completed Dialog Box

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 22-8, for example, the `Act1` text file did not load successfully.



Data Load Completed

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:

Figure 22-8, Partial Data Load

To fix the data source

1. Open the error log file. It's located in `\ESSBASE\CLIENT\DATALOAD.ERR` for data loads and `\ESSBASE\CLIENT\DIMBUILD.ERR` for dimension builds. If there is no error log for data load, set one and restart the load. If there is no error log for the dimension build, it means the dimension build was successful. See "Setting the Error Log File" on page 22-10.

Note: Essbase only creates an error log file if you are using a rules file.

2. Browse through the error log file. It contains a list of each of the data sources and records that didn't load. Figure 22-9 is the error log that Essbase created while trying to load the `Act1` file.

```
\\ Member  Sales Not Found In Database
California,Caffeine Free Cola, Sales,
145,132,125,110,106,96,87,87,109,109,116,102

\\ Member  COGS Not Found In Database
California,Caffeine Free Cola, COGS,
95,104,109,123,127,141,154,154,122,122,113,127

\\ Member  Marketing Not Found In Database
California,Caffeine Free Cola, Marketing, 30,33,34,39,40,45,49,49,39,39,36,40

\\ Member  Payroll Not Found In Database
California,Caffeine Free Cola, Payroll, 22,22,22,23,23,23,22,22,22,22,22,22

\\ Member  Misc Not Found In Database
California,Caffeine Free Cola, Misc, 0,0,1,1,0,0,0,1,0,0,1,1
```

Figure 22-9, Error Log for Partial Data Load

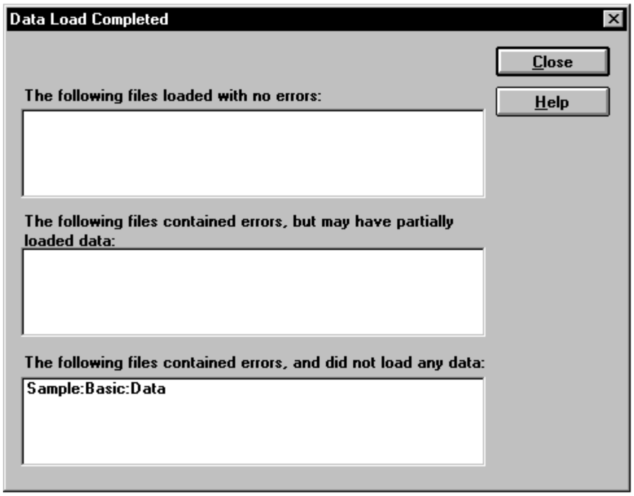
3. After you determine what didn't load, open the data sources or records that didn't load and fix them. To fix the data source in Figure 22-9, for example, either edit the data source to remove the invalid members (that is, Sales, COGS, Marketing, Payroll, and Misc) or, if those members are all in the same column, ignore all fields in that column, see "Ignoring All Fields in a Column" on page 21-3.

   To view a specific record in a data source from the Data Prep Editor, see "Setting the Records Displayed" on page 20-17.

4. When you have fixed the problem with the database, you may be able to reload just the records that failed. See "Loading the Error Log File" on page 23-4 for more information.

No Load

When no data sources or records were loaded into the database, the following dialog box appears:



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

Figure 22-10, No Data Loaded

To fix the data sources:

1. Open the error log file for the data load. If you didn't set an error log, set one and restart the load. See "Setting the Error Log File" on page 22-10.
2. Browse through the error log file. It contains a list of each of the data sources and records that didn't load. In Figure 22-10, for example, `Data` did not load.
3. After you determine what didn't load, open the data sources or records that didn't load and fix them. To jump directly to a record in a data source, see "Setting the Records Displayed" on page 20-17.
4. When you have fixed the problem with the database, you can reload the records.

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, don't load data directly into a parent.
- If you must load data into the parent member: set `AGGMISSING` to off so the children don't aggregate into the parent. 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 "Calculating #MISSING Values" on page 32-29 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 21-2.
4. Define a reject 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 20-14.

Note: You cannot reject more records than the error log file can hold. By default, this is 1000, but you can change it by setting the `DATAERRORLIMIT` in the `ESSBASE.CFG` file. See the online *Technical Reference* in your `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 Essbase Visual Basic API. See ESBIMPORT() in the online *API Reference* in your `DOCS` directory. See the Spreadsheet Add-in Online Help for information about using VBA with the Visual Basic API.

Chapter 23 Debugging and Optimizing Data Loads

This chapter describes how to debug data loads by finding the problem, fixing the problem, and loading the failed records. In addition, this chapter describes how to optimize your data loads. This chapter contains the following sections:

- "Debugging a Data Load" on page 23-1
- "Optimizing Data Loads" on page 23-6

Debugging a Data Load

If you tried to load a data source into Hyperion Essbase OLAP Server, but it did not load correctly, check the following:

- Were 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 file generated for that data load. It lists the errors that occurred when Essbase tried to load the data source. The error log file is located on the client machine in `\ESSBASE\CLIENT\DATALOAD.ERR`.

If there is no error log file, check the following:

- Did the person running the data load set one up? See "Setting the Error Log File" on page 22-10 for information on setting up an error log. By default, Essbase creates an error log whenever you load data using a rules file.
- Are you sure that the data source and Essbase server are available? See the "Verifying that the Server Is Available" on page 23-3 and "Verifying that the Data Source Is Available" on page 23-4 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 23-5.

If the error log file exists but is empty, that means that Essbase doesn't think an error occurred during loading. Check the following:

- Does the rules file contain select/reject criteria that rejected every record in the data source? See "Selecting Records" on page 20-13 if you don't know how to set up select/reject criteria.
- Is the rules file correct? Does the rules file validate properly? See "Problems Validating a Rules File" on page 23-5.

Not All Errors in the Log File

When Essbase can't load a record, it writes it to the error log file, `DATALOAD.ERR` on the client. There is, however, a limit to the number of records that an error log can contain. By default, it's 1000 records, but you can set it to be lower than 1000 by setting DATAERRORLIMIT in the `ESSBASE.CFG` file. See the online *Technical Reference* in your `DOCS` directory for more information.

When Essbase writes the maximum allowed number of records in the error log file, it doesn't log any other errors it encounters. The data load, however, continues. Any subsequent errors are lost.

Data Loaded Incorrectly

If the data source loaded 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. 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 21-14.
- 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 12-18.
- Did you add incoming data to existing data instead of replacing it using a rules file? See "Changing Data Values" on page 21-20.
- Have you selected or rejected any records that you didn't intend to select or reject using a rules file? See Chapter 20, 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 user-defined attributes in your rules file using a rules file? See “Flipping Field Signs” on page 21-24.
- Did you clear data combinations that you didn’t intend to clear using a rules file? See “Clearing Existing Data Values” on page 21-21.
- Did you scale the incoming values incorrectly using a rules file? See “Scaling Data Values” on page 21-23.
- Are all member and alias names less than 80 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 43, Performing Interactive and Batch Operations Using ESSCMD. 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 the “Loading the Error Log File” on page 23-4.

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 can’t, check with your system administrator.
- Is the Essbase 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?

Loading the Error Log File

If your Isolation Level transaction setting is Committed, you must re-start the data load from the beginning. If your Isolation Level is Uncommitted, you can load just the records that failed by loading the error log. Essbase copies each unloaded record to the error log file during loading. Just reloading these records is much faster than loading each data source again, including loading those records that succeeded during the first load.

For more information on Isolation Level settings, see “About Isolation Levels” on page 41-2.

Make sure you fixed the problem that caused the errors. Then load the error log:

1. If you are loading from the server, rename the `DATALOAD.ERR` file to `DATALOAD.TXT`. Essbase can only load text files that end in `.TXT` on the server. If you are loading from the client, the file can have any name valid on the local operating system.

Note: If you are reloading the dimension build error file, it’s called `DIMBUILD.ERR`.

2. Load the `DATALOAD.TXT` file using the same rules file you used for the original data sources. If you don’t know how to load a data source, see Chapter 19, Introducing Data Loading.

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 your Isolation Level transaction setting is Committed, you must re-start the data load from the beginning. If your 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 “About Isolation Levels” on page 41-2.

Problems Validating a Rules File

If you can’t validate your rules file, check to make sure that it is set up correctly. Make sure that:

- 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 aren’t also in the data records.
- You didn’t define more than one field as a data field.

Ignoring 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 21-4 for information on how to do this.
3. Define a rejection criterion that rejects the end of file marker. See “Rejecting Records” on page 20-14 for information on how to do this.

Optimizing Data Loads

Loading large data sources into an Essbase database can take a great deal of time. However, you can speed up the data loading process. To speed up a data load, it is important to:

- Minimize the time spent reading and parsing the data source.
- Minimize the time spent reading and writing to the database.

This section contains the following subsections:

- “How Does Essbase Load Data?” on page 23-7
- “Grouping Sparse Member Combinations Together” on page 23-7
- “Positioning Data in the Same Order as the Outline” on page 23-9
- “Loading from the Server” on page 23-9
- “Making the Data Source as Small as Possible” on page 23-10
- “Making the Fields as Small as Possible” on page 23-11

How Does Essbase Load Data?

When Essbase loads a data source, it does the following:

1. Uses the index to find the correct block on the disk. The index is composed of the sparse dimensions.
2. Loads the block into the cache, and loads the data into it.
3. Loads the block on the disk that the next record corresponds to. If the correct block is already in the cache, Essbase doesn't have to load the first block.
4. Repeats this process until each field is loaded.

See Chapter 4, Basic Architectural Elements, for information on sparse and dense data combinations.

Grouping Sparse Member Combinations Together

If you arrange your data source so that records with the same sparse member combinations are consecutively grouped, the data loads more quickly. The order of your fields is irrelevant, so long as the data that is changing from record to record is in the same block as the previous record. Continue this until there's no more data for the block and then change one of the sparse dimensions to access a different block, and so on. Thus, your data source should group all records by block.

The file in Figure 23-1, for illustration purposes, displays its fields such that you can easily see the sparse dimensions on the left and dense dimensions on the right, based on the structure of the Sample Basic database.

Sparse dimensions:

- Scenario
- Product
- Market

Dense dimensions:

- Measures
- Year

```
Scenario   Product   Market    Measures  Year
Actual     Cola      Ohio      Sales     Jan     25
```

Figure 23-1, Sample Basic Database Showing Sparse and Dense Dimensions

Sort the records in the data source so that records with like values in the sparse dimensions are together. Then specify all the combinations of members in the dense dimensions, before specifying a different member in a sparse dimension. Figure 23-2, for example, sorts the records to put like records together. The values for the Measures dense dimension change first. Essbase accesses one block.

```
Jan
Actual     Cola      Ohio      Sales     25
Actual     Cola      Ohio      Margin    18
Actual     Cola      Ohio      COGS      20
Actual     Cola      Ohio      Profit    5
```

Figure 23-2, Sorted Records

Figure 23-3, on the other hand, does not sort its records and changes its sparse dimensions before the dense ones. It loads more slowly than Figure 23-2, because Essbase accesses four different blocks instead of one.

```
Jan
Actual   Cola         Ohio      Sales   100
Budget   "Root Beer"  Florida   Sales   96
Actual   "Root Beer"  Ohio      Sales   145
Budget   Cola         Florida   Sales   85
```

Figure 23-3, Unsorted Records

If you are using a data source that loads more than one cell per record, use the same idea as Figure 23-2, but arrange the data so that the expanded dimension in the record is a dense dimension.

For more information on creating rules files, see the “Introduction to Rules Files” on page 19-8. For more information on dense and sparse dimensions, see Chapter 4, Basic Architectural Elements.

Why Does This Speed up the Data Load?

Positioning your data based on sparse member combinations in the data source speeds up the data load because of how Essbase stores sparse and dense dimensions. All data is stored in blocks. A block contains cells for all possible dense dimension intersections. Essbase creates a block offset that points to the intersections in the block where the data is stored. The intersection of the sparse dimensions forms an index entry that points to the block where the data is stored.

So, when you put the sparse member combinations together, Essbase uses the index to find the block where the data is stored and loads that block into its cache. Essbase then uses the block offset to determine which parts of the block to change and writes to that block multiple times. Because the correct block is in the cache, you don’t have to open it each time you write to parts of the block. This reduces the number of physical disk I/O’s required, which speeds up your data load.

Positioning Data in the Same Order as the Outline

After you arrange your data source so that the sparse data combinations are together, rearrange it so that sparse dimensions are in the same order as the outline.

Why Does This Speed up the Data Load?

Positioning fields in the data source to match sparse dimensions in the outline speeds up the data load because of the way Essbase accesses data using the index. Essbase uses the index cache size to determine how much of the index can be paged into memory. Essbase pages portions of the index in and out of memory as requested by the data load or other operations. If the data in your data source lists records in the same order as the outline, then less paging of the index occurs, thus reducing the I/O's required.

Note: If your index cache size is large enough to hold the index in memory, then this method does not speed your data load.

For more information about setting the index cache size, see "Using the Settings Dialogs in the Application Manager" on page 40-5. For more information about choosing the size of the index cache, see Chapter 14, Sizing Your Database.

Loading from the Server

If you load the data source from the server instead of the client, the data loads more quickly. To load a data source from the server, move the data source to the server and then start the load.

Why Does This Speed up the Data Load?

Using the server speeds up the data load because the data does not have to be transported over the network from the client to the server.

Making the Data Source as Small as Possible

Make your data source as small as possible.

If you are using free-form data, set up ranges in the data source. Ranges reduce the number of fields Essbase must read before loading data values. Figure 23-4 shows a file that doesn't use ranges and Figure 23-5 shows the same file optimized to use ranges. Figure 23-4 contains 32 fields that Essbase must read in order to load the data values properly.

```
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 23-4, Data Source Without Ranges

Figure 23-5 contains only 22 fields that Essbase must read in order to load the data values properly. In ranges, the last range cycles the fastest. In Figure 23-5, for example, the first range encountered is Jan and Feb (from the Year dimension), the second range is New York and Ohio (from the Market dimension), and the third range is Cola and Diet Cola (from the Product dimension). The last range encountered, in this case, is Cola and Diet Cola. So Essbase assigns the first value, 4, to Jan, New York, Cola. Essbase assigns the second value, 3, to Jan, New York, Diet Cola. The third value, 8, is assigned to Jan, Ohio, Cola. Essbase continues in this order until the file is loaded.

```
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
```

Figure 23-5, Data Source with Ranges

Why Does This Speed up the Data Load?

The less there is to read in a data source, the less time it takes Essbase to read it. Therefore, as long as a data source is complete, the smaller it is, the faster Essbase can read and load it.

Making the Fields as Small as Possible

Make the fields in the data source as small as possible by:

- Removing excess white space in the data source.
- Rounding off computer-generated numbers to only the precision you need. For example, if the data value has nine decimal points and you only care about two, round the number off.
- Using `#MI` instead of `#MISSING`.

Why Does This Speed up the Data Load?

The less there is to read in a data source, the less time it takes Essbase to read it. Therefore, as long as a data source is complete, the smaller it is, the faster Essbase can read and load it.

Part V Calculating Your Data

Part V describes how to calculate the data in Hyperion Essbase OLAP Server databases, including how to create formulas, define the order in which Essbase calculates dimensions and members, calculate data value dynamically, calculate time series data, create calculation scripts, and optimize calculations. Part V contains the following chapters:

- Chapter 24, Introduction to Database Calculations, introduces you to the basic concepts behind database calculations.
- Chapter 25, Developing Formulas, introduces you to 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 Calc Scripts, introduces you to calc scripts and describes how to create calc scripts using the Calc Script Editor.
- Chapter 31, Examples of Calc Scripts, contains detailed examples of calc scripts.
- Chapter 32, Optimizing Your Calculations, describes how to make your calc scripts execute more quickly.
- Chapter 33, Optimizing Your Calculation Using Intelligent Calculation, describes how to use intelligent calculation to make your calc scripts execute more quickly.

Chapter 24 Introduction to Database Calculations

This chapter describes how you can calculate your database. It also explains the concept of calculating a multidimensional database.

This chapter includes the following sections:

What Are Database Calculations?

Your database contains two types of values. It contains the values you enter, which are called *input data*, and the values that have been calculated from this input data.

For example:

- You enter regional sales figures for a variety of different products. You calculate the total sales for each product for all regions.
- You enter the budget and actual values for the Cost of Goods for several products in different regions. You calculate the variance between the two 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%.

Hyperion Essbase offers two ways that you can calculate your database:

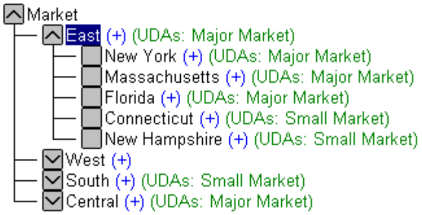
- Outline calculation
- Calc script calculation

Which way you choose depends on the type of calculation you want to do.

Outline Calculation

Outline calculation is the simplest method of calculation. Essbase calculates your database based on the relationships between members in the database outline and on any formulas that you have attached to members in the outline.

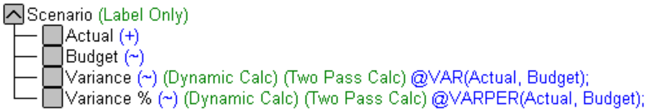
For example, Figure 24-1 shows the relationships between the members of the Market dimension in the Sample Basic database. The values for New York, Massachusetts, Florida, Connecticut, and New Hampshire are added together to calculate the value for East. The values for East, West, South, and Central are added together to calculate the total value for Market.



```
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 24-1, Relationship Between Members of the Market Dimension

Figure 24-2 shows the Scenario dimension from the Sample Basic database. The Variance and Variance% members are calculated using the formulas attached to them.



```
Scenario (Label Only)
├─Actual (+)
├─Budget (~)
├─Variance (~) (Dynamic Calc) (Two Pass Calc) @VAR(Actual, Budget);
└─Variance % (~) (Dynamic Calc) (Two Pass Calc) @VARPER(Actual, Budget);
```

Figure 24-2, Calculation of Variance and Variance%

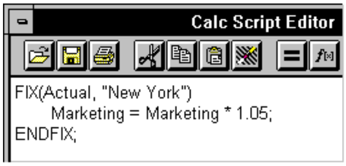
For more information on creating database outlines, see Chapter 8, Creating and Changing Database Outlines.

When you design your 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 do this. For more information, see Chapter 28, Dynamically Calculating Data Values.

Calc Script Calculation

Calc script calculation is the second method of calculation. Using a calc (calculation) script you can choose exactly how to calculate your database. For example, you can calculate part of your database or copy data values between members.

A calc script contains a series of calculation commands, equations, and formulas. For example, the following calc script increases the actual Marketing expenses in the New York region by 5%.



```
FIX(Actual, "New York")
      Marketing = Marketing * 1.05;
ENDFIX;
```

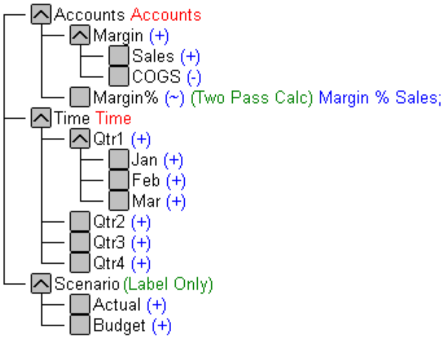
Figure 24-3, Calc Script Editor

For more information on calc scripts, see Chapter 30, Developing Calc 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:



```
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 (+)
```

Figure 24-4, 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. In the example, we consider 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.

The intersection of one member on each dimension represents a data value. Our example has three dimensions; therefore, we can represent the dimensions and data values in the database as a cube:

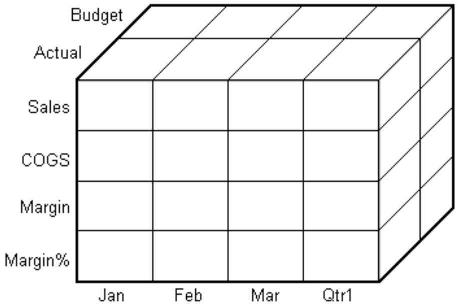


Figure 24-5, Multidimensional Database Concepts

From the following diagram you can see that when you refer to Sales, you are referring to a slice of the database containing eight Sales values:

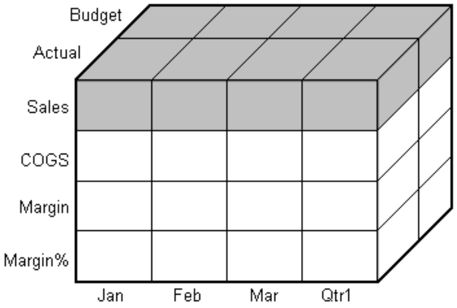


Figure 24-6, Multidimensional Database Concepts

When you refer to Actual Sales, you are referring to four Sales values:

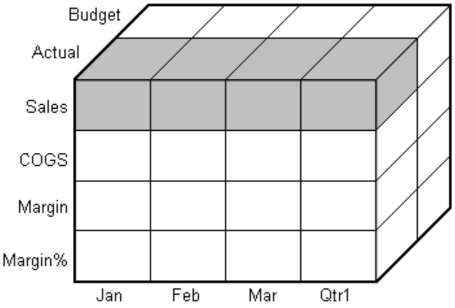


Figure 24-7, Multidimensional Database Concepts

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 the following diagram, the cell containing the data value for Sales, Jan, Actual is shaded.

In Essbase member combinations are denoted by the cross-dimensional operator. The symbol for the cross-dimensional operator is ->. So Sales, Jan, Actual is written Sales->Jan->Actual.

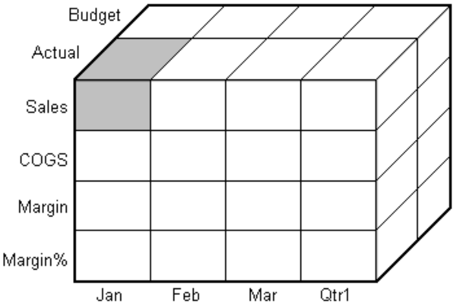


Figure 24-8, Multidimensional Database Concepts

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:

1. Margin->Jan->Actual as a percentage of Sales->Jan->Actual. It then places the result in Margin%->Jan->Actual.
2. Margin->Feb->Actual as a percentage of Sales->Feb->Actual. It then places the result in Margin%->Feb->Actual.

3. Margin->Mar->Actual as a percentage of Sales->Mar->Actual. It then places the result in Margin%->Mar->Actual.
4. Margin->Qtr1->Actual as a percentage of Sales->Qtr1->Actual. It then places the result in Margin%->Qtr1->Actual.
5. Margin->Jan->Budget as a percentage of Sales->Jan->Budget. It then places the result 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 the database calculation order, see Chapter 27, Defining the Calculation Order.

Setting the Default Calculation

By default, the default calculation for your database is a CALC ALL of the database outline, which consolidates the dimensions and members and calculates any formulas in the outline.

However, you can specify any calc script as the default database calculation. This lets you assign a frequently used script to the database rather than loading it each time you want to perform the calculation. Also, if you want your calc script to work with settings defined in the **Calc Options** group of the **Database Settings** dialog box, you must set the calc script as the default calculation.

To set the default calculation:

1. In Hyperion Essbase Application Manager, select Database | Set Default. Essbase displays the **Set Default Calc** dialog box.
2. Select **Use Calc Script Object**.
3. Select a calc script from the list.
4. Click OK.

You can use the SETDEFAULTCALCFILE command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

You can use the SETDEFAULTCALC command in ESSCMD to set a calculation string as the default database calculation. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Calculating Your Database

You can calculate your database using:

- Application Manager
- The Hyperion Essbase Spreadsheet Add-in
- ESSCMD

This section includes information on calculating your database using Application Manager and ESSCMD. For information on calculating your database from the Spreadsheet Add-in, see the *Spreadsheet Add-in User's Guide*.

Calculating Your Database

Application Manager lets you run the default outline calculation from the desktop. To calculate a database from Application Manager:

1. In the **Server** dialog box, select the appropriate application and database.

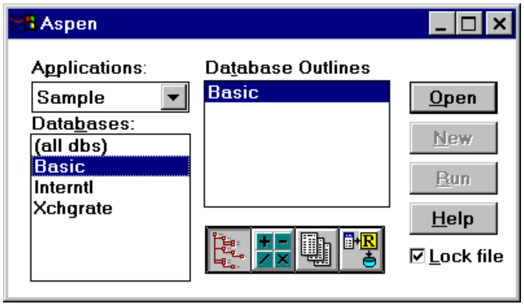
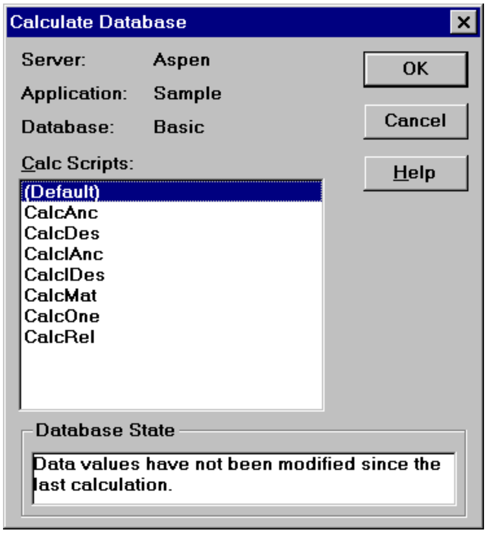


Figure 24-9, Server Dialog Box

2. Choose Database | Calculate from the Application Manager menu. Essbase displays the **Calculate Database** dialog box:



Server: Aspen

Application: Sample

Database: Basic

Calc Scripts:

(Default)
CalcAnc
CalcDes
CalcIAnc
CalcIDes
CalcMat
CalcOne
CalcRel

Database State

Data values have not been modified since the last calculation.

Figure 24-10, Calculate Database Dialog Box

Under **Database State,** Essbase indicates the current calculation state of the database:

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 any 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 any calculation of a subset of the database.

3. Select **Default** from the list to run the default outline calculation. Or, select a calc script to run the calc script. Only calc scripts to which you have security access appear in the list. For information on Essbase security privileges, see Chapter 16, Managing Security at Global and User Levels.

4. Click OK. Essbase calculates the database.

Note: Essbase displays a message while it is calculating the database. For lengthy calculations, you may want Essbase to perform the calculations in the background. Use the Windows `Alt+Tab` key combination to switch to another Windows application. While Essbase is calculating the database, other Application Manager functions are disabled.

You can use the CALC, CALCDEFAULT, and CALCLINE commands in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about each of these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Canceling a Calculation

To stop a calculation before Essbase completes it:

- Click the **Cancel** button in the **Calculating** 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 Storage Manager Isolation Level settings. For more information on these settings, see Chapter 41, Ensuring Data Integrity.

Security Considerations

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. This means that you can calculate a value even if your security filter does not let you read or update the value. 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 16, Managing Security at Global and User Levels.

Chapter 25 Developing Formulas

This chapter explains how to develop and use formulas to calculate your 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:

- “What Is a Formula?” on page 25-1
- “Example of Creating a Formula in an Outline” on page 25-3
- “Building Formulas in the Formula Editor” on page 25-6
- “Writing Formulas” on page 25-21
- “Working with Formulas in Application Partitions” on page 25-36

What Is a Formula?

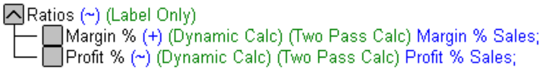
Formulas calculate relationships between members in your database outline. You can use these formulas in two ways:

- Apply them to members in the database outline.
- Place them in a calc script.

To optimize your calculation performance, consider carefully how you use formulas. For more information, see Chapter 32, Optimizing Your Calculations.

In most cases you can optimize your calculation performance by applying formulas to members in the database outline. However, if you need to control your database calculation more carefully, you can use a calc script and include formulas. For more information, see Chapter 30, Developing Calc Scripts.

The following shows the Measures dimension from the Sample Basic database. The Margin% and Profit% members are calculated using the formulas applied to them.



```
Ratios (~) (Label Only)
├─Margin % (+) (Dynamic Calc) (Two Pass Calc) Margin % Sales;
└─Profit % (~) (Dynamic Calc) (Two Pass Calc) Profit % Sales;
```

Figure 25-1, Calculation of Margin% and Profit%

How you use formulas can have significant implications for your calculation performance. Consider the information in Chapter 32, Optimizing Your Calculations, before designing and creating your formulas.

Calculating Formulas

For formulas applied to members in the database outline, Hyperion Essbase calculates formulas when:

- You do a default (CALC ALL) calculation of your database.
- You run a calc script that calculates the member containing the formula; for example, a CALC DIM of the dimension containing the member, or the member name itself. See Chapter 30, Developing Calc Scripts.

However, if the formula is associated with a dynamically-calculated member, Hyperion Essbase calculates the formula when the user requests the data values. For more information, see Chapter 28, Dynamically Calculating Data Values.

For formulas in a calc script, Hyperion Essbase calculates the formula when it occurs in the calc script.

In a calc script, you cannot do a member calculation of 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 your database outline, or in a calc script, can significantly affect calculation performance. This is because Essbase has to interrupt the regular calculation to perform the dynamic calculation. For more information, see Chapter 28, Dynamically Calculating Data Values.

Example of Creating a Formula in an Outline

This section provides a step-by-step example of creating and saving a simple formula in your outline.

For an example of creating a formula in a calc script, see Chapter 30, Developing Calc Scripts.

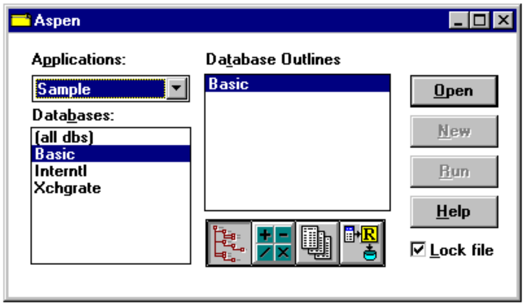
For detailed information on creating formulas, and obtaining the required calculation results, consider all the information in Part III, Designing and Building a Security System.

This example is based on the Sample Basic database, which is supplied with your Hyperion Essbase OLAP Server installation.

This example shows you how to create the formula on the Variance member of the Scenario dimension. This formula calculates the variance between Budget and Actual values.

To create this formula:

1. Start Hyperion Essbase Application Manager, and connect to your Essbase server.
2. Select the Sample application and the Basic database, and click the **Open** button to open the Sample Basic outline.



Applications: Sample

Databases: Basic

Database Outlines: Basic

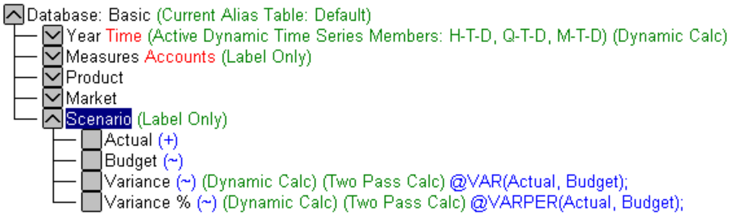
☑ Lock file

Figure 25-2, Application Desktop Window

If you do not have Sample Basic installed, contact your Essbase administrator.

If another user has Sample Basic open and locked, you can uncheck **Lock file** in the bottom right-hand corner of the Application Desktop window. However, if you do this, you will not be able to save your work.

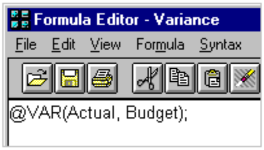
3. Double-click the Scenario dimension to display its members.



```
Database: Basic (Current Alias Table: Default)
  Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D, M-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);
```

Figure 25-3, Sample Basic Outline

4. Select the Variance member in the outline, and click the = button. Essbase displays the formula in the Formula Editor.



```
@VAR(Actual, Budget);
```

Figure 25-4, Formula Editor Showing Variance Formula

5. We are going to recreate this formula, so select the existing formula and choose Edit | Delete in the Formula Editor.

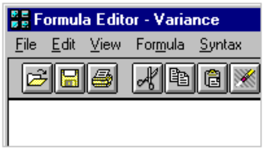
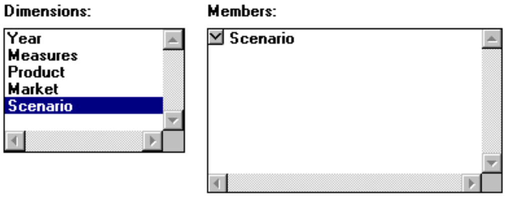


Figure 25-5, Formula Editor With Variance Formula Deleted

6. In the **Dimensions** list, select Scenario. Scenario appears in the **Members** list.



Dimensions:
Year
Measures
Product
Market
Scenario

Members:
Scenario

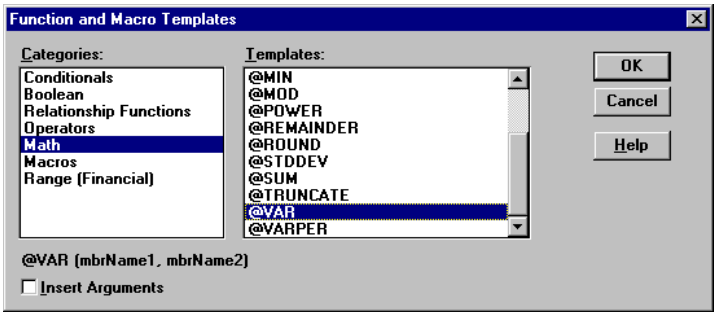
Figure 25-6, Formula Editor Dimensions and Members Lists With Scenario Selected

7. In the **Members** list, double-click the ⌵ button next to Scenario to display the members under Scenario.

8. Choose Formula | Paste Function or click the *f(x)* button. The **Function and Macro Templates** dialog box appears.

9. Select **Math** in the **Categories** list.

10. Select @VAR in the **Templates** list. Essbase displays the function, operator or macro and the default arguments below the **Categories** list.



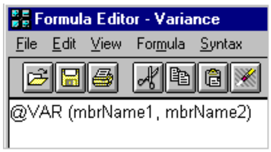
@VAR (mbrName1, mbrName2)

☐ Insert Arguments

Figure 25-7, Function and Macro Templates Dialog Box

11. Check **Insert Arguments** to insert default, temporary arguments in the function.

12. Click OK. Essbase inserts `@VAR (mbrName1, mbrName2)` at the cursor position.



@VAR (mbrName1, mbrName2)

Figure 25-8, Formula Editor With Variance Formula Added

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 the 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 the Formula Editor

You use the Formula Editor to build formulas. You can type the formulas directly into the formula text area, or you can use the Formula Editor user interface features to build the formula.

Formulas are ASCII text. If required, you can create your formula in the text editor of your choice and paste it into the formula editor.

Essbase provides a comprehensive set of operators and functions, which you can use to construct formula calculations on your database.

You can construct formulas from:

- Operators
- Functions
- Dimension names, member names and numeric constants

Operators

The following table shows the types of operators you can use in your formulas:

Operator	**Description**
Mathematical	Perform common arithmetic operations. For example, you can add or subtract values. For a complete list of the mathematical operators, see the online *Technical Reference* in your `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 online *Technical Reference* in your `DOCS` directory. For more information on writing conditional formulas, see “Specifying Conditions” on page 25-22.
Cross-dimensional	Point to the data values of specific member combinations; for example, to point to the sales value for a specific product in a specific region. For more information, see “Using the Cross-Dimensional Operator (->)” on page 25-34.

See “Inserting Text and Operators in Your Formula” on page 25-11 for information on how to add operators to your formulas.

Functions

The following table shows the types of functions you can use in your formulas:

Operator	Description
Mathematical	Perform specialized statistical 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.
Index	Look up data values within a database during a calculation. For example, you can use the @NEXT function to return the value of the next member after the current member on a chosen dimension. 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.
Macro	Generate lists of members based on a specified member. For example, you can use the @ICHILDREN function to expand a member to include its children. For more information, see the online *Technical Reference* in your `DOCS` directory.
Boolean	Provide a conditional test by returning either a TRUE (1) or FALSE (0) value. For example, you can use the @ISMBR function to check if the current member is one that you specify. For more information, see Chapter 26, Examples of Formulas.

For a complete list of the operators, functions, and syntax, see the online *Technical Reference* in your `DOCS` directory.

Formula Syntax

When you create member formulas, you need to apply the following rules:

- End each statement in the formula with a semicolon (;). For example,

  ```
  Margin % Sales;
  ```

- Enclose the member name in double quotation marks ("") if the member name:
  - Contains spaces
  - Is the same as an operator name
  - Includes any non-alphanumeric character; for example, a hyphen (-)
  - Is all numeric; for example, 100

  For example,

  ```
  "Opening Inventory" = "Ending Inventory" - Sales + Additions;
  ```

- End each IF statement with an ENDIF statement. For example, you can apply the following formula to the Commission member in a database outline:

  ```
  IF(Sales < 100) Commission = 0;
  ENDIF
  ```

  The IF and ENDIF statements do not need to be followed by a semicolon (;).

- If you are using an IF...ELSEIF... statement nested within another IF...ENDIF statement, supply an ENDIF statement for each ELSEIF statement as well as all the IF statements. For more information, see the online *Technical Reference* in your `DOCS` directory.

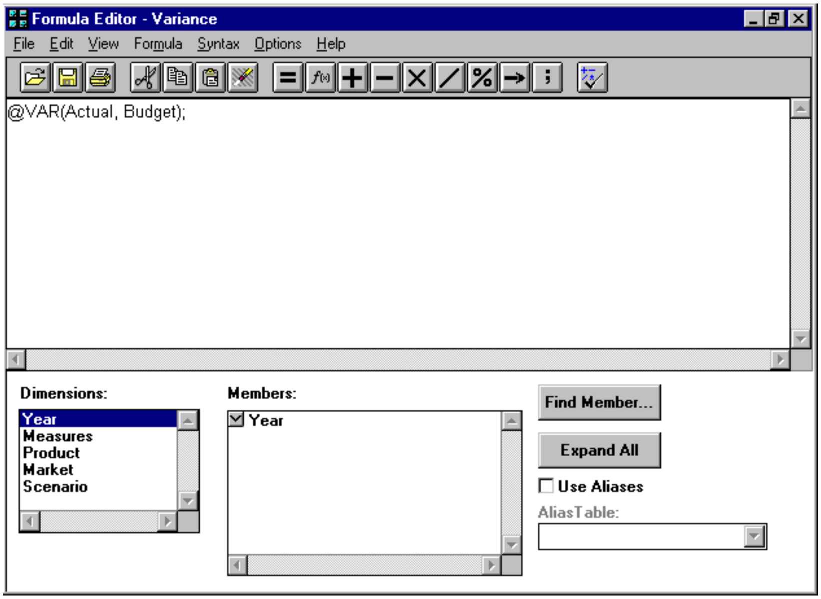
When writing formulas, you can check the syntax using the Formula Editor syntax checker. For more information, see “Checking Syntax” on page 25-20.

For detailed information on formula syntax, see the online *Technical Reference* in your `DOCS` directory.

Opening the Formula Editor

To open the Formula Editor:

1. In the Application Manager Outline Editor, highlight the member whose formula you want to create or edit.
2. Choose Edit | Formula or click the **Formula Editor** button =.
3. Essbase opens the Formula Editor for the selected member. If the member already has a formula, the formula appears in the Formula Editor. The following shows the Formula Editor for Variance in the Sample Basic database.



```
@VAR(Actual, Budget);
```

Figure 25-9, Formula Editor Window

Displaying a Formula

Open the database outline to display the members and associated formulas in the Outline Editor. You can also highlight the member for which you want to see the formula and open the Formula Editor. See “Opening the Formula Editor” on page 25-9.

You can use the GETMBRCALC command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD for information about ESSCMD.

Adding a Formula

Highlight the member for which you want to add a formula and open the Formula Editor. See “Opening the Formula Editor” on page 25-9. Type or insert the formula in the Formula Editor window. See “Inserting Text and Operators in Your Formula” on page 25-11.

Changing a Formula

Highlight the member which has the formula you want to edit and open the Formula Editor. See “Opening the Formula Editor” on page 25-9. Make the required changes to the formula.

Saving a Formula

To save a formula:

1. In the Formula Editor, choose File | Save or click the 💾 button to save the changes in the Formula Editor.

2. Close the Formula Editor and click the 💾 button in the Outline Editor to save the changes in the database outline. Essbase displays the formula beside the member in the database outline.

Printing a Formula

In the Formula Editor, choose File | Print, or click the 🖨 button.

Deleting a Formula

To delete a formula:

1. In the Outline Editor, select the member with the formula that you want to delete.
2. Click the = button to open the Formula Editor. The formula appears in the Formula Editor window.
3. Select the text of the formula.
4. Choose Edit | Delete to delete the text of the formula.
5. Click the 💾 button to save the changes in the Formula Editor.
6. Close the 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.

Undoing the Last Action

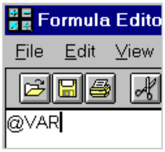
In the Formula Editor, select Edit | Undo, or click the ✎ button.

Inserting Text and Operators in Your Formula

You can type text and operators directly into the formula text area, or you can use the toolbar buttons to add the text and operators.

Typing Text

To type text, in the Formula Editor, click in the formula text area below the toolbar, and begin to type. Text appears at the cursor position as you type.



@VAR

Figure 25-10, Adding a Formula in the Formula Editor

Cutting, Copying and Pasting Text

To cut text, in the Formula Editor, select the text that you want to cut and do one of the following:

- Choose Edit | Cut.
- Click the ✂ button.
- Press `Ctrl+X`.

To copy text, in the Formula Editor, select the text that you want to copy and do one of the following:

- Choose Edit | Copy.
- Click the 📄 button.
- Press `Ctrl+C`.

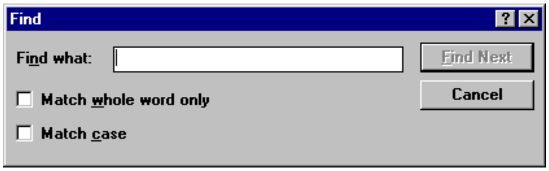
To paste text, in the Formula Editor, select the text that you want to paste and do one of the following:

- Choose Edit | Paste.
- Click the 📋 button.
- Press `Ctrl+V`.

Finding and Replacing Text

To find and replace text in your formula:

1. In the Formula Editor, choose Edit | Find. Essbase displays the **Find** dialog box.



Find

Find what:

□ Match whole word only

□ Match case

Figure 25-11, Formula Editor Find Text Dialog Box

2. In the **Find what** text box, type the characters that you want to search for and click the **Find Next** button.

To search for case-sensitive characters:

1. Check **Match case** in the **Find** dialog box.

   For example to search for Margin, but not margin, type `Margin` in the **Find what** text box, and check **Match case**.

2. Click **Find Next**.

To search for whole words only:

1. Check **Match whole word only** in the **Find** dialog box.

   For example, to search for Margin, but not Margin%, type `margin` in the **Find what** text box, and check **Match whole word only**.

2. Click **Find Next**.

Inserting an Equal (=) Sign

In the Formula Editor, place the cursor where you want to insert the equal (=) sign, and type = or click the = button.

Inserting a Mathematical Operator (+, -, X, /, %)

In the Formula Editor, place the cursor where you want to insert the mathematical operator, and type the operator, or click one of the following toolbar buttons:

+ − × ⁄ %

For example, to insert an addition operator (+):

1. Place the cursor where you want to insert the addition operator (+).
2. Type `+`, or click the + button.

Inserting the Cross-Dimensional Operator (->)

In the Formula Editor, place the cursor where you want to insert the cross-dimensional operator, and 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 25-34.

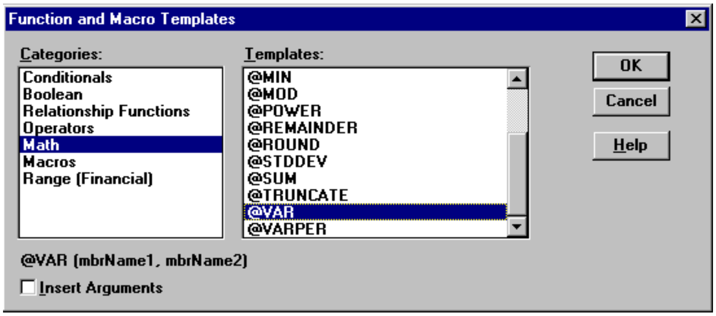
Inserting the Semicolon Formula End-of-Line Character (;)

In the Formula Editor, place the cursor at the end of the formula, and type a ; (semicolon), or click the ; button.

Inserting a Function, Operator, or Macro in Your Formula

To insert a function, operator, or macro:

1. In the Formula Editor, place the cursor where you want to insert the function or macro.
2. Choose Formula | Paste Function, or click the f(x) button.
3. Select the function category in the **Categories** list. For example, to insert the @VAR function, select **Math**.
4. Select the required function, operator, or macro in the **Templates** list. For example, scroll down the list and select @VAR. Essbase displays the function, operator, or macro and the default arguments below the **Categories** list.



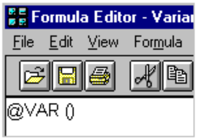
@VAR (mbrName1, mbrName2)

□ Insert Arguments

Figure 25-12, Function and Macro Templates Dialog Box With Math Category Selected

5. If required, check **Insert Arguments**, to insert default, temporary arguments in the function.

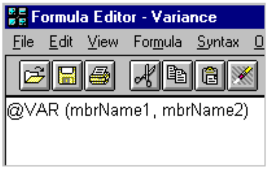
6. Click OK. Essbase inserts @VAR at the cursor position.



@VAR ()

Figure 25-13, Formula Editor Showing @VAR Formula

If you checked **Insert Arguments**, Essbase inserts @VAR and default, temporary arguments. You can then type over these with the correct arguments.



@VAR (mbrName1, mbrName2)

Figure 25-14, Formula Editor Showing @VAR with Arguments

Inserting Members in Your Formula

To insert members in your formula:

1. In the Formula Editor, place the cursor where you want to insert the member name.
2. In the **Dimensions** list, select the dimension that contains the member you want to insert in your formula.

   The dimension member name appears in the **Members** list. If a ☑ button appears to the left of the dimension member name, then the dimension has children. The following shows the Scenario dimension in the Sample Basic database.

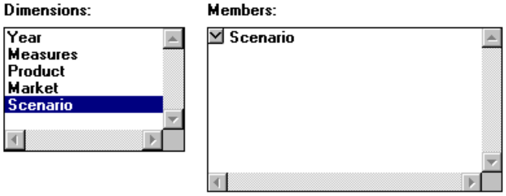


Figure 25-15, Formula Editor Dimensions and Members Lists

3. To insert the dimension member name itself in your formula, click the dimension name in the **Members** list. Essbase inserts the dimension member name at the cursor position. Otherwise, expand the member branch and select the member you want to insert. See “Expanding a Member Branch to Display a Member’s Children” on page 25-16.

Expanding 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.

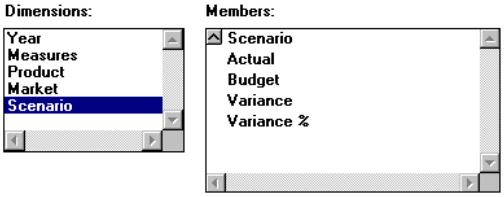
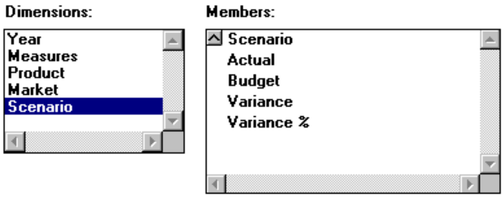


Figure 25-16, Formula Editor Dimensions and Members Lists: Expanding the Scenario Member

The ⌄ button changes to a ⌃ button. You can click the ⌃ button to collapse the member branch.

Collapsing a Member Branch

In the **Members** list, double-click the ⌃ button next to collapse the member branch.

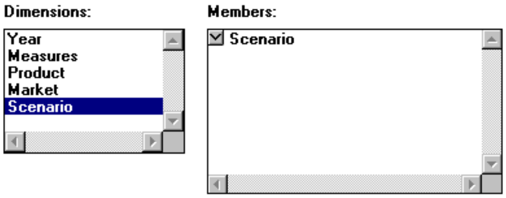


Dimensions:
- Year
- Measures
- Product
- Market
- Scenario

Members:
- ⌃ Scenario
  - Actual
  - Budget
  - Variance
  - Variance %

Figure 25-17, Formula Editor Dimensions and Members Lists: Collapsing the Scenario Member

Essbase does not display the member’s children:



Dimensions:
- Year
- Measures
- Product
- Market
- Scenario

Members:
- ⌄ Scenario

Figure 25-18, Formula Editor Dimensions and Members Lists: With the Scenario Member

The ⌃ button changes to a ⌄ button. You can click the ⌄ button to expand the member branch.

Searching for Members

To find a specific member:

1. In the **Dimensions** list, select the dimension in which you want to search for the member. For example, select the Measures dimension from the Sample Basic database.
2. Click the **Find Member** button. Essbase displays the **Find** dialog box.

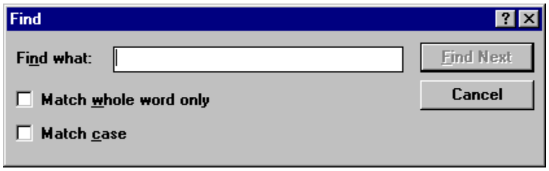
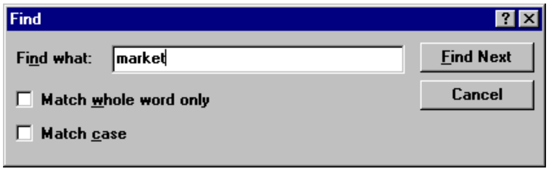


Figure 25-19, Formula Editor Find Dialog Box

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, enter `market`.



Find what: market

Figure 25-20, Formula Editor Find Dialog Box With Member

4. Click the **Find Next** button. Essbase finds and selects the Marketing member.

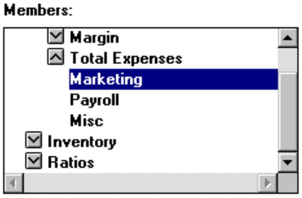


Figure 25-21, Formula Editor Members List With Marketing Selected

To enable whole words or to do a case-sensitive search, see “Finding and Replacing Text” on page 25-12.

Expanding a Dimension to Display All Members

To expand a dimension:

1. In the Formula Editor **Dimensions** list, select the dimension for which you want to display all members. For example, select the Product dimension in the Sample Basic database.

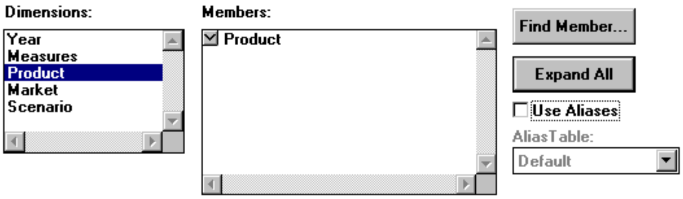


Figure 25-22, Formula Editor Dimensions and Members List With Product Selected

2. Click the **Expand All** button. In the **Members** list, Essbase displays all the members in the dimension.

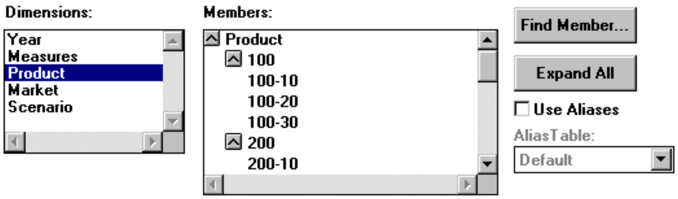
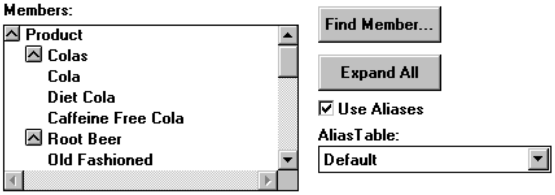


Figure 25-23, Formula Editor Dimensions and Members List Showing the Children of Product

Displaying and Inserting Alias Names

To display and insert alias names:

1. In the 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.



Members:

- Product
  - Colas
    - Cola
    - Diet Cola
    - Caffeine Free Cola
  - Root Beer
    - Old Fashioned

☑ Use Aliases

AliasTable: Default

Figure 25-24, Formula Editor Dimensions and Members List With Alias Names

2. To choose a different alias table, choose the table from the **Alias Table** list box.

   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 formula syntax checker that tells you about any syntax errors in your formulas. For example, Essbase tells you if you have mistyped a function name.

Note: The syntax checker cannot tell you about semantic errors in your formula. Semantic errors occur when your formula does not work as you expect. To find semantic errors, always run your calculation and check the results to ensure they are as you expect.

To check the syntax of a formula:

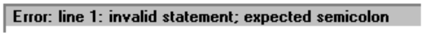
- In the Formula Editor, choose Syntax | Check, or click the button. Essbase displays the syntax checker results at the bottom of the Formula Editor window.

  If Essbase finds no syntax errors, it displays the following message:

No errors

Figure 25-25, Formula Editor Syntax Checker: No Errors Message

If Essbase finds one or more syntax errors, it displays the line number which includes the error, and a brief description. For example, if you do not include a semicolon end-of-line character at the end of your formula, Essbase displays a message similar to the following:



Error: line 1: invalid statement; expected semicolon

Figure 25-26, Formula Editor Syntax Checker Showing Syntax Error Message

Stepping Through Syntax Errors

To step through errors, choose Syntax | Next Error or Syntax | Previous Error. When you reach the first or last error, Essbase displays the message:

No more errors

Figure 25-27, Formula Editor Syntax Checker: No More Errors Message

Essbase maintains the list of error messages until you check the syntax again.

Writing Formulas

The following sections discuss and give examples of the three main types of formulas:

- Basic Equation
- Conditional
- Interdependent

These sections also discuss how to use the cross-dimensional operator in your formulas.

For more examples of formulas, see Chapter 26, Examples of Formulas.

Writing Basic Equations

You can apply a mathematical operation to a formula to create a basic equation formula. For example, you could apply the following formula to the Margin member in Sample Basic.

```
Sales - COGS;
```

In a calc script you define basic equations as follows:

```
Member = mathematical_operation;
```

where *Member* is a member name from your database outline and *mathematical_operation* is any valid mathematical operation.

For example:

```
Margin = Sales - COGS;
```

Whether it is in the database outline or in a calc script, this example cycles through the database subtracting the values in COGS from the values in Sales and placing the results in Margin.

As another example, you could apply the following formula to a Markup member:

```
(Retail - Cost) % Retail;
```

In a calc script, this would be:

```
Markup = (Retail - Cost) % Retail;
```

This example 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 tests to control the flow of your calculation.

The IF and ENDIF operators define a *conditional block*. The formulas between the IF and the ENDIF operators are executed only if the test returns True (1).

You can use the ELSE and ELSEIF operators to specify alternative outcomes if the test returns False (0). The formulas following each ELSEIF operator are only executed if the previous test returns False (0).

When you use a conditional formula in a calc 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 statement, 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*.

Boolean functions let you determine which formula to use based on characteristics of the current member combination. For example, you might want to restrict a certain calculation to those members in the Product dimension that contain input data. In this case, you would 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 attributes of the current cell to return a value of True (1).

You can use the following functions:

To determine if...	Use this function
The current member has a specified Accounts tag (for example, an Expense tag)	@ISACCTYPE
The current member is an ancestor of a specified member	@ISANCEST
The current member is an ancestor of a specified member, or the specified member itself	@ISIANCEST
The current member is a child of a specified member	@ISCHILD
The current member is a child of a specified member, or the specified member itself	@ISICHILD
The current member is an descendant of a specified member	@ISDESC
The current member is a descendant of a specified member, or the specified member itself	@ISIDESC
The current member of a specified dimension is of a specified generation	@ISGEN
The current member of a specified dimension is of a specified level	@ISLEV
The current member matches any of the specified members	@ISMBR
The current member is the parent of a specified member	@ISPARENT
The current member is the parent of a specified member, or the specified member itself	@ISIPARENT
The current member (of the same dimension as the specified member) is of the same generation as a specified member	@ISSAMEGEN

To determine if...	Use this function
The current member (of the same dimension as the specified member) is of the same level as a specified member	@ISSAMELEV
The current member is a sibling of a specified member	@ISSIBLING
The current member is a sibling of a specified member, or the specified member itself	@ISISIBLING
A specified User Defined Attribute (UDA) exists for the current member of a specified dimension	@ISUDA

When you place formulas on your database outline, you can use only the IF, ELSE, ELSEIF and ENDIF, and Boolean functions to control the flow of your calculations. You can use other control commands in a calc script. For more information, see Chapter 30, Developing Calc Scripts.

Examples

You could apply the following formula to a Commission member in the database outline. In this 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 calc script, you need to associate it with the Commission member as follows:

```
Commission(IF(Sales > 500000)
Commission = Sales * .01;
ENDIF)
```

Essbase cycles through the database, performing the following 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 this second example, the formula tests to see if the current member is a descendant of the parent member East, West or Central on the Market dimension. Essbase then calculates the Payroll value accordingly.

```
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 calc script, you need to associate it 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 @ISIDESC see the online *Technical Reference* in your `DOCS` directory.

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, which Essbase may not yet have calculated.

A good example is that of cash flow, in which the opening inventory is dependent on the ending inventory from the previous month.

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 your 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 calc script, you need to associate it 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 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 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.

Specifying a Member List or Range

In some functions you may need to specify more than one member, or a range of members. For example, the @ISMBR function tests to see if a member Essbase is currently calculating matches any of list or range of specified members. You can specify members using the following syntax:

To specify...	Use...
A single member	The member name. For example: Mar97
A list of members	A comma (,) separated list of member names. For example: Mar97, Apr97, May97
A range of all the members on the same level, between and including two defining members	The two defining member names separated by a colon (:). For example: Jan97:Dec97
A range of all the members of the same generation, between and including two defining members	The two defining member names separated by two colons (::). For example: Q197::Q491
A function generated list or range of members	See “Generating Member Lists” on page 25-29.
A combination of ranges and lists	Separate each range, list, and function with a comma (,). For example: Q194::Q497, FY94, FY95, FY96 Or: @SIBLINGS(Dept01), Dept65:Dept73, Total_Dept

If you do not specify a list or range of members in a function that requires one, 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 based on a specified member using the following functions.

To generate a list of members including...	Use this function
All the ancestors of a specified member, including ancestors of the specified member as a shared member. This function does not include the specified member itself.	@ALLANCESTORS
All the ancestors of a specified member, including ancestors of the specified member as a shared member. This function includes the specified member itself.	@IALLANCESTORS
All the ancestors of a specified member, optionally up to a specified generation or level, but not the member itself.	@ANCESTORS
All the ancestors of a specified member, optionally up to a specified generation or level, including the member itself.	@IANCESTORS
All the children of a specified member, but not the member itself.	@CHILDREN
All the children of a specified member, including the member itself.	@ICHILDREN
A range of members based on the member combination Essbase is currently calculating.	@CURRMBRRANGE
All the descendants of a specified member, optionally up to a specified generation or level, but not the member itself.	@DESCENDANTS
All the descendants of a specified member, optionally up to a specified generation or level, including the member itself.	@IDESCENDANTS
All the members of a specified generation in a specified dimension.	@GENMBRS
All the members of a specified level in a specified dimension.	@LEVMBRS
All the siblings of a specified member, but not the member itself.	@SIBLINGS
All the siblings of a specified member, including the member.	@ISIBLINGS
All the siblings that precede a specified member in the database outline, but not the specified member.	@LSIBLINGS
All the members that match a specified wildcard selection.	@MATCH
All the siblings that precede a specified member in the database outline, including the specified member.	@ILSIBLINGS

To generate a list of members including...	Use this function
All the members of a specified generation or level that are above or below a specified member.	@RELATIVE
All the siblings that follow a specified member in the database outline, but not the specified member.	@RSIBLINGS
All the members that have a common User-Defined Attribute (UDA), defined on the Essbase server.	@UDA
All the siblings that follow a specified member in the database outline, including the specified member.	@IRSIBLINGS

Calculating Statistics

You can calculate most of the standard statistics in your formulas using the following functions.

To calculate...	Use this function
The absolute value of a member or an expression	@ABS
The average value of members or an expression	@AVG
The average value of a member across a range of members	@AVGRANGE
The generation number of the current member combination for a specified dimension	@CURGEN
The level number of the current member combination for a specified dimension	@CURLEV
The factorial of a member or an expression	@FACTORIAL
The next lowest integer value of a member or expression	@INT
The generation number of a specified member	@GEN
The level number of a specified member	@LEV
The maximum value of specified members or an expression	@MAX
The maximum value of a member across a range of members	@MAXRANGE
The minimum value of specified members or an expression	@MIN
The minimum value of a member across a range of members	@MINRANGE
The modulus of a division of two specified members	@MOD
The value of a specified member raised to a specified power	@POWER
The remainder value of an expression	@REMAINDER
A member or expression rounded to a specified number of decimal places	@ROUND

To calculate...	Use this function
The standard deviation of specified members	@STDDEV
The standard deviation of specified members across a range of members	@STDDEVRANGE
The summation of all the values of specified members	@SUM
The summation of all the values of specified members across a range of members	@SUMRANGE
The integer value of an expression, by truncating the value	@TRUNCATE
The variance (difference) between two specified members. See "Calculating a Variance or Percentage Variance Between Actual and Budget Values" on page 25-31.	@VAR
The percentage variance (difference) between two specified members. See "Calculating a Variance or Percentage Variance Between Actual and Budget Values" on page 25-31.	@VARPER
The summation of all the values of specified members across a range of members	@SUMRANGE

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.

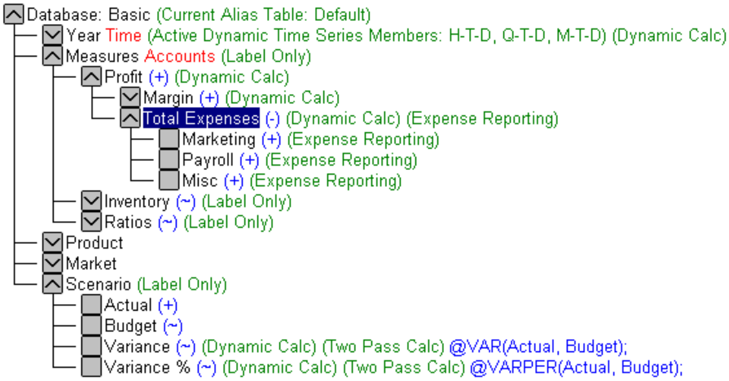
To tell Essbase that a member is an expense item:

1. Select the member in the Application Manager Outline Editor. 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 on the Scenario dimension calculate the variance between the Actual and Budget values.



```
Database: Basic (Current Alias Table: Default)
    Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D, M-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);
```

Figure 25-28, Sample Basic Showing Expense Items

Looking Up Values Based on the Current Member Combination

You can look up specific values based on the member combination that Essbase is currently calculating.

To look up...	Use this function
The ancestors of a specified member combination	@ANCESTVAL
The ancestors of a specified member combination across multiple dimensions	@MDANCESTVAL
The shared ancestors of a specified member combination	@SANCESTVAL
The next or *n*th member in a range of members	@NEXT
The parents of a specified member combination	@PARENTVAL
The parents of a specified member combination across multiple dimensions	@MDPARENTVAL
The shared parents of a specified member combination	@SPARENTVAL
The previous or *n*th previous member in a range of members	@PRIOR
The next or *n*th member in a range of members, retaining all other members identical to the current member	@SHIFT
The next or *n*th member in a range of members, retaining all other members identical to the current member across multiple dimensions	@MDSHIFT

Calculating Financial Functions

You can include financial calculations in your formulas using the following functions.

To calculate...	Use this function
The accumulation of values up to a specified member	@ACCUM
The proceeds of a compounded interest calculation	@COMPOUND
A series of values that represent a compound growth of a specified member across a range of members	@COMPOUNDGROWTH
Depreciation for a specific period using the declining balance method	@DECLINE
The simple interest for a specified member at a specified rate	@INTEREST
The internal rate of return on a cash flow	@IRR
The Net Present Value of an investment based on a series of payments and incomes	@NPV

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 right-angle bracket (>). 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.

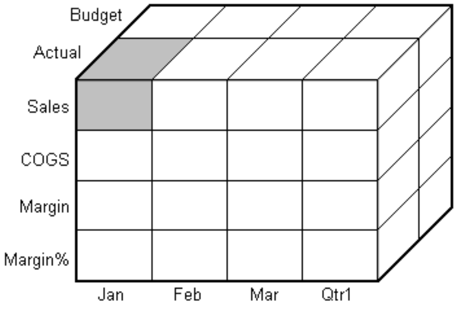


Figure 25-29, Defining a Single Data Value Using the Cross-Dimensional Operator

The following example illustrates how to use the cross-dimensional operator. It allocates miscellaneous expenses to each product in each market.

The total value of Misc_Expenses for all products in all markets is known. The formula allocates a percentage of this total Misc_Expenses value 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 the following calculation:

1. Essbase takes the Sales value for the current member combination and divides it 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 total 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 32, Optimizing Your Calculations.

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

When you run a calc 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 calc script.

For more information on creating and assigning values to substitution variables, see Chapter 7, Creating Applications and Databases.

To use a substitution variable in a calc script, type an ampersand sign (&) followed by variable name. Essbase treats any text string preceded by a & as a substitution variable.

For example, assume that the substitution variable UpToCurr is defined as Jan:Jun. You could use the following @ISMBR function as part of a conditional test in a calc script:

```
@ISMBR(&UpToCurr)
```

Before Essbase runs the calc script, it replaces the substitution variable, as follows:

```
@ISMBR(Jan:Jun)
```

Working with Formulas in Application Partitions

A Hyperion Essbase OLAP Server partitioned application can span multiple servers, processors, or computers. For more information on partitioning, see Chapter 6, Designing Partitioned Applications, and Chapter 15, Building and Maintaining Partitions.

You can use formulas in partitioning, in just the same way as you would 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 your overall database calculation. For more information, see the information on writing calc scripts for application partitions in “Writing Calc Scripts for Application Partitions” on page 30-51.

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 calc scripts, see Chapter 31, Examples of Calc Scripts.

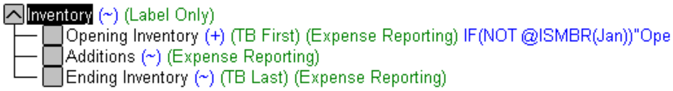
This chapter includes the following examples:

- "Calculating Period-to-date Values" on page 26-1
- "Calculating a Rolling Average Value" on page 26-3
- "Calculating Monthly Asset Movements" on page 26-4
- "Testing for #MISSING Values" on page 26-5

Calculating Period-to-date Values

You can calculate period-to-date values using the @PTD function. You can also calculate period-to-date values using Dynamic Time Series members. For detailed information, see Chapter 29, Calculating Time Series Data.

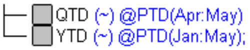
For example, the following shows the Inventory branch of the Measures dimension from the Sample Basic database.



```
Inventory (~) (Label Only)
  Opening Inventory (+) (TB First) (Expense Reporting) IF(NOT @ISMBR(Jan))"Ope
  Additions (~) (Expense Reporting)
  Ending Inventory (~) (TB Last) (Expense Reporting)
```

Figure 26-1, Inventory Branch from Sample Basic Outline

To calculate period-to-date values for the year and for the current quarter, you add two members to the Year dimension, QTD and YTD:



QTD (~) @PTD(Apr:May)
YTD (~) @PTD(Jan:May);

Figure 26-2, Calculating Period-to-Date Values

Assuming the current month is May, the formula on the QTD member is:

```
@PTD(Apr:May);
```

and the formula on the YTD member is:

```
@PTD(Jan:May);
```

Hyperion Essbase sums the values for the range of months when appropriate. However, Opening Inventory has a FIRST Accounts tag and Ending Inventory has a LAST Accounts tag. Essbase takes these values and treats them accordingly. For more information on Accounts tags, see Chapter 29, Calculating Time Series Data.

The following table shows the 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 Accounts 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 Accounts 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 a Rolling Average Value

You can calculate rolling averages using the @AVGRANGE function and year-to-date values using the @ACCUM function.

For example, a database contains monthly Sales data values. You add two members, AVG_Sales and YTD_Sales.

The formula on the AVG_Sales member is:

```
@AVGRANGE(SKIPNONE, Sales, @CURRMBRRANGE(Year, LEV, 0, , 0));
```

and the formula on the YTD_Sales member is:

```
@ACCUM(Sales);
```

Essbase calculates the average Sales values across the months in the Time dimension. The SKIPNONE parameter means that all values are included, even #MISSING values. For more information on #MISSING values, see Chapter 32, Optimizing Your Calculations. Essbase places the results in AVG_Sales.

Essbase calculates the cumulative Sales values and places the results in YTD_Sales.

The following table shows the results:

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 calculate values based on a previous month's value using the @PRIOR function.

For example, a database contains assets data values, which are stored on a month-by-month basis. You can calculate the difference between the assets values each month (the asset movement), by subtracting the previous month's value.

The following example shows three members: Assets for the monthly asset values, Asset_MVNT for the asset movement values, and Opening_Balance for the opening asset value at the beginning of the year.

For Jan, the Asset_MVNT value is calculated by subtracting the Opening_Balance value.

The formula on the Asset_MVNT member is:

```
IF(@ISMBR(Jan) Asset_MVNT = Assets - Opening_Balance;
ELSE Asset_MVNT = Assets - @PRIOR(Assets);
ENDIF
```

The following table shows the results:

Assets/Time	**Opening_Balance**	**Jan**	**Feb**	**Mar**
Assets	1200	1400	1300	1800
Asset_MVNT		**200**	**-100**	**500**

Essbase cycles through the months, performing the following calculations:

1. The IF statement and @ISMBR function check if the current member on the Year dimension is Jan. This 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 your database. For more information on #MISSING values, see Chapter 32, Optimizing Your Calculations.

Assume that your 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. The following formula calculates Commission when applied to a Commission member in the database outline:

```
IF(Sales < > #MISSING) Commission = Sales * .1;
ELSE Commission = #MISSING;
ENDIF
```

If you place the formula in a calc 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.

Chapter 27 Defining the Calculation Order

This chapter describes the order in which Hyperion Essbase calculates your database. If you use Dynamic Calculations, see Chapter 28, Dynamically Calculating Data Values, for information on the calculation order for the dynamically-calculated values.

The following information assumes that you understand the concepts of data blocks and of sparse and dense dimensions. It also assumes that you understand the use of levels and generations. For more information, see Part I, Designing Hyperion Essbase Applications.

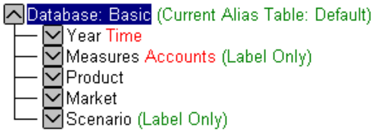
This chapter includes the following sections:

Storing Data 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 that unique combination of sparse dimension members.

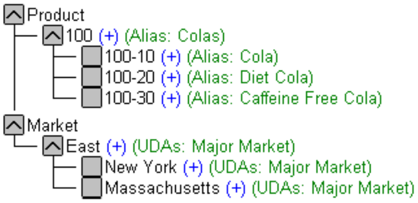
In the Sample Basic database the Year, Measures, and Scenario dimensions are dense. The Product and Market dimensions are sparse.



Database: Basic (Current Alias Table: Default)
Year Time
Measures Accounts (Label Only)
Product
Market
Scenario (Label Only)

Figure 27-1, Dimensions from the Sample Basic Database

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.



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)

Figure 27-2, Product and Market Dimensions from the Sample Basic Database

In Essbase, member combinations are denoted by the cross-dimensional operator. The symbol for the cross-dimensional operator is -> (a hyphen followed by a right-angle bracket). 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 39, Introducing the Essbase Storage Manager.

Member Calculation Order

Essbase calculates your 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. Your database outline tells Essbase how to order the blocks. Within each block, Essbase calculates the values in order according to the hierarchy in your database outline. Therefore, overall, Essbase calculates your database based on your database outline.

When you perform a default calculation (CALC ALL) on your database, Essbase effectively calculates the dimensions in the following 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:

1. The dimension tagged as Accounts
2. The 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)

Otherwise, Essbase calculates:

1. Dense dimensions (in the order they appear in the database outline)
2. Sparse dimensions (in the order they appear in the database outline)

In the Sample Basic database, the dimensions would be calculated in the following order: Measures, Year, Scenario, Product, then Market.

You can override this default order using a calc script. For more information on developing calc scripts, see Chapter 30, Developing Calc 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 the dimension, Level 0 values are calculated first followed by the Level 1, parent value. Then the Level 0 values of the next branch are calculated followed by the Level 1, parent value. The calculation continues in this way until all the levels are calculated.

The following example 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.

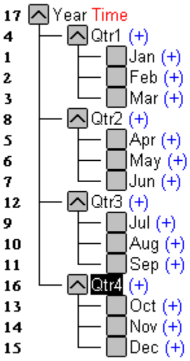


Figure 27-3, Year Dimension from the Sample Basic database

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, which are the other two members in the branch.

Essbase then calculates Qtr1by consolidating Jan, Feb and Mar. In this example the members are added together.

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 the members are added together.

Member Consolidation

You can choose how Essbase consolidates members by applying any unary operator (+, -, /, *, %, ~) to the members in your database outline.

If an Accounts member has a Time Series tag (First, Last, or Average), Essbase consolidates it accordingly. For more information on Time Series, 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 might use a different calculation order. For information on the calculation order for the 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 dimensions in your database outline if:

- You use unary operators to divide (/), multiply (*), or calculate percentages (%) for members in your database outline.
- You place formulas on members in your database outline.

You do not need to consider the calculation order if you use only unary operators to add (+) and subtract (–) members in your database outline, and you do not use formulas in your 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. Formulas 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 27-13.

Using the Unary Operators *, /, and %

If you use unary operators to multiply (*), divide (/), and calculate percentages (%) for members in your 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 27-13.

When you use a multiplication (*), division (/) or percentage (%) operator to consolidate members, carefully order the members in the branch to achieve the required result.

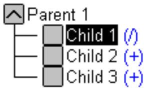


Figure 27-4, Unary Operators in the Database Outline

In the above example, assume that the user wants to divide the total of Child 2 and Child 3 by Child 1. You can achieve this by making Child 1 the last member in the branch.

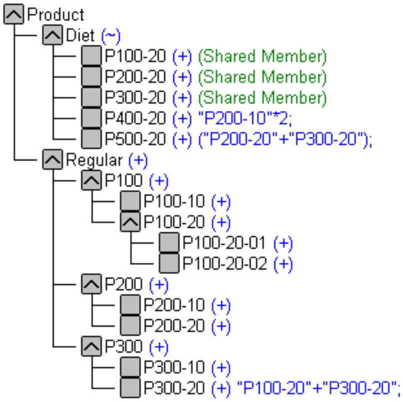
However, if Child 1 is the first member, Essbase starts with Child 1, taking the value of Parent 1 (currently `#MISSING`) and dividing it by Child 1. The result is `#MISSING`. Essbase then adds Child 2 and Child 3. Obviously, this is not the required result. To calculate the correct results, make Child 1 the last member in the branch. For more information on `#MISSING` values, see Chapter 32, Optimizing Your Calculations.

You could apply a formula to a member on the database outline to achieve the same result. However, it is far more efficient to use unary operators on members as in the above example.

Avoiding Forward Calculation References

To obtain the calculation results you expect, ensure that your 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 might not produce the required calculation results.

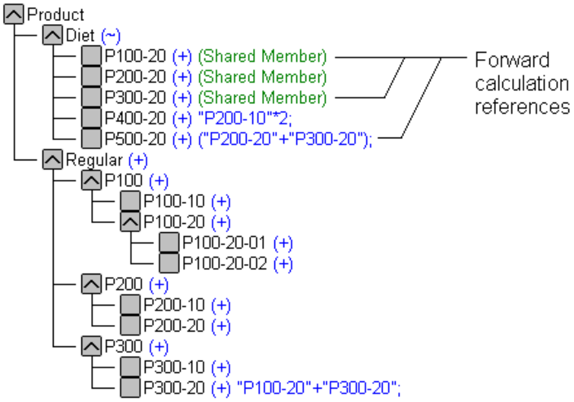
For example, consider the following Product dimension:



```
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";
```

Figure 27-5, Example Product Dimension

This Product dimension has three forward calculation references. Two shared members and one non-shared member have forward calculation references:



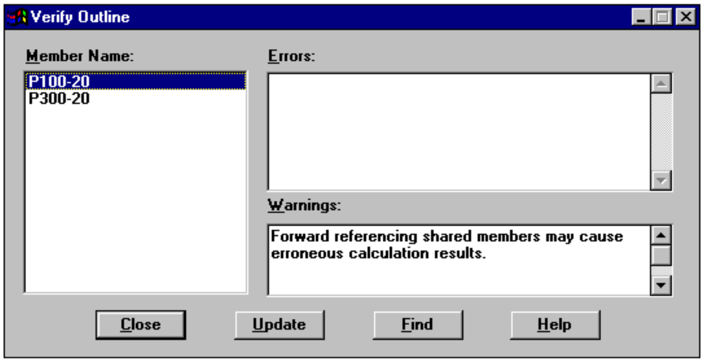
```
Product
  Diet (~)
    P100-20 (+) (Shared Member)
    P200-20 (+) (Shared Member)
    P300-20 (+) (Shared Member)
    P400-20 (+) "P200-10"*2;
    P500-20 (+) ("P200-20"+"P300-20");
  Regular (+)
    P100 (+)
      P100-10 (+)
      P100-20 (+)
        P100-20-01 (+)
        P100-20-02 (+)
    P200 (+)
      P200-10 (+)
      P200-20 (+)
    P300 (+)
      P300-10 (+)
      P300-20 (+) "P100-20"+"P300-20";
```

Forward calculation references

Figure 27-6, Example Product Dimension Showing Forward Calculation References

In the Hyperion Essbase Application Manager Outline Editor, you can choose Outline | Verify to identify shared members with forward calculation references. Essbase displays these members in the **Verify Outline** dialog box.

Note: Choosing Outline | Verify does *not* identify non-shared members that have forward calculation references.



P100-20
P300-20

Forward referencing shared members may cause erroneous calculation results.

Figure 27-7, Verify Outline Dialog Box Showing Forward Calculation References

You can still 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.

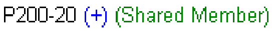
P300-20 (+) (Shared Member)

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");

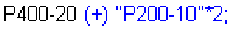
The formula applied to P500-20 references members that Essbase has not yet calculated. One referenced member, P300-20, has its own formula, and Essbase needs to calculate P300-20 before it can accurately calculate P500-20.

The members P200-20 and P400-20 *will* calculate correctly, as they *do not* have forward calculation references:



```
P200-20 (+) (Shared Member)
```

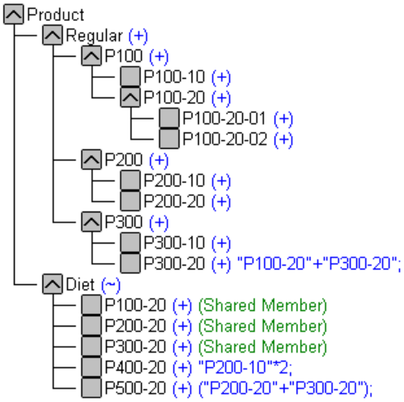
This member is *not* a forward calculation reference, even though Essbase calculates the shared member P200-20 before it calculates the real member P200-20. The real member P200-20 has no calculation dependencies (it has no children and no formula). Therefore Essbase does not need to calculate the real member before the shared member. Essbase simply takes the value of the real member.



```
P400-20 (+) "P200-10"*2;
```

This is *not* a forward calculation reference, even though the formula 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 the first three members, 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. For example,



```
Product
  Regular (+)
    P100 (+)
      P100-10 (+)
      P100-20 (+)
        P100-20-01 (+)
        P100-20-02 (+)
    P200 (+)
      P200-10 (+)
      P200-20 (+)
    P300 (+)
      P300-10 (+)
      P300-20 (+) "P100-20"+"P300-20";
  Diet (~)
    P100-20 (+) (Shared Member)
    P200-20 (+) (Shared Member)
    P300-20 (+) (Shared Member)
    P400-20 (+) "P200-10"*2;
    P500-20 (+) ("P200-20"+"P300-20");
```

Figure 27-8, Changed Product Dimension Without Forward Calculation References

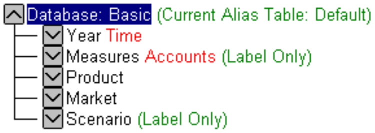
Now Essbase calculates:

- 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 your database outline as a starting point. It defines the sparse member combinations from this first dimension.

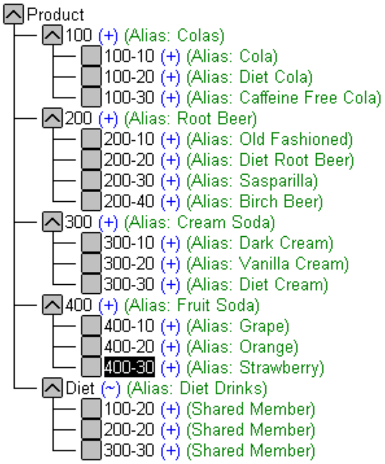
In the Sample Basic database, Product is the first sparse dimension in the database outline.



Database: Basic (Current Alias Table: Default)
Year Time
Measures Accounts (Label Only)
Product
Market
Scenario (Label Only)

Figure 27-9, Dimensions in the Sample Basic Database

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.



```
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)
```

Figure 27-10, Product Dimension from the Sample Basic Database

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.

The following 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.

The following 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 would 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 your 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 it and moves on to the next block. For more information, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

Data Block Renumbering

Essbase renumbers the data blocks when you:

- 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 your database. How you have configured your database defines the member calculation order of dense dimension members *within each block*. It also defines the calculation order of blocks representing sparse dimension members.

Cell Calculation Order Examples

The following examples describe the cell calculation order for different database configurations.

Example 1

Consider the simplest case in which:

- No dimensions have Time or Accounts tags.
- The AGGMISSG setting is turned on so that Essbase aggregates `#MISSING` values.

In the following example, 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. Cells with multiple consolidation paths are 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 27-3, Essbase calculates dense dimensions in the order they appear in the database outline. Assuming the Year dimension appears before the Market dimension in the database outline, the Year dimension is calculated first and the Market dimension last.

The cells are calculated in the following 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 Jan->East, Feb->East and Mar->East. When consolidated on Year, it is an aggregation of Qtr1->New York and Qtr1->Massachusetts.

Essbase knows that Qtr1->East has multiple consolidation paths. Therefore, it calculates Qtr1->East only once using the consolidation path of the dimension calculated last. In the above example, this is Market.

The results are shown in the following table. Qtr1->East has been calculated only once by aggregating the values for Qtr1.

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, 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 calc script to calculate the dimensions in the order you choose. For more information, see Chapter 30, Developing Calc Scripts.

Example 2

Consider a second case in which:

- No dimensions have Time or Accounts tags.
- The AGGMISSG setting is turned off (the default).

Again, in the following example, 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. Cells with multiple consolidation paths are 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 27-3, Essbase calculates dense dimensions in the order they appear in the database outline. Assuming the Year dimension appears before the Market dimension in the database outline, the Year dimension is calculated first and the Market dimension last.

The cells are calculated in the following 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 Jan->East, Feb->East and Mar->East. Second, it is calculated as an aggregation of Qtr1->New York and Qtr1->Massachusetts.

The results are identical to the previous case. However, Qtr1->East has been calculated twice. This is important when you need to load data at parent levels. For more information, see "Example 3" on page 27-16.

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 it is calculated last.

Example 3

Now consider the previous case in which:

- No dimensions have Time or Accounts tags.
- The AGGMISSG setting is turned off (the default).

But also in which

- Data values have been loaded at a parent levels.

Again, in the following example, 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 27-3, Essbase calculates dense dimensions in the order they appear in the database outline. Assuming the Year dimension appears before the Market dimension in the database outline, the Year dimension is calculated first and the Market dimension last.

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 AGGMISSG setting is turned off, Essbase does not aggregate the #MISSING values. This means that, the data loaded at parent levels is not overwritten by the #MISSING values below it.

However, if any of the child data values are not #MISSING, these values are consolidated and overwrite the parent values. For example, if Jan->"New York" contains 50000.00, this value will overwrite the values 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.

The following 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**

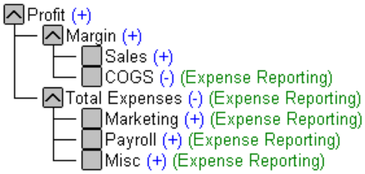
Essbase needs to calculate the Qtr1->East cell twice in order to ensure that a value is calculated for the cell. If it had calculated Qtr1->East according to the last consolidation path only, then the result would have been #MISSING for Qtr1->East, which is not the required result.

Example 4

Now consider a case in which:

- The Year dimension is tagged as Time.
- The Measures dimension is tagged as Accounts.
- The AGGMISSG setting is turned off (the default).

The following 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.



Profit (+)
Margin (+)
Sales (+)
COGS (-) (Expense Reporting)
Total Expenses (-) (Expense Reporting)
Marketing (+) (Expense Reporting)
Payroll (+) (Expense Reporting)
Misc (+) (Expense Reporting)

Figure 27-11, Measures Dimension in Sample Basic Database

Again, the following 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.

Notice that 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
Expenses	**2**	**5**	**8**	**11/16**
Profit	**3**	**6**	**9**	**12/17**

As described in "Member Calculation Order" on page 27-3, 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 AGGMISSG setting is turned off, Essbase does not aggregate the `#MISSING` values. This means that any data loaded at parent levels is not overwritten by the `#MISSING` values. It also means that Essbase calculates the three cells with multiple consolidation paths twice.

The results are shown in the following 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
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 will be 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, the dimension calculated last overwrites previous cell calculations for cells with multiple-consolidation paths. If required, you can use a calc script to change the order in which the dimensions are calculated. For more information, see Chapter 30, Developing Calc Scripts.

For more information on developing formulas, see Chapter 25, Developing Formulas.

Calculation Passes

Whenever possible, Essbase calculates your database in one calculation pass through the database. This means that 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 your 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 your database (CALC ALL), Essbase attempts to calculate your database in one calculation pass. If you have dimensions tagged as Accounts or Time, Essbase might have to do more than one calculation pass through your database.

The following table shows the number of calculation passes Essbase performs if you have dimensions 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/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 tagged as Two Pass, Essbase might need to do an *extra* calculation pass to calculate these formulas. For more information on using Two-Pass calculations, see Chapter 32, Optimizing Your Calculations.

When you use a calc script to calculate your database, the number of calculation passes Essbase needs to perform depends upon your calc script. For more information, see “Calculation Passes” on page 27-20 and Chapter 33, Optimizing Your Calculation Using Intelligent Calculation. For more information on grouping formulas and calculations, see Chapter 32, Optimizing Your Calculations.

When you calculate your database, Essbase automatically displays the calculation order of the dimensions for each pass through the database. This tells you how many times Essbase has cycled through your database during the calculation.

Essbase displays this information in the ESSCMD window and in the Event Log file. To display the Event Log file, choose 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 your database, each block is processed in order, according to its block number.

Essbase calculates the blocks in the following way:

- If you have Intelligent Calculation turned on and if the block does not need to be calculated (it is marked as *clean*), then Essbase skips it and moves on to the next block. For more information, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.
- If the block needs recalculating, Essbase checks 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 "Storing Data in Data Blocks" on page 27-1.
- 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 27-13.
- 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.

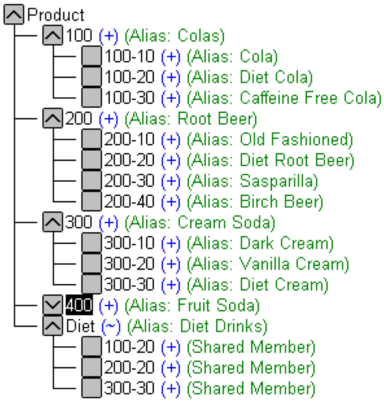
  If the sparse member combination for the block has more than one parent member, Essbase chooses the parent member's dimension that is last in the calculation order. For example, if the block is for Product->East, and you perform a default calculation on the Sample Basic database, Essbase chooses the 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 27-3.

  Based on the chosen sparse dimension, Essbase either aggregates the values or performs a sparse calculation on the data block:

  - If the data block's member on the chosen sparse dimension has a formula applied to it, Essbase performs a formula calculation on the sparse dimension. Essbase evaluates each cell in the data block. The formula affects only this member on the sparse dimension, so the 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 together under Diet. They are also consolidated under their individual product types: Colas, Root Beer and Cream Soda.



```
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)
```

Figure 27-12, Calculating Shared Members

The members under the Diet parent are shared members. For more information on shared members, see Chapter 8, 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 your database including a shared member, Essbase calculates the real member.

Chapter 28 Dynamically Calculating Data Values

This chapter explains how to calculate data values dynamically, and the benefits of doing so. Dynamically calculating some of the values in your database can significantly improve the performance of your overall database calculation.

The information in this chapter assumes that you are familiar with the concepts member combinations, dense and sparse dimensions, and data blocks. For this information, see Part I, Designing Hyperion Essbase Applications.

This chapter includes the following sections:

What Are Dynamic Calculations?

When you design your 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.

In Hyperion Essbase you can define a member to have a *dynamic calculation*. This definition tells Essbase to calculate the data values ‘on-the-fly’ as you request them. This shortens the regular database calculation time, but may increase the retrieval time for dynamically calculated data values.

In Essbase you specify dynamic calculations on a per-member basis. You can define members in your database outline as one of two types of a dynamically calculated member:

- Dynamic Calc
- Dynamic Calc And Store

Understanding Dynamic Calc Members

Essbase does not calculate data values for members tagged as Dynamic Calc during the regular database calculation; for example, during a CALC ALL. Instead Essbase calculates the data values when you retrieve the data; for example, when you retrieve the data into your Hyperion Essbase Spreadsheet Add-in.

Specifically, Essbase dynamically calculates data values when you request the data value by:

- Retrieving the data value into your Spreadsheet Add-in
- Running a Report Script that displays the data value

Essbase does not store the calculated values; it recalculates the values for any subsequent retrieval.

Understanding Dynamic Calc And Store Members

Essbase calculates the data values for the members tagged as Dynamic Calc And Store when you retrieve the data, in the same way as Dynamic Calc members. For Dynamic Calc And Store members, however, Essbase stores the data value. Subsequent retrievals of the same data value do not require recalculation unless Essbase detects that the value needs recalculating.

Ensuring That Essbase Recalculates Updated Values

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 4, Basic Architectural Elements and Chapter 24, Introduction to Database Calculations.

If the data block needs recalculating, Essbase detects this, and places an indicator on the data block when:

- You do a regular (batch) calculation.
- You restructure the database.
- You use the CLEARBLOCK DYNAMIC calculation command. For more information, see the online *Technical Reference* in your `DOCS` directory.

Essbase recalculates the indicated data blocks when you next retrieve the data value.

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 would be recalculated only during the next batch calculation. For example,

- You do a data load, or
- You do a Lock and Send from the Spreadsheet Add-in.

If you load data into the children of a Dynamic Calc And Store member that is a roll-up of its child members, Essbase does *not* know to recalculate the Dynamic Calc And Store member on the next retrieval. The parent member would be recalculated only during the next batch calculation.

After loading data, you need to do a regular batch calculation of your 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 28-17 and the online *Technical Reference* in your `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 first dynamically calculate these child member combinations in order to calculate the parent value. Essbase does not store these child values, even if they are Dynamic Calc And Store members.

For example, assume that Market is a parent member, and 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 these values.

Why Use Dynamic Calculations?

Dynamically calculating some database values can significantly improve the performance of your overall database calculation.

When you tell Essbase to calculate some data values dynamically, you can:

- Reduce the regular calculation time of your database because Essbase has fewer member combinations to calculate.
- Reduce disk usage because Essbase stores fewer calculated data values. The database size and the index size are smaller. For more information on database and index sizes, see Chapter 39, Introducing the Essbase Storage Manager.
- Reduce database restructure time. For example, adding or deleting a Dynamic Calc member on a dense dimension does not change the data block size, and so Essbase does not need to restructure the database for this change. For more information, see “Restructuring Your Database” on page 28-20.
- Reduce the time required to back up the database. Because the database size is smaller, 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 28-8.

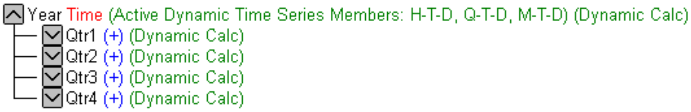
Using Dynamic Calculations

You can tag any member as Dynamic Calc or Dynamic Calc And Store, except:

- 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 balancing your needs for reduced calculation time and disk usage with speed of data retrieval for users. See “Choosing Which Values to Calculate Dynamically” on page 28-5.

In the Hyperion Essbase Application Manager Outline Editor, Essbase indicates which members are Dynamic Calc and Dynamic Calc And Store.



```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D, M-T-D) (Dynamic Calc)
   Qtr1 (+) (Dynamic Calc)
   Qtr2 (+) (Dynamic Calc)
   Qtr3 (+) (Dynamic Calc)
   Qtr4 (+) (Dynamic Calc)
```

Figure 28-1, Sample Basic Outline Showing Dynamic Calc Members

In the Spreadsheet Add-in, users can display visual cues to distinguish dynamically calculated values. For more information, see your *Spreadsheet Add-in User’s Guide*.

Spreadsheet designers may want to develop spreadsheets that include dynamically calculated values in data-less mode, so that Essbase does not dynamically calculate values while they are building the spreadsheet.

Choosing Which Values to Calculate Dynamically

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

- Regular calculation time (batch calculation)
- Low disk space usage
- Reduced database restructure time
- Speed of data retrieval for users
- Reduced backup time

Dynamically calculating some data values decreases the 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.

Note: 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.

Recommendations for Dense Dimension Members

Consider making the following changes to dense dimension members:

- Tag upper-level dense dimension members as Dynamic Calc.
- Try tagging Level 0 dense dimension members with simple formulas as Dynamic Calc, and assess the increase in retrieval time.
- Do *not* tag dense dimension members 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.

Recommendations for Sparse Dimension Members

Consider making the following changes to sparse dimension members:

- Tag some upper-level sparse dimension members that have six or fewer children as Dynamic Calc or Dynamic Calc And Store.
- Tag sparse 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.
- Tag upper-level members on 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 28-11.

Recommendations for Two-Pass Members

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.

Using Dynamic Calculations in a Calc Script

When Essbase calculates, for example, a CALC ALL or CALC DIM statement in a calc script, it bypasses the calculation of Dynamic Calc and Dynamic Calc And Store members.

You cannot do a calc script member calculation of a Dynamic Calc or Dynamic Calc And Store member. If you want to use a calc script to calculate a member explicitly, do not tag the member as Dynamic Calc. For example, the following calc script is valid only if Qtr1 is *not* a Dynamic Calc member:

```
FIX (East, Colas)
Qtr1;
ENDFIX
```

Including Dynamically-Calculated Members in a Formula

You *can* include a dynamically-calculated member in a formula when you apply the formula to your database outline. For example, if Qtr1 is a Dynamic Calc member, you could place the following formula on Qtr1 in your database outline:

```
Qtr1=Jan+Feb;
```

You *cannot* make a dynamically-calculated member the target of a formula calculation in a calc script. This is because 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 calc 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 calc script:

```
Year=Qtr1+Qtr2;
```

because Essbase is not assigning a value to the dynamically-calculated member.

Note: When you reference a dynamically-calculated member in a formula on your database outline or in a calc script, Essbase interrupts the regular calculation to do the dynamic calculation. This can significantly lower calculation performance.

Calculating a Regular Member from Dynamically-Calculated Children

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 applies to sparse dimension and dense dimension members.

For example, in the following outline:

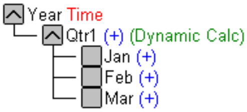


Figure 28-2, Sample Basic Outline Showing Qtr1 as a Dynamic Calc Member

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 roll it up to Year. Essbase calculates Qtr1 even though Qtr1 is a Dynamic Calc member.

Reducing the Impact on Retrieval Time

The increase in retrieval time when you dynamically calculate dense dimension members is not significant unless the member contains a complex formula. See "Choosing Which Values to Calculate Dynamically" on page 28-5.

Note: The increase in retrieval time when you tag sparse dimension members as Dynamic Calc or Dynamic Calc And Store may be significant.

To help you estimate any increase in retrieval time, Essbase calculates a *retrieval factor* for the 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 is the number of data blocks that Essbase has to retrieve from disk or from the database in order to calculate this block. If your database has only dense dimension Dynamic Calc or Dynamic Calc And Store members (no Dynamic Calc or Dynamic Calc And Store members on sparse dimensions), the retrieval factor is 1.

An outline with a high retrieval factor can cause long delays when users retrieve data; for example, an outline with a retrieval factor greater than 2000. 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 might reduce the retrieval time, because the database and index are smaller.

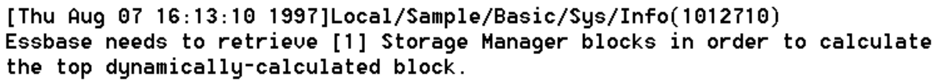
Displaying a Retrieval Factor

When you add Dynamic Calc or Dynamic Calc And Store members to your database outline and save the outline, Essbase calculates an estimated retrieval factor for the most expensive dynamically-calculated data block. Essbase displays the value in the Event log file.

In Application Manager:

1. Select the application and database in the Application Desktop window.
2. Choose Application | View Event Log.
3. In the **View Log File** dialog box, choose **Display All** or **Date**.
4. Click OK. Essbase displays the **Log Viewer** window for the database you selected.

A message similar to the following indicates a retrieval factor.



```
[Thu Aug 07 16:13:10 1997]Local/Sample/Basic/Sys/Info(1012710)
Essbase needs to retrieve [1] Storage Manager blocks in order to calculate
the top dynamically-calculated block.
```

Figure 28-3, Retrieval Factor in the Application Event Log

This message tells you that Essbase needs to retrieve one Essbase Storage Manager (ESM) block in order to calculate the most expensive dynamically calculated data block.

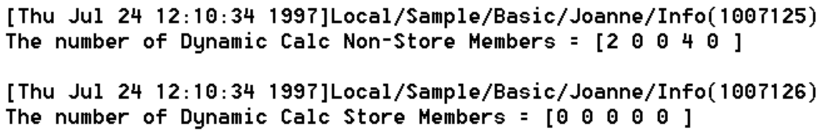
Displaying a Summary of Dynamically-Calculated Members

When you add Dynamic Calc or Dynamic Calc And Store members to your 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 Event log file.

In Application Manager:

1. Select the application and database in the Application Desktop window.
2. Choose Application | View Event Log.
3. In the **View Log File** dialog box, choose **Display All** or **Date**.
4. Click OK. Essbase displays the **Log Viewer** window for the database you selected.

A message similar to the following appears.



```
[Thu Jul 24 12:10:34 1997]Local/Sample/Basic/Joanne/Info(1007125)
The number of Dynamic Calc Non-Store Members = [2 0 0 4 0 ]

[Thu Jul 24 12:10:34 1997]Local/Sample/Basic/Joanne/Info(1007126)
The number of Dynamic Calc Store Members = [0 0 0 0 0 ]
```

Figure 28-4, Event Log Summary of Dynamic Calc and Dynamic Calc And Store Members

This message tells you that there are two Dynamic Calc members on the first dimension in your database outline, and four on the fourth dimension in your database outline. There are no 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 your Hyperion Essbase Spreadsheet Add-in, or using the Report Writer, Essbase uses the Retrieval Buffer to optimize the retrieval. Essbase processes the data in sections. Increasing the Retrieval Buffer size can significantly reduce the retrieval time because Essbase can process larger sections of data at one time.

By default, the Retrieval Buffer is 10KB. However, you can speed up retrieval time if you set the Retrieval Buffer size greater than 10KB.

To set the Retrieval Buffer Size in Application Manager:

1. Select the application and database in the Application Desktop window.
2. Choose Database | Settings. Essbase displays the **Database Settings** dialog box.
3. Type the required size in the **Retrieval Buffer Size** text box.
4. Click OK.

You can use the SETDBSTATEITEM command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

For more information on this setting, see Chapter 37, Optimizing Your Reports.

Choosing Between Dynamic Calc and Dynamic Calc And Store

In most cases you can optimize your calculation 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

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 including a Dynamic Calc member, Essbase calculates only the requested values in a data block. This can be a subset of the data block.

However, when Essbase calculates data values for a member combination including 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. This means that the calculation takes longer and the initial retrieval time is greater.

Essbase stores only the data blocks containing 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.

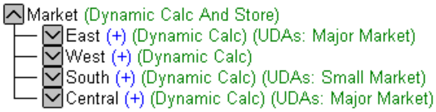
Calculating the intermediate data blocks can significantly increase the initial retrieval time. For example, in the Sample Basic database, Market and Product are the sparse dimensions. Assume that Market and the children of Market are Dynamic Calc And Store members. When a user retrieves the data value 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.

When to Use Dynamic Calc And Store Members Instead of Dynamic Calc Members

Using Dynamic Calc And Store can make the initial retrieval slower, but subsequent retrievals are faster than for Dynamic Calc members. Use Dynamic Calc And Store members instead of Dynamic Calc members 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 application partitions, see "Dynamically Calculating Data in Application Partitions" on page 28-22.
- 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 is a complex formula.
- An upper-level sparse dimension member that users retrieve frequently, making fast retrieval time very important.

For example, in the Sample Basic database, if most users retrieve data at the Market level, you might want to tag Market as Dynamic Calc And Store and its children as Dynamic Calc.



```
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)
```

Figure 28-5, Sample Basic Outline Showing Market as a Dynamic Calc And Store Member

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 the retrieval time, and in the regular calculation time. In addition, the database size (disk usage) does not decrease significantly because Essbase still 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 in the same way.

If many users are concurrently accessing data, you may see significantly lower retrieval times using Dynamic Calc members instead of Dynamic Calc And Store members.

Considering the Effects of Dynamic Calculations

Using dynamically-calculated data values changes the order in which Essbase calculates the values, and can have implications for the way you administer your database.

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 the calculation order, see Chapter 27, Defining the Calculation Order.

During regular calculations, Essbase calculates the database in the following 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 does the sparse calculation based on the Time dimension.
    b. Otherwise, Essbase uses the same dimension as it would for a regular calculation. For more information, see Chapter 27, Defining the Calculation Order.
2. Dense dimensions:
    a. Time series calculations
    b. Dimension tagged as Accounts, if dense
    c. Dimension tagged as Time, if dense
    d. Remaining dense dimensions
    e. Two-pass calculations

Calculation Order for Two-Pass Calculations

Consider the following information to ensure that Essbase produces the required calculation result when dynamically calculating data values for members tagged as Two Pass.

For more information on Two-pass calculations, see Chapter 32, Optimizing Your Calculations.

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, then calculates the Two Pass members in the following order:

1. Two-Pass members on the Accounts dimension, if any exist.
2. Two-Pass members on the Time dimension, if any exist.
3. Two-Pass members on the remaining dense dimensions in the order in which the dimensions appear in the outline.

For example, in the Sample Basic database, assume that:

- Margin% on the dense Measures dimension (the Accounts dimensions) is tagged as both Dynamic Calc and Two Pass.
- Variance on the dense Scenario dimension is tagged as both Dynamic Calc and Two Pass.

Essbase calculates the Accounts dimension member first. So, Essbase calculates Margin% and then calculates Variance.

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 28-13. Essbase then calculates Margin%.

This does not produce the required result because Essbase needs to calculate Margin%->Variance using the formula on Variance, 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.

Calculating Asymmetric Data

Because Essbase uses a different calculation order for dynamic calculations, in some database outlines you may get different calculation results if you tag certain members as Dynamic Calc or Dynamic Calc And Store. This happens when Essbase dynamically calculates asymmetric data calculations.

Symmetric calculations produce the same results whichever dimension is calculated. Asymmetric data calculations calculate differently along different dimensions.

Table 28-1, 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

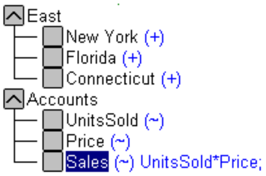
The value for Qtr1->Profit produces the same result whether you calculate it along the Time dimension or the Accounts dimension. Calculating along the Time dimension, you add the values for Jan, Feb and Mar: 50+100+150=300. Calculating along the Accounts dimension, you subtract the value Qtr1->Sales from Qtr1->COGS: 600–300=300.

Table 28-2, Example of an Asymmetric Calculation

Accounts	New York	Florida	Connecticut	East
UnitsSold	10	20	20	50
Price	5	5	5	15
Sales (Price*UnitsSold)	50	100	100	250

The value for East->Sales produces the correct result when you calculate it along the Market dimension, but an incorrect result when you calculate it along the Accounts dimension. Calculating along the Market dimension, you add the values for New York, Florida, and Connecticut: 50+100+100=250, which is correct. Calculating along the Accounts dimension, you multiply the value East->Price by the value East->Units: 15*50=750, which is incorrect.

Consider the example of an asymmetric data calculation in Table 28-2. In the following outline, East is a sparse dimension and Accounts is a dense dimension:



```
East
├─ New York (+)
├─ Florida (+)
└─ Connecticut (+)
Accounts
├─ UnitsSold (~)
├─ Price (~)
└─ Sales (~) UnitsSold*Price;
```

Figure 28-6, Outline Showing Calculation Order of Dynamic Calculations

If East and Sales are tagged as Dynamic Calc members, 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 calculates:

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. This produces the correct result.

If East and Sales are Dynamic Calc members, Essbase calculates:

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. This produces an incorrect result.

To avoid this problem, and ensure that you get the required results, do not tag the Sales member as Dynamic Calc or Dynamic Calc And Store.

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 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 do this, after loading data, you need to ensure that Essbase recalculates Dynamic Calc And Store members. To do this, see "Ensuring That Essbase Recalculates Updated Values" on page 28-3.

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 online *Technical Reference* in your `DOCS` directory.

Including Dynamic Calc and Dynamic Calc And Store Members in Calc Scripts

When calculating your 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 calc script. For more information, see "Using Dynamic Calculations in a Calc Script" on page 28-7.

Creating Dynamic Calc and Dynamic Calc And Store Members

In the 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 attributes to indicate Dynamic Calc and Dynamic Calc And Store members.

Tagging a Member as Dynamic Calc

In Application Manager:

1. Open the database in the Outline Editor.
2. Select the member that you want to tag as Dynamic Calc.
3. Click the button. Alternatively, you can choose Edit | Attributes from the Outline Editor menu, and select **Dynamic Calc** in the **Data Storage** group of the **Member Specification** dialog box.

   Essbase labels the member as Dynamic Calc.

Tagging a Member as Dynamic Calc And Store

In Application Manager:

1. Open the database in the Outline Editor.
2. Select the member that you want to tag as Dynamic Calc And Store.
3. Click the button. Alternatively, you can choose Edit | Attributes from the Outline Editor menu, and select **Dynamic Calc And Store** in the **Data Storage** group of the **Member Specification** dialog box.

   Essbase labels the member as Dynamic Calc And Store.

Removing a Dynamic Calc or Dynamic Calc And Store Tag

In Application Manager:

1. Open the database in the Outline Editor.
2. Select the member from which you want to remove the Dynamic Calc or Dynamic Calc And Store tag.
3. Click the button. Alternatively, you can choose Edit | Attributes from the Outline Editor menu, and select **Store Data** in the Data Storage group of the **Member Specification** dialog box.

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 attribute X for Dynamic Calc and V for Dynamic Calc And Store. For more information, see Chapter 12, Introducing Dynamic Dimension Building.

Restructuring Your 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 data block 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 data blocks. For more information on the database index, see Chapter 39, Introducing the Essbase Storage Manager.

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 your database. Essbase has only to save the database outline, which is very fast.

Essbase does *not* restructure the database or change the index when you:

- 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 your 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:

- 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:

- Add, delete, or move a dense dimension Dynamic Calc And Store member. (But Essbase does *not* restructure the database if you 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 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 Chapter 39, Introducing the Essbase Storage Manager.

Dynamically Calculating Data in Application Partitions

You can define Dynamic Calc and Dynamic Calc And Store members on transparent, replicated, or linked regions of your application partitions. For more information on application partitions, see Chapter 6, 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. Essbase needs to retrieve the child values from the other database, which increases retrieval time. 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. This is because 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 use Time Series tags to calculate account reporting values. 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" on page 29-1
- "Calculating Period-to-Date Values" on page 29-6
- "Calculating Time Series Data in Application Partitions" on page 29-14

Calculating First, Last, or Average Values

Using Time Series and Accounts tags, you can tell Hyperion Essbase how to calculate your accounts data.

Essbase usually calculates a parent in the Time dimension by consolidating or calculating the formulas on the parent's children. However, you can use Time Series tags to consolidate a different value. For example, if you tag an Accounts dimension parent member as First, then Essbase calculates the member by consolidating the value of the member's first child. For example, in the Sample Basic database, if you tag Qtr1 as First, then Essbase consolidates values for Jan to Qtr1. This is useful, for example, for inventory tracking.

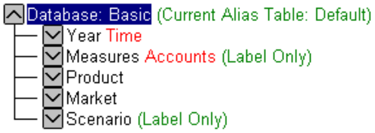
To use Time Series tags, you must have a dimension tagged as Accounts and a dimension tagged as Time. You use the First, Last, Average, and Expense tags only on members of a dimension tagged as Accounts.

The dimensions you tag as Time and Accounts can be either dense or sparse dimensions.

Note: If you are using Intelligent Calculation, changing a Time Series attribute in your database outline does not cause Essbase to restructure the database. You may have explicitly to tell Essbase to recalculate the required data values. See Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

Specifying Accounts and Time Dimensions

When you tag a dimension as Accounts, Essbase knows that this dimension contains members with Accounts tags. When you tag a dimension as Time, Essbase knows that this is the dimension on which to base the time periods for the Accounts tags.



```
Database: Basic (Current Alias Table: Default)
  Year Time
  Measures Accounts (Label Only)
  Product
  Market
  Scenario (Label Only)
```

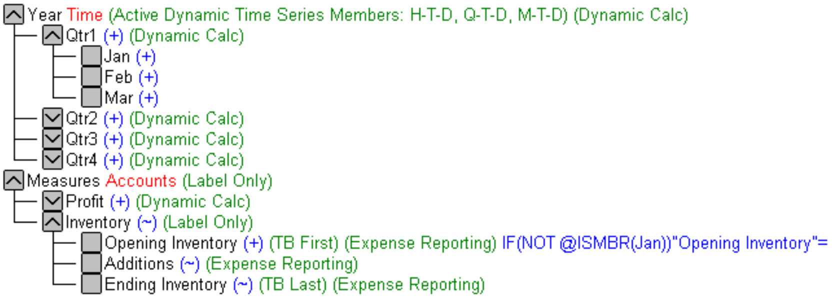
Figure 29-1, Sample Basic Outline Showing Accounts Tag

For information on tagging Accounts and Time dimensions, see Chapter 9, Setting Dimension and Member Attributes.

Reporting the Last Value for Each Time Period

For an Accounts dimension member, you can tell Essbase to consolidate the last value for each time period up to the next level. To do this, tag the member on the Accounts dimension as Last.

For example, in the Sample Basic outline, the accounts value Ending Inventory consolidates the value of the last month in each quarter, and consolidates it up to that month's parent. For example, it consolidates the value for Mar up to Qtr1.



```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D, M-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) IF(NOT @ISMBR(Jan))"Opening Inventory"=
    Additions (~) (Expense Reporting)
    Ending Inventory (~) (TB Last) (Expense Reporting)
```

Figure 29-2, Sample Basic Outline Showing Last Tag

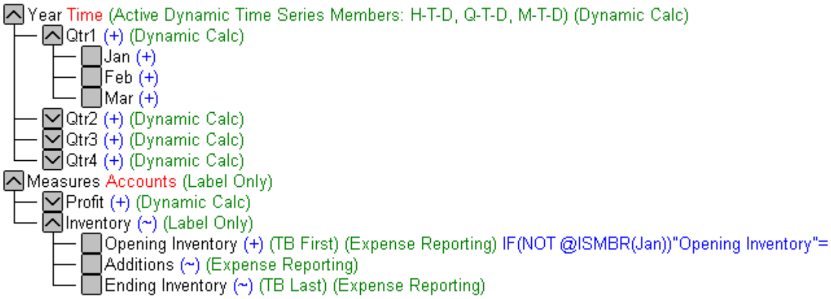
For information on tagging members as Last, see Chapter 9, Setting Dimension and Member Attributes.

By default, Essbase does not skip `#Missing` or zero values when calculating the parent value. You can choose to skip these values. See “Skipping #Missing and Zero Values” on page 29-4.

Reporting the First Value for Each Time Period

For an Accounts dimension member, you can tell Essbase to consolidate the first value for each time period up to the next level. To do this, tag the member on the Accounts dimension as First.

For example, in the Sample Basic outline, the accounts value Opening Inventory consolidates the value of the first month in each quarter, and consolidates it up to that month’s parent. For example, it consolidates the value for Jan up to Qtr1.



```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D, M-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) IF(NOT @ISMBR(Jan))"Opening Inventory"=
        Additions (~) (Expense Reporting)
        Ending Inventory (~) (TB Last) (Expense Reporting)
```

Figure 29-3, Sample Basic Outline Showing First Tag

For information on tagging a member as First, see Chapter 9, Setting Dimension and Member Attributes.

By default, Essbase does not skip `#Missing` or zero values when calculating the parent value. You can choose to skip these values. See “Skipping #Missing and Zero Values” on page 29-4.

Reporting the Average Value for Each Time Period

For an Accounts dimension member, you can tell Essbase to consolidate the average value across each time period up to the next level. For example, you can tell Essbase to consolidate the average of the values for Jan, Feb and Mar to Qtr1. To do this, tag the member on the Accounts dimension as Average.

For information on tagging a member as Average, see Chapter 9, Setting Dimension and Member Attributes.

By default, Essbase does not skip zero (0) values when calculating the parent value. However, if Essbase finds #Missing values when calculating the average, Essbase divides by the number of members that have non-zero values, and skips the #Missing values. You can tell Essbase to skip zero values. See “Skipping #Missing and Zero Values” on page 29-4.

Skipping #Missing and Zero Values

You can tell Essbase how to treat #Missing and zero (0) values when doing Time Series calculations. A #Missing value is a marker in Essbase indicating 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 the parent value, except when Essbase calculates an average value. See “Reporting the Average Value for Each Time Period” on page 29-4.

You can override this default using the following tags:

To...	Use this Tag
Skip #Missing values when calculating the parent value.	**Skip Missing**. In the Outline Editor, select the Accounts member and click the #MI button.
Skip 0 values when calculating the parent value.	**Skip Zeros**. In the Outline Editor, select the Accounts member and click the 0 button.
Skip #Missing values, and 0 values when calculating the parent value.	**Skip Missing** and **Skip Zeros**. In the Outline Editor, select the Accounts member and click the #MI, 0, X button.

For example, if you tag an Accounts dimension member as Last, Skip Missing, then Essbase consolidates the last non-missing child to the parent. For example:

Accounts Member	Jan	Feb	Mar	Qtr1
Member (**Skip Last, Skip Missing**)	60	70	#MI	70

Considering the Effects of First, Last, and Average Tags

The following table shows how Essbase consolidates the Time dimension based on the First, Last, and Average tags on Accounts dimension members.

Accounts Member	Jan	Feb	Mar	Qtr1	
Member1	11	12	13	36	Value of: Jan+Feb+Mar
Member2 (**TB First**)	20	25	21	20	Value of: Jan
Member3 (**TB Last**)	25	21	30	30	Value of: Mar
Member4 (**TB Average**)	20	30	28	26	Average of: Jan, Feb, Mar

Placing Formulas on a Time or Accounts Dimension

If you place a member formula on a Time or Accounts dimension, it may be overwritten by the Time Series calculation.

Consider the following example, in which Opening Inventory is tagged as First:

Measures/Year	Jan	Feb	Mar	Qtr1
Opening Inventory: First	30000	28000	27000	

Because Opening Inventory is tagged as First, Essbase calculates Opening Inventory for Qtr1 by taking the Opening Inventory for Jan value. Any member formula placed on Qtr1 in the database outline is overwritten by this Time Series 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, this is the total of the values for April and May.

In Essbase you can calculate period-to-date values in two ways:

- During the batch calculation, using the @PTD function
- Dynamically, when the user requests the values, using dynamic time series members

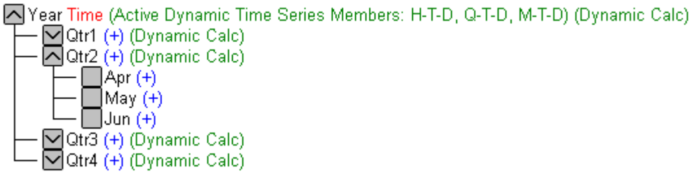
This section covers dynamically calculating period-to-date values using dynamic time series members, which is the most efficient method in almost all cases. For an example of calculating period-to-date values using the @PTD function, see Chapter 26, Examples of Formulas.

About Dynamic Time Series Members

In order to calculate period-to-date values dynamically, you need to create a dynamic time series member for the period on a dimension tagged as Time. See "Specifying Accounts and Time Dimensions" on page 29-2.

You do not create the dynamic time series member directly in your database outline. Instead, you enable predefined dynamic time series members, and associate them with the corresponding generation numbers. When you do this, Essbase creates a dynamic time series member for you.

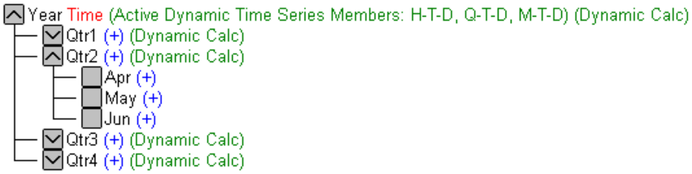
For example, if you want to calculate quarter-to-date values, you enable the Q-T-D member, and associate it with the generation that you want to apply the dynamic time series member to. In Sample Basic, this is generation number 2, which contains the Qtr1, Qtr2, Qtr3, and Qtr4 members. Essbase creates 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 29-8.



```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D, M-T-D) (Dynamic Calc)
  Qtr1 (+) (Dynamic Calc)
  Qtr2 (+) (Dynamic Calc)
    Apr (+)
    May (+)
    Jun (+)
  Qtr3 (+) (Dynamic Calc)
  Qtr4 (+) (Dynamic Calc)
```

Figure 29-4, Sample Basic Outline Showing Time Dimension

Dynamic time series members do not appear as members in your database outline. Instead, Essbase lists the currently active dynamic time series members in a comment on the Time dimension. In this outline, H-T-D (history-to-date), Q-T-D (quarter-to-date) and M-T-D (year-to-date) are active. H-T-D is associated with generation 1; Q-T-D and M-T-D are associated with generation 2.



```
Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D, M-T-D) (Dynamic Calc)
    Qtr1 (+) (Dynamic Calc)
    Qtr2 (+) (Dynamic Calc)
        Apr (+)
        May (+)
        Jun (+)
    Qtr3 (+) (Dynamic Calc)
    Qtr4 (+) (Dynamic Calc)
```

Figure 29-5, Sample Basic Outline Showing Dynamic Time Series

The available predefined dynamic time series members are:

•	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, 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) information, and history-to-date (H-T-D) information, up to a specific year.

If your database tracks data for seasonal time periods, then you might want to report period-to-date (P-T-D) or season-to-date (S-T-D) information.

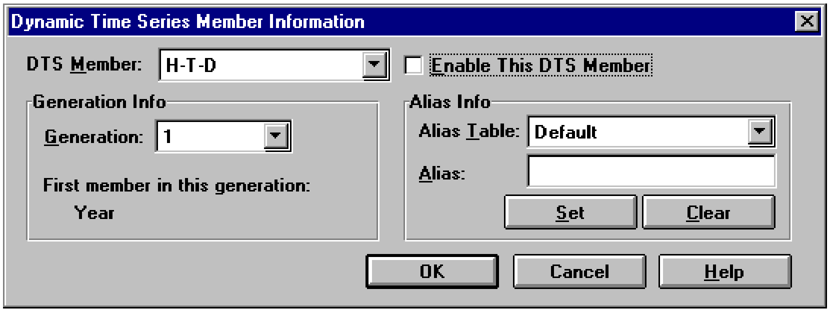
You can associate any dynamic time series member with any generation on the Time dimension except the highest generation number, irrespective of the data. For example, if you choose, you can use the P-T-D member to report quarter-to-date information. You cannot associate dynamic time series members with level 0 members of the Time dimension.

To use these members you need to enable them. If required, you can specify aliases for these time series members. See "Specifying Alias Names for Dynamic Time Series Members" on page 29-11.

Enabling Dynamic Time Series Members

In Hyperion Essbase Application Manager:

1. Open the database in the Outline Editor.
2. Choose Outline | Dynamic Time Series. Essbase displays the **Dynamic Time Series Member Information** dialog box.



Dynamic Time Series Member Information

DTS Member: H-T-D

□ Enable This DTS Member

Generation Info

Generation: 1

First member in this generation:
Year

Alias Info

Alias Table: Default

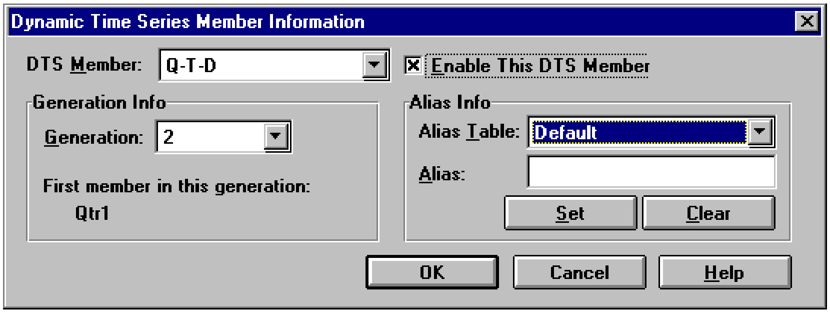
Alias:

Figure 29-6, Dynamic Time Series Member Information Dialog Box for Sample Basic

3. Choose the required dynamic time series member in the **DTS Member** list. For example, choose Q-T-D (quarter-to-date).
4. Choose the corresponding generation number in the **Generation** list. For example, in the Sample Basic database, choose generation 2 to associate the Q-T-D member with that generation.

Note: The number of generations displayed in the **Generation** list depends on how many generations you have in your Time dimension. You cannot associate dynamic time series members with the highest generation (level 0 members) in this dimension.

5. Check **Enable This DTS Member**. In Sample Basic, the DTS member may already be enabled by default.



Dynamic Time Series Member Information

DTS Member: Q-T-D ☒ Enable This DTS Member

Generation Info

Generation: 2

First member in this generation:
Qtr1

Alias Info

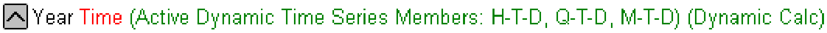
Alias Table: Default

Alias:

Figure 29-7, Dynamic Time Series Member Information Dialog Box Showing Q-T-D on Sample Basic

6. If desired, you can create an alias name for the DTS member. See “Specifying Alias Names for Dynamic Time Series Members” on page 29-11.

7. Click OK to enable the dynamic time series member. In the database outline, Essbase adds a comment to the dimension member of the dimension tagged as Time. The following shows the Sample Basic database with H-T-D, Q-T-D, and M-T-D defined.



Year Time (Active Dynamic Time Series Members: H-T-D, Q-T-D, M-T-D) (Dynamic Calc)

Figure 29-8, Sample Basic Outline Showing Dynamic Time Series Members

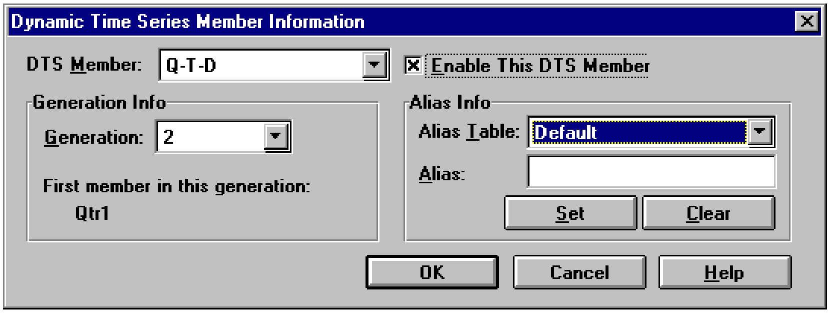
Disabling Dynamic Time Series Members

To disable a Dynamic Time Series member, you tell Essbase not to use the predefined member. In Application Manager:

1. Open the database in the Outline Editor.

2. Choose Outline | Dynamic Time Series. Essbase displays the **Dynamic Time Series Member Information** dialog box.

3. In the **DTS Member** list, choose the dynamic time series member you want to disable.



Dynamic Time Series Member Information

DTS Member: Q-T-D

☒ Enable This DTS Member

Generation Info

Generation: 2

First member in this generation:
Qtr1

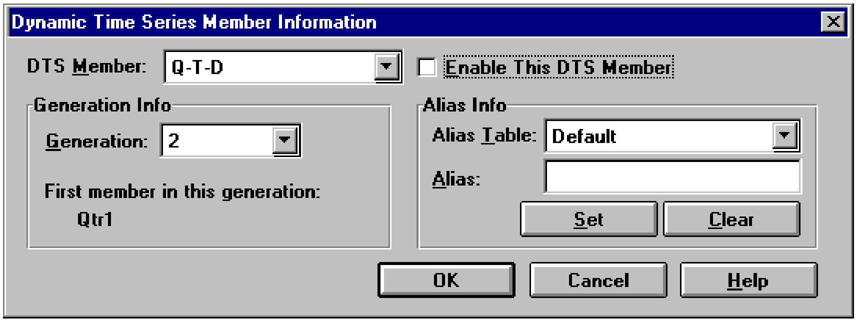
Alias Info

Alias Table: Default

Alias:

Figure 29-9, Dynamic Time Series Member Information Dialog Box Showing Enabled Q-T-D

4. Uncheck **Enable This DTS Member**.



Dynamic Time Series Member Information

DTS Member: Q-T-D

☐ Enable This DTS Member

Generation Info

Generation: 2

First member in this generation:
Qtr1

Alias Info

Alias Table: Default

Alias:

Figure 29-10, Dynamic Time Series Member Information Dialog Box Showing Q-T-D Disabled

5. Click OK.

Specifying Alias Names for Dynamic Time Series Members

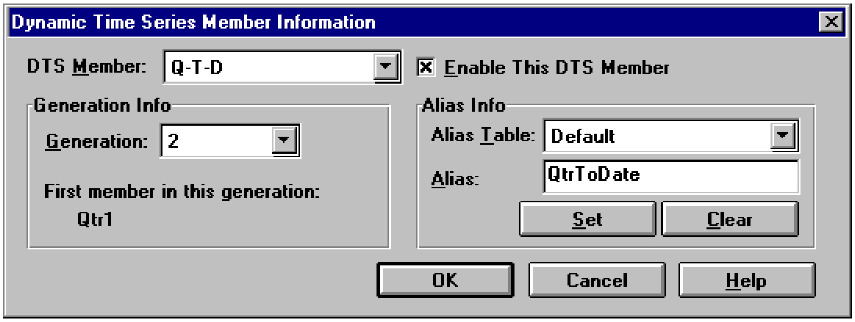
You can specify alias names for the predefined time series members. You can then use these alias names to retrieve the dynamic time series members in your Hyperion Essbase Spreadsheet Add-in or report.

You can create up to 8 different alias names for each dynamic time series member. Essbase saves each alias name in the dynamic time series alias table you specify.

To specify an alias name for a dynamic time series member:

1. Enable the DTS member in the **Dynamic Time Series Member Information** dialog box. See "Enabling Dynamic Time Series Members" on page 29-8.
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 this DTS member.

   For example, type `QtrToDate`.



Dynamic Time Series Member Information

DTS Member: Q-T-D

☒ Enable This DTS Member

Generation Info

Generation: 2

First member in this generation: Qtr1

Alias Info

Alias Table: Default

Alias: QtrToDate

Figure 29-11, Dynamic Time Series Member Information Dialog Box Showing Q-T-D Alias

4. Click the **Set** button.
5. Click OK to save the alias name to the chosen alias table.

For more information on specifying and displaying alias names, see Chapter 8, Creating and Changing Database Outlines.

Generation Names for 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 appear in the **Generation and Level Names** dialog box. To open this dialog box, choose Outline | Gen/Level Names in the Application Manager Outline Editor. For more information on creating generation names, see Chapter 8, Creating and Changing Database Outlines.

This table shows the dynamic time series members and 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 do create a generation name on the Time dimension with one of these predefined generation names, 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, which 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

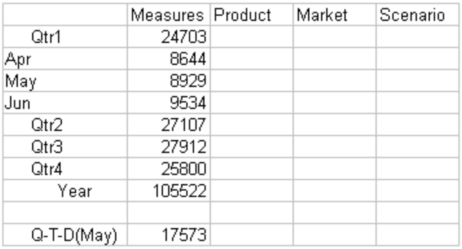
When you retrieve dynamic time series members, you need to tell Essbase the time period up to which you want to calculate the period-to-date value. This is known as the *latest time period* and must be a Level 0 member on the Time dimension.

To specify the latest time period:

- For a specific member, in your Spreadsheet Add-in specify the latest period member name after the dynamic time series member or alias name. For example, Q-T-D(May), which returns the quarter-to-date value by adding values for April and May.
- For a retrieval,
    - Use the <LATEST command in the Report Writer. For more information, see Part VI, Reporting on Your Data.
    - Specify the **Latest Time Series** option in the Spreadsheet Add-in **Options** dialog box. For more information, see your *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 settings.

The following example shows Sample Basic data. Q-T-D(May) displays the period-to-date value for May. By adding the values for Apr and May (8644+8929= 17573):



	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			

Figure 29-12, Spreadsheet Add-in Showing Period-To-Date Value for May

Calculating Time Series Data in Application Partitions

For more information on partitioning, see Chapter 6, Designing Partitioned Applications.

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 times 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 the Dynamic Time Series members are *not* part of the shared area between databases, Essbase gets the data from the source database. You do not need to define the Dynamic Time Series members in both databases. However, this is generally less efficient than including the Dynamic Time Series member in the partition definition.

Chapter 30 Developing Calc Scripts

This chapter explains how to develop and use calc scripts to control how Hyperion Essbase calculates your database. It provides some examples of calc scripts, which you may want to adapt for your own use. For more examples, see Chapter 31, Examples of Calc Scripts.

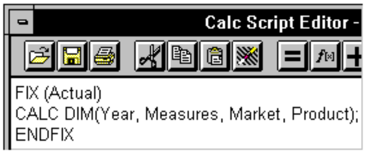
This chapter includes the following sections:

For information on developing formulas, see Chapter 25, Developing Formulas.

What Is a Calc Script?

A calc script contains a series of calculation commands, equations, and formulas. You use a calc script to define calculations other than those of your default database outline.

For example, the following calc script calculates the Actual values in the Sample Basic database.



```
FIX (Actual)
CALC DIM(Year, Measures, Market, Product);
ENDFIX
```

Figure 30-1, Calc Script Editor

You can use a calc script to specify exactly how you want Essbase to calculate your database. For example, you can calculate part of your database or copy data values between members. You can quickly design and run custom database calculations by separating calculation logic from your default database outline.

Why Use a Calc Script?

For most database calculations, a default calculation provides the required results. However, in certain cases, you may need to write a calc script to control how Essbase calculates your database.

For example, you need to write a calc script if you want to:

- Calculate a subset of your database using the FIX command. For more information, see "Calculating a Subset of Your Database" on page 30-48, and the online *Technical Reference* in your `DOCS` directory.
- Change the calculation order of the dense and sparse dimensions in your database.
- Perform a complex calculation in a specific order or one that requires multiple iterations through the data. For example, some Two-Pass calculations require a calc script. For more information, see Chapter 32, Optimizing Your Calculations.
- Perform any Two-Pass calculation on a dimension without an Accounts tag. For more information, see Chapter 32, Optimizing Your Calculations.

- Perform a currency conversion. For more information, see Part VIII, Designing and Building Currency Applications.
- Calculate member formulas that differ from those in the database outline. Formulas in a calc script override those in the database outline.
- Use an API interface to create a custom calculation dynamically.
- Use logic in your 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 30-10, and the online *Technical Reference* in your `DOCS` directory.
- Clear or copy data from specific members. For more information, see "Clearing Data" on page 30-47, "Copying Data" on page 30-48, and the online *Technical Reference* in your `DOCS` directory.
- Define temporary variables for use in your database calculation. For more information, see "Declaring Data Variables" on page 30-10, and the online *Technical Reference* in your `DOCS` directory.
- Force a recalculation of data blocks after you have changed a formula or Time Series attribute on the database outline. For more information, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.
- Control how Essbase uses the Intelligent Calculation feature when calculating your database. For more information, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

Example of Creating a Calc Script

This section provides a step-by-step example of creating and saving a calc script.

For detailed information on creating formulas and obtaining the required calculation results, consider all the information in Part V, Calculating Your Data.

This example is based on the Sample Basic database, which is supplied with your Essbase server installation. This example increases all Budget values by 5%. The example assumes that you have the Hyperion Essbase Spreadsheet Add-in installed on your machine.

Creating This Example Formula

1. Start Hyperion Essbase Application Manager and connect to your Essbase server.
2. Select the Sample application and the Basic database, and click the **Calc Scripts** button, .

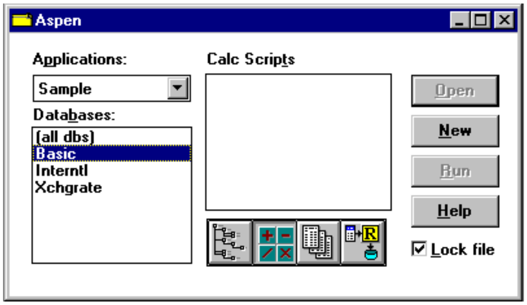


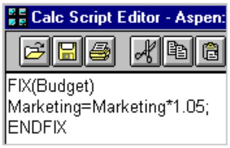
Figure 30-2, Application Desktop Window

If you do not have Sample Basic installed, contact your Essbase administrator.

If another user has Sample Basic open and locked, you can uncheck **Lock file** in the bottom right corner of the Application Desktop window. However, if you do this, you will not be able to save your work.

3. Click the **New** button to open the Calc Script Editor.
4. Type the following calc script. This increases the Budget Marketing expenses values by 5%.

```
FIX(Budget)
Marketing=Marketing*1.05;
ENDFIX
```



```
FIX(Budget)
Marketing=Marketing*1.05;
ENDFIX
```

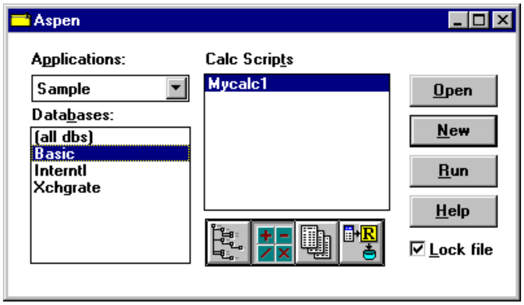
Figure 30-3, Simple Calc Script

For more information on the FIX command, see “Using the FIX Command” on page 30-49, and the online *Technical Reference* in your `DOCS` directory.

5. Click the **Check Syntax** button to verify that the syntax of the formula you have entered is correct. The message `No errors` should appear at the bottom of the Calc Script Editor.

6. Save your calc script by clicking the **Save** button, . Type `Mycalc1` for the calc script object name, and save it on the server (the default).

7. Close the Calc Editor window. Mycalc1 appears in the list of calc scripts for Sample Basic.



Aspen

Applications: Sample

Databases: (all dbs), Basic, Interntl, Xchgrate

Calc Scripts: Mycalc1

☑ Lock file

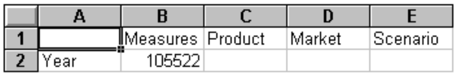
Figure 30-4, Application Desktop Window Showing Calc Script

Calculating Sample Basic

You’re now ready to calculate the Sample Basic database, increasing the Budget values by 5%. However, let’s take a look at the Sample Basic Budget values before we run the calculation.

1. Open the Spreadsheet Add-in and connect to Sample Basic by choosing Essbase | Connect. This example assumes that you have not changed the default Essbase Spreadsheet Add-in Retrieval Options. For more information, see the *Spreadsheet Add-in User’s Guide*.

2. Choose Essbase | Retrieve. Essbase displays the data value for the total of all dimensions.

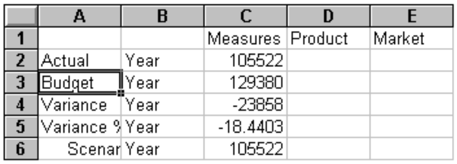


	A	B	C	D	E
1		Measures	Product	Market	Scenario
2	Year	105522			

Figure 30-5, Spreadsheet Add-in Showing Retrieved Data

3. Double-click Scenario to display its members.

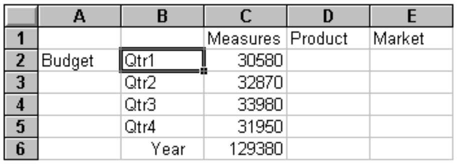
4. Select Budget and choose Essbase | Keep Only. Essbase displays the Budget values.



	A	B	C	D	E
1			Measures	Product	Market
2	Actual	Year	105522		
3	Budget	Year	129380		
4	Variance	Year	-23858		
5	Variance %	Year	-18.4403		
6	Scenar	Year	105522		

Figure 30-6, Spreadsheet Add-in Showing Retrieved Actual and Budget Data

5. Double-click Year. Essbase displays the Budget values for each quarter in the year.

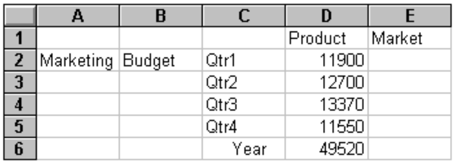


	A	B	C	D	E
1			Measures	Product	Market
2	Budget	Qtr1	30580		
3		Qtr2	32870		
4		Qtr3	33980		
5		Qtr4	31950		
6		Year	129380		

Figure 30-7, Spreadsheet Add-in Showing Retrieved Budget Data

6. Double-click Measures, then Profit, and then Total Expenses to display the Marketing member.

7. Select Marketing and choose Essbase | Keep Only. Essbase displays the Budget Marketing values. These are the values that will increase by 5%.



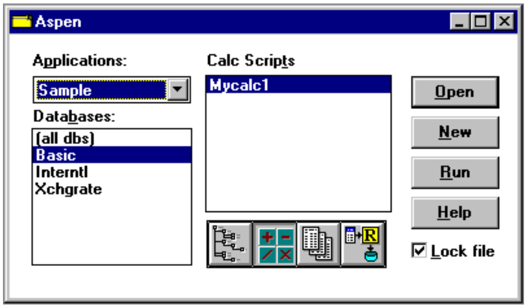
	A	B	C	D	E
1				Product	Market
2	Marketing	Budget	Qtr1	11900	
3			Qtr2	12700	
4			Qtr3	13370	
5			Qtr4	11550	
6			Year	49520	

Figure 30-8, Spreadsheet Add-in Showing Retrieved Data

Running the Calc Script

Now you are ready to run your Mycalc1 calc script, which increases these values by 5%.

1. Minimize, but do not close the Spreadsheet Add-in window.
2. In Application Manager, connect to your Essbase server, if you are not already connected.
3. Select the Sample application and the Basic database, and click the **Calc Scripts** button .



Applications: Sample

Databases: (all dbs) Basic Interntl Xchgrate

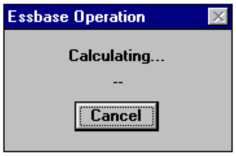
Calc Scripts: Mycalc1

☑ Lock file

Figure 30-9, Application Desktop Window Showing Calc Script

4. Select Mycalc1 and click **Run**.
5. Essbase prompts you to select a database. Ensure that Sample Basic is selected in the **Select Database** dialog box, and click OK.

   Essbase calculates the database.



Calculating...

--

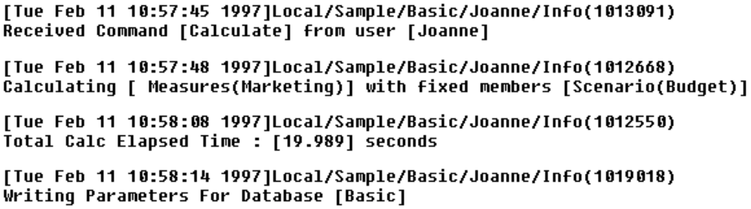
Figure 30-10, Calculating Message Box

Checking the Calculated Dimensions and Calculation Time

When Essbase has finished the calculation, you can check the dimensions calculated, and the calculation time in the Application Event Log.

1. In Application Manager, choose Application | View Event Log, and choose **Date** in the **View Log File** dialog box to view the entries for the current date.

   Essbase displays the Application Event Log. Scroll to the end of the file to see the entries for your calculation. The entries will be similar to the following:



```
[Tue Feb 11 10:57:45 1997]Local/Sample/Basic/Joanne/Info(1013091)
Received Command [Calculate] from user [Joanne]

[Tue Feb 11 10:57:48 1997]Local/Sample/Basic/Joanne/Info(1012668)
Calculating [ Measures(Marketing)] with fixed members [Scenario(Budget)]

[Tue Feb 11 10:58:08 1997]Local/Sample/Basic/Joanne/Info(1012550)
Total Calc Elapsed Time : [19.989] seconds

[Tue Feb 11 10:58:14 1997]Local/Sample/Basic/Joanne/Info(1019018)
Writing Parameters For Database [Basic]
```

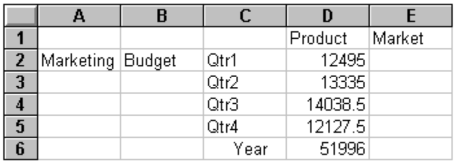
Figure 30-11, Application Event Log Showing Calculation Messages

   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.

   This is the default level of messages that Essbase provides. If required, you can display more detailed calculation messages in the Application Event Log by using the SET MSG command. For more information, see the online *Technical Reference* in your `DOCS` directory.

2. Close the **Application Log Viewer** window.

3. Maximize the Spreadsheet Add-in window. The pre-calculation data values should still be displayed. Choose Essbase | Retrieve to display the new data values. If the pre-calculation data values are not displayed, repeat the steps in “Calculating Sample Basic” on page 30-5 to display the new values.



	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	

Figure 30-12, Spreadsheet Add-in Showing Retrieved Data

   As you can see, the data values have increased by 5%. The calculation is successful. If required, you can reload the default data back into Sample Basic. See Part IV, Loading Data.

Building Calc Scripts in the Calc Script Editor

You use the Calc Script Editor to build calc scripts. You can type the calc scripts directly into the Calc Script Editor text area, or you can use the Calc Script Editor user interface features to build the calc script.

Calc scripts are ASCII text. If required, you can create your calc script in the text editor of your choice, and paste it into the Calc Script Editor.

Essbase provides a flexible set of commands, which you can use to control how your database is calculated.

You can construct calc scripts from commands and formulas. There are several types of commands discussed in the following sections, including:

- Computation
- Control Flow
- Data Declaration
- Global Settings

Calculating Your Database, Dimensions, and Members

You can use the following calculation commands to calculate your database based on the structure and formulas in the database outline.

Note: For a complete list of calculation commands and syntax, see the online *Technical Reference* in your `DOCS` directory.

To calculate...	Use...
The entire database, based on the outline.	CALC ALL
A specified dimension or dimensions.	CALC DIM
All members tagged as TWOPASS 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

To calculate...	Use...
All members tagged as LAST on the dimension tagged as ACCOUNTS. See Chapter 29, Calculating Time Series Data.	CALC LAST
Currency conversions. See Part VIII, Designing and Building Currency 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 online *Technical Reference* in your DOCS directory.

To...	Use...
Calculate a subset of your database.	FIX...ENDFIX
Specify the number of times that commands are iterated.	LOOP...ENDLOOP

You can also use the IF ENDIF functions to specify conditional calculations. See "Controlling the Flow of Calculations" on page 30-10.

Note: You cannot branch from one calc script to another calc script.

Declaring Data Variables

You can use the following commands to declare temporary variables and, if required, set their initial values. *Temporary variables* store the results of intermediate calculations.

You can also use substitution variables in your calc script. See "Using Substitution Variables" on page 30-46.

For detailed information on the these commands, see the online *Technical Reference* in your DOCS directory.

To...	Use...
Declare one-dimensional array variables.	ARRAY
Declare a temporary variable that contains a single value.	VAR

Values stored in temporary variables exist only while the calc script is running. You cannot report on the values of any temporary variables you use.

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 the same time.

```
ARRAY Discount[Scenario]
```

Specifying Global Settings for Your Database Calculation

You can use the following commands to define calculation behavior.

Note: For a complete list of commands, see the online *Technical Reference* in your `DOCS` directory.

To...	Use...
Specify how Essbase treats `#MISSING` values during a calculation.	SET AGGMISSG
Adjust the default Calculator cache size.	SET CACHE
Optimize the calculation of large, flat database outlines. See Chapter 32, Optimizing Your Calculations.	SET CALCHASHTBL
Optimize the calculation of sparse dimension formulas in large database outlines. See Chapter 32, Optimizing Your Calculations.	SET FRMLBOTTOMUP

Display messages to trace your calculation.	SET MSG
	SET NOTICE

Turn on and off Intelligent Calculation. See Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.	SET UPDATECALC
Control how Essbase marks data blocks for the purpose of Intelligent Calculation. See Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.	SET CLEARUPDATESTATUS
Specify the maximum number of blocks that Essbase can lock concurrently when calculating a sparse member formula.	SET LOCKBLOCK
For currency conversions, restrict aggregations to parents that have the same defined currency. See Part VIII, Designing and Building Currency Applications.	SET UPTOLOCAL

SET commands in a calc script are procedural. The first occurrence of a SET command in a calc script stays in effect until the next occurrence of the same SET command.

For example, in the following calc 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.

In this second calc script:

```
SET AGGMISSG ON;
Qtr1;

SET AGGMISSG OFF;
East;
```

Essbase calculates member combinations for Qtr1 with the AGGMISSG (aggregate missing values) setting turned on. Essbase then does a second calculation pass through the database and calculates member combinations for East with the AGGMISSG setting turned off. For more information on the aggregate missing values setting, see the SET AGGMISSG command in the online *Technical Reference* in your DOCS directory. For more information on calculation passes, see Chapter 32, Optimizing Your Calculations.

Adding Comments

You can include comments to annotate your calc scripts. Essbase ignores these comments when it runs the calc script.

To include a comment start it with /* and end it with */. For example:

```
/*     This is a calc script comment
       that spans two lines.*/
```

Calc Script Syntax

When you create a calc script, you need to apply the following rules:

- End each formula or calc script command with a semicolon (;). For example,

  ```
  CALC DIM(Product, Measures);
  ```

- If a member name contains spaces, or is the same as an operator name, enclose the member name in double quotes (""). For example, `"Opening Inventory"`
- 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 a database outline:

  ```
  Commission (IF(Sales < 100) Commission = 0;ENDIF)
  ```

- End each IF statement with an ENDIF statement. The IF and ENDIF statements do not need to be followed by a semicolon (;).
- If you are using an IF... ELSEIF ... statement nested within another IF...ENDIF statement, supply an ENDIF statement for each ELSEIF statement as well as all the IF statements. For more information, see the online *Technical Reference* in your `DOCS` directory.
- 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 calc script, you can check the syntax using the Calc Script Editor syntax checker. For more information, see “Checking Syntax” on page 30-43.

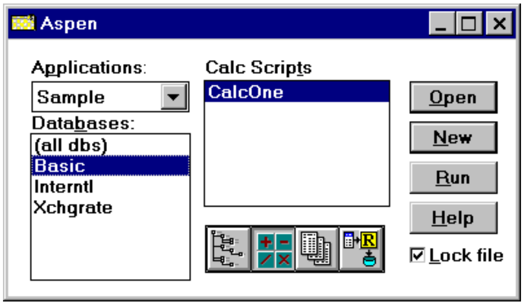
Note: For detailed information on calc script syntax, see the online *Technical Reference* in your `DOCS` directory.

Opening the Calc Script Editor

Note: For information on opening an existing calc script, see “Changing a Calc Script” on page 30-16.

To open the Calc Script Editor:

1. Open Application Manager and connect to your Essbase server.
2. In the Application Desktop Server window, select the required application and database.
3. Click the + − / × button. Essbase displays a list of all the calc scripts associated with the application and database you chose.



Aspen

Applications: Sample

Databases:
(all dbs)
Basic
Interntl
Xchgrate

Calc Scripts: CalcOne

☑ Lock file

Figure 30-13, Application Desktop Window

4. To create a new calc script, click the **New** button. To open an existing calc script, select it in the **Calc Scripts** list, and click the **Open** button.

   Essbase opens the Calc Script Editor:

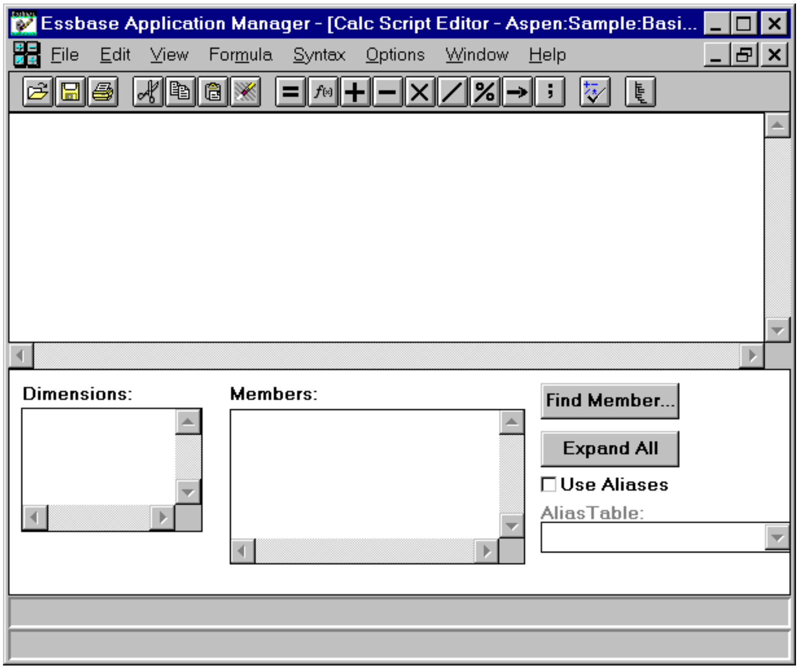


Figure 30-14, Calc Script Editor

Adding a Calc Script

In Application Manager:

1. In the Application Desktop Server window, select the application and database with which you want to associate the new calc script.

2. Click the button and then click the **New** button. Alternatively, you can choose File | New | Calc Script from the Application Manager menu.

   Essbase opens the Calc Script Editor. You can now build your calc script. Essbase prompts you to name your calc script when you save it. See “Saving a Calc Script” on page 30-20.

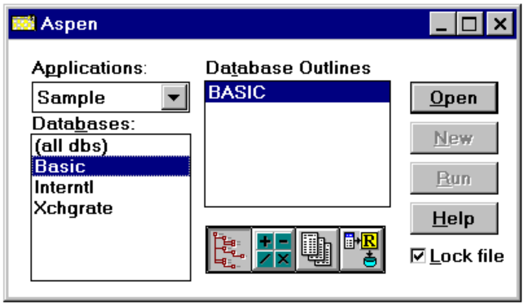
Changing a Calc Script

To change a calc script, open it in the Calc Script Editor. How you do that depends on where the calc script is stored.

If the Calc Script is on the Current Server:

1. In the Application Desktop Server window, select the application and database that contains the calc script.

   The following example shows the Application Desktop Server window for an Essbase server called Aspen. The Sample Basic database is selected.



Applications: Sample

Databases: Basic

Database Outlines: BASIC

☑ Lock file

Figure 30-15, Application Desktop Server Window

2. Click the ⊞ button.

   Essbase displays the calc script files (`.CSC` files) stored in the `\ARBORPATH\APP\appname\dbname` directory on the server machine, where `ARBORPATH` is the directory in which you installed Essbase, and `appname` and `dbname` are the current application and database.

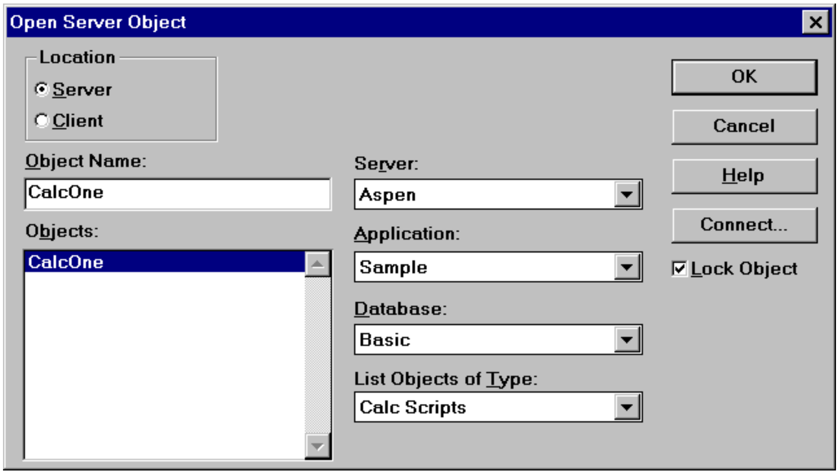
   For example, assuming the Essbase install directory is `C:\ESSBASE`, if you select Sample Basic, Essbase displays the `.CSC` files in the `C:\ESSBASE\APP\SAMPLE\BASIC` directory.

3. Select the required calc script in the **Calc Scripts** list.

4. Click the **Open** button. Essbase opens the Calc Script Editor. You can now edit the calc script.

If the Calc Script is on a Different Server:

1. Ensure that the focus is on the Application Desktop Server window.
2. Choose File | Open from the Application Manager menu. The **Open Server Object** dialog box appears.



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

Figure 30-16, Open Server Object Dialog Box

3. Click the **Connect** button to connect to the other server, and click OK.
4. In the **Open Server Object** dialog box, select the application and database that contain the calc script.
5. Select the required calc script in the **Objects** list, and click OK.

   Essbase displays the calc script in the Calc Script Editor.

If the Calc Script is on your Client Machine:

1. In the Application Desktop Client window, select the application and database that contains the calc script.

2. Click the +−/× button.

   Essbase displays the calc script files (`.CSC` files) stored in the `\`*`ARBORPATH`*`\CLIENT\`*`appname`*`\`*`dbname`* directory on your *client* machine, where *`ARBORPATH`* is the directory in which you installed Essbase, and *`appname`* and *`dbname`* are the current application and database on your client machine.

   For example, assuming the Essbase install directory is `C:\ESSBASE`, if you have an application called MYAPP01 and a database called MYDB01 on your client machine, Essbase displays the `.CSC` files in the `C:\ESSBASE\CLIENT\MYAPP01\MYDB01` directory on your client machine.

   In this example, there are three calc scripts already created for the MYDB01 database.

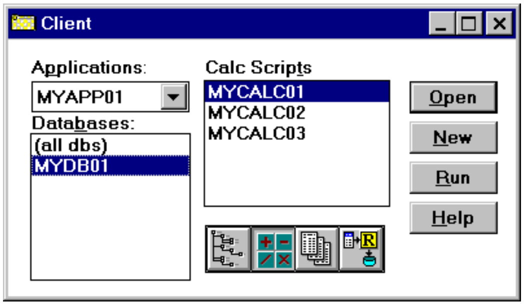


Figure 30-17, Application Desktop Client Window

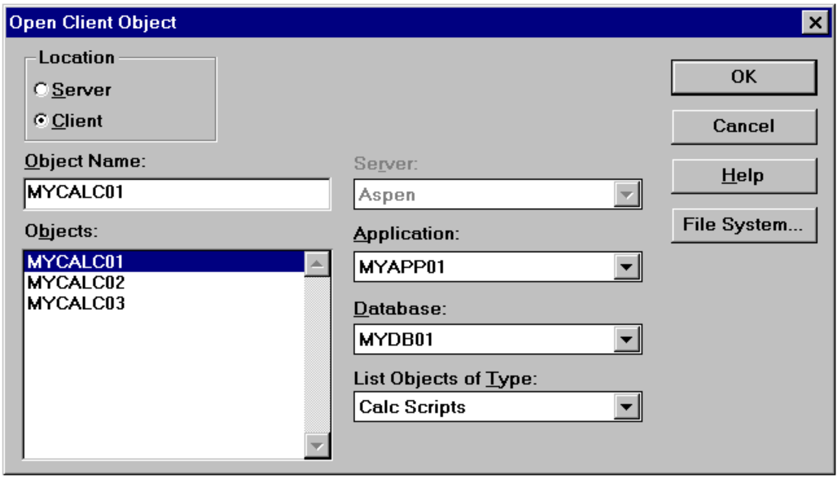
3. Select the required calc script in the **Calc Scripts** list.

4. Click the **Open** button. Essbase opens the Calc Script Editor. You can now edit the calc script.

If the Calc Script is on Your Client Machine, But Not Saved as an Essbase Object:

If the calc script is on your client machine, but not saved as an Essbase object in the `\`*`ARBORPATH`*`\CLIENT\`*`appname`*`\`*`dbname`* directory:

1. Choose File | Open from the Application Manager menu. The **Open Client Object** dialog box appears.



Open Client Object

Location

○ Server

● Client

Object Name: MYCALC01

Objects:

MYCALC01

MYCALC02

MYCALC03

Server: Aspen

Application: MYAPP01

Database: MYDB01

List Objects of Type: Calc Scripts

Figure 30-18, Open Client Object Dialog Box

2. Click the **File System** button to connect to the other server. The **Open Client File** dialog box appears.
3. Select the file that contains the required calc script, and click OK.

   Essbase displays the calc script in the Calc Script Editor.

Saving a Calc Script

You can save a calc script as:

- An object on the Essbase server or on your client machine. If you want other users to have access to the calc script, you need to save it on the Essbase server. You can associate the calc script object with:
    - An application and all the databases within the application. This means that you can run the calc script on any of the databases in the application.
    - A database, which means that you can run the calc script on the database.
- A file on your client machine.

If you want other users to have access to the calc script, you need to save it on the Essbase server. If you save a calc script on your client machine, other users do not have access to the calc script. While you are developing a calc script, you might want to save it on your client machine and then move it to the Essbase server when complete.

When you save a calc script from the Calc Script Editor, by default Essbase associates it with the current application and database.

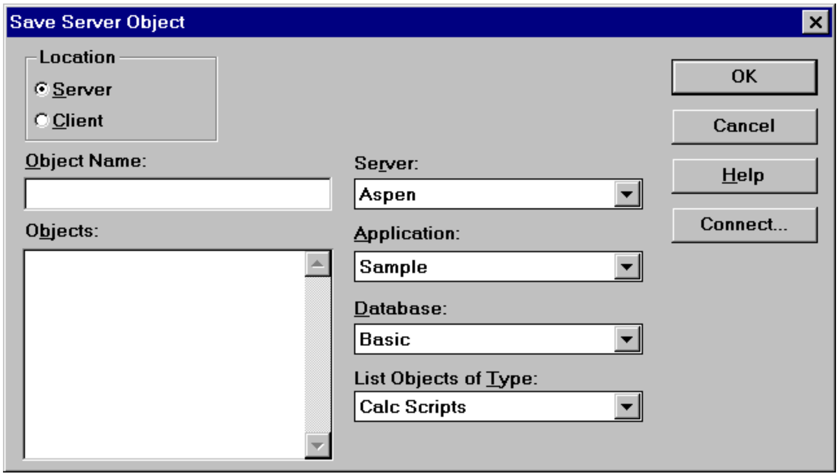
Calc scripts are given a `.CSC` extension by default. If you run a calc script from the Application Manager, or from your Spreadsheet Add-in, it must have a `.CSC` extension. However, a calc script is an ASCII file, and you can run any ASCII file as a calc script using ESSCMD.

A calc script can also be a string defined in memory. You can access this string via the API on the Essbase Client or Essbase Application Server. This means that you can dynamically create a calc script based on user selections from dialog boxes.

Saving a Calc Script as an Object on the Essbase Server

To save a calc script on the Essbase server:

1. In the Calc Script Editor, click the 💾 button. The **Save Server Object** dialog box appears.



Location: ● Server ○ Client

Server: Aspen

Application: Sample

Database: Basic

List Objects of Type: Calc Scripts

Figure 30-19, Save Server Object Dialog Box

2. Associate the calc script with an application or database.

   To associate the calc script with an application and all the databases within the application:

   a. Select the required application in the **Application** list.

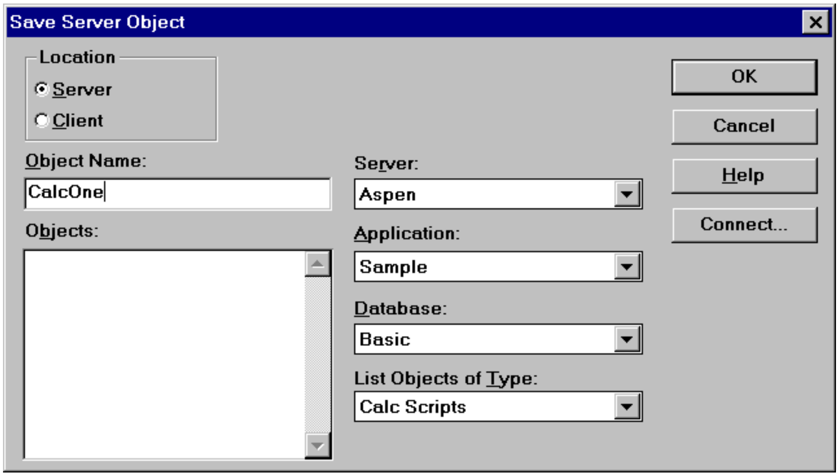
   b. Select **(all dbs)** (all databases) in the **Database** list.

   To associate the calc script with a database:

   a. Select the application containing the database in the **Application** list.

   b. Select the required database in the **Database** list.

3. In the **Object Name** text box, type the name that you want to give the calc script. You can type up to 8 alphanumeric characters.



Save Server Object

Location

◉ Server

○ Client

Object Name: CalcOne

Objects:

Server: Aspen

Application: Sample

Database: Basic

List Objects of Type: Calc Scripts

Figure 30-20, Save Server Object Dialog Box

4. Click OK.

Essbase saves the calc script as an calc script object on the Essbase server.

Calc script objects associated with an application are saved in the \ARBORPATH\APP\appname directory on the Essbase server machine. Calc script objects associated with a database are saved in the \ARBORPATH\APP\appname\dbname directory on the Essbase server machine. *ARBORPATH* is the Essbase install directory, and *appname* and *dbname* are the application and database with which you have associated the calc script.

For example:

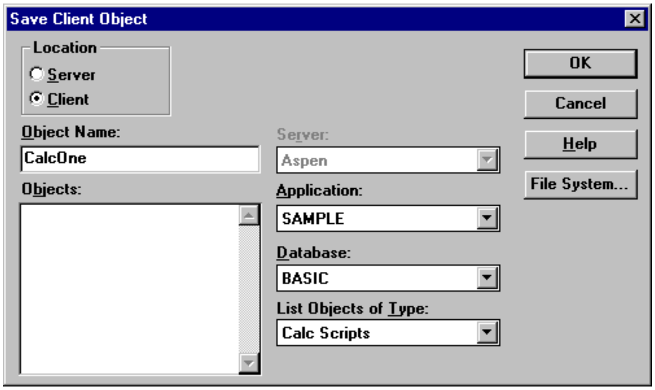
- If you associate a calc script called CalcTwo with the Sample Basic database, assuming the Essbase install directory is `C:\ESSBASE`, Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\APP\SAMPLE\BASIC`.
- If you associate a calc script called CalcTwo with the Sample application, assuming the Essbase install directory is `C:\ESSBASE`, Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\APP\SAMPLE`.

Saving a Calc Script as an Object on Your Client Machine

To save a calc script on your client machine:

1. In the Calc Script Editor, click the 💾 button. If this is a new calc script, either the **Save Server Object** dialog box or the **Save Client Object** dialog box appears depending on whether you opened the Calc Script Editor from the Application Desktop Server or Client window.

   If the **Save Server Object** dialog box appears, select the **Client** option button under **Location**. The **Save Client Object** dialog box replaces the **Save Server 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

Figure 30-21, Save Client Object Dialog Box

2. Associate the calc script with an application or database.

   To associate the calc script with an application on your client machine and all the databases within the application:

   a. Select the required application in the **Application** list.

   b. Select **(all dbs)** (all databases) in the **Database** list.

   To associate the calc script with a database on your client machine:

   a. Select the application containing the database in the **Application** list.

   b. Select the required database in the **Database** list.

3. In the **Object Name** text box, type the name that you want to give the calc script. You can type up to 8 alphanumeric characters.

4. Click OK.

Essbase saves the calc script as an calc script object on your client machine.

Calc script objects associated with an application are saved in the `\`*`ARBORPATH`*`\CLIENT\`*`appname`* directory on the Essbase client machine. Calc script objects associated with a database are saved in the `\`*`ARBORPATH`*`\CLIENT\`*`appname`*`\`*`dbname`* directory on the Essbase client machine. *`ARBORPATH`* is the Essbase install directory, and *`appname`* and *`dbname`* are the application and database with which you have associated the calc script.

For example:

- If you associate a calc script called CalcTwo with a MYDB01 database in the MYAPP01 application, assuming the Essbase install directory is `C:\ESSBASE`, Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\CLIENT\MYAPP01\MYDB01`.
- If you associate a calc script called CalcTwo with the MYAPP01 application, assuming the Essbase install directory is `C:\ESSBASE`, Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\CLIENT\MYAPP01`.

Saving a Calc Script in Your Client Machine's File System

1. In the Calc Script Editor, click the 💾 button.

   If this is a new calc script, either the **Save Server Object** dialog box or the **Save Client Object** dialog box appears depending on whether you opened the Calc Script Editor from the Application Desktop Server or Client window.

   If the **Save Server Object** dialog box appears, select the **Client** option button under **Location**. The **Save Client Object** dialog box replaces the **Save Server Object** dialog box.

2. In the **Save Client Object** dialog box click the **File System** button. The **Save Client File** dialog box appears.

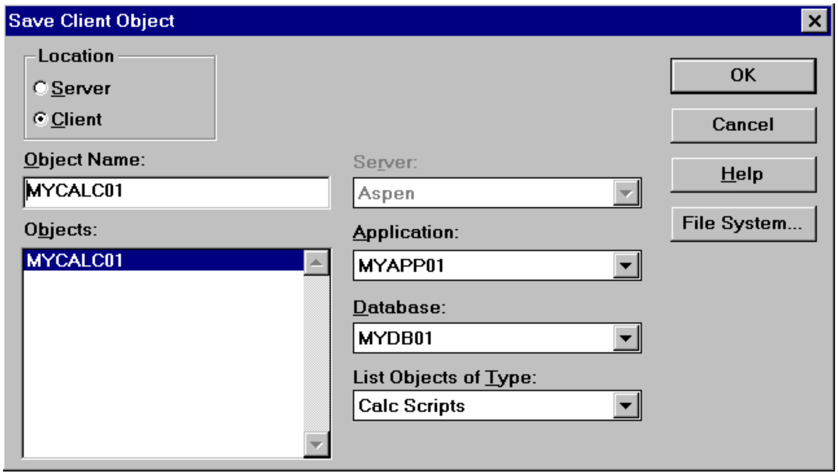
3. Enter the required directory and file name, and click OK.

Essbase saves the calc script in the directory you specified.

Copying a Calc Script from Your Client Machine to the Essbase Server

1. In the Calc Script Editor, open the calc script that you want to copy. For more information, see "Changing a Calc Script" on page 30-16.

2. Choose File | Save As from the Application Manager menu. The **Save Client Object** dialog box appears.



Save Client Object

Location
○ Server
● Client

Object Name: MYCALC01

Objects: MYCALC01

Server: Aspen

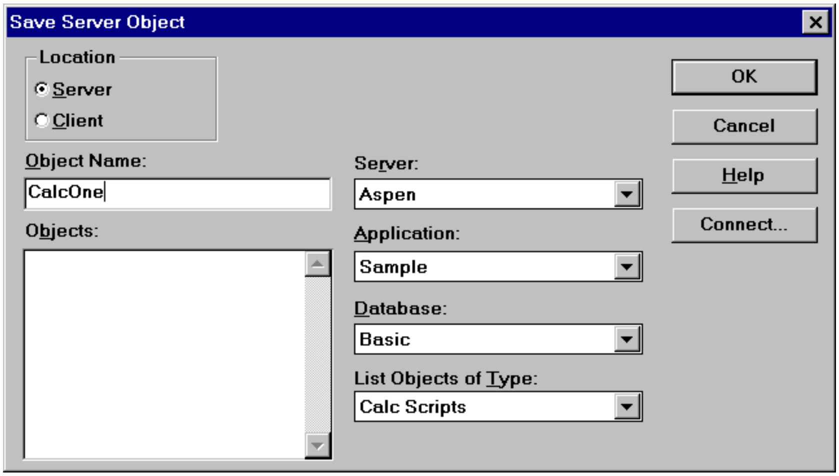
Application: MYAPP01

Database: MYDB01

List Objects of Type: Calc Scripts

Figure 30-22, Save Client Object Dialog Box

3. Under **Location**, select the **Server** option button. The **Save Server Object** dialog box replaces the **Save Client Object** dialog box.



Save Server Object

Location

● Server

○ Client

Object Name: CalcOne

Objects:

Server: Aspen

Application: Sample

Database: Basic

List Objects of Type: Calc Scripts

Figure 30-23, Save Server Object Dialog Box

4. Choose the application or database with which you want to associate the calc script. For more information, see“Saving a Calc Script as an Object on the Essbase Server” on page 30-21.

5. In the **Object Name** text box, type the name that you want to give the calc script. You can type up to 8 alphanumeric characters.

6. Click OK.

Essbase saves the calc script on the Essbase server. For more information, see “Saving a Calc Script as an Object on the Essbase Server” on page 30-21.

Running a Calc Script

You can run a calc script from:

- Application Manager
- Your Spreadsheet Add-in (See your *Spreadsheet Add-in User's Guide*.)
- ESSCMD

If you run a calc script from the Application Manager, you can run it on your Essbase Client machine or on the Essbase Application Server.

When you run a calc script from the Application Manager or from your Spreadsheet Add-in, you can view the calculation messages in the Event Log file. When you run a calc script using ESSCMD, Essbase displays the messages in the ESSCMD window. To display the Event Log file, choose Application | View Event Log from the Application Manager menu.

Essbase displays:

- The calculation order of the dimensions for each pass through the database
- The total calculation time

You can use these messages to tune your database during calculation. You can display more detailed information using the SET MSG SUMMARY, SET MSG DETAIL, and SET NOTICE commands in a calc script. For more information, see "Specifying Global Settings for Your Database Calculation" on page 30-11.

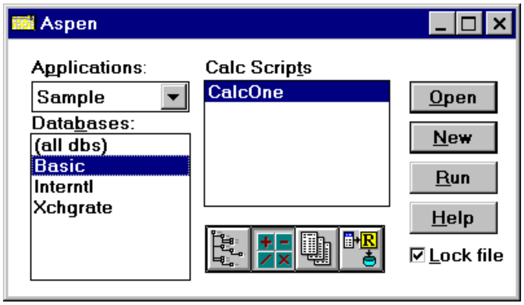
You can run a calc script from the Application Manager desktop or from the Application Manager menu.

Note: Before you can run a calc script in Application Manager, you must save it as a calc script object on the Essbase server or your client machine. See "Saving a Calc Script" on page 30-20. To run a calc script saved as an object on your client machine, you must run the calc script from the desktop.

To run a calc script from the desktop:

1. If the calc script is saved on the Essbase server, open the Application Desktop Server window. If the calc script is saved on your client machine, open the Application Desktop Client window.
2. In the Application Desktop Server or Client window, select the application and database that contains the calc script you want to run.

3. Click the ⊞ button to display the calc scripts associated with the application and database you selected. The following example shows the Application Desktop Server window for a server called Aspen:



Aspen

Applications: Sample

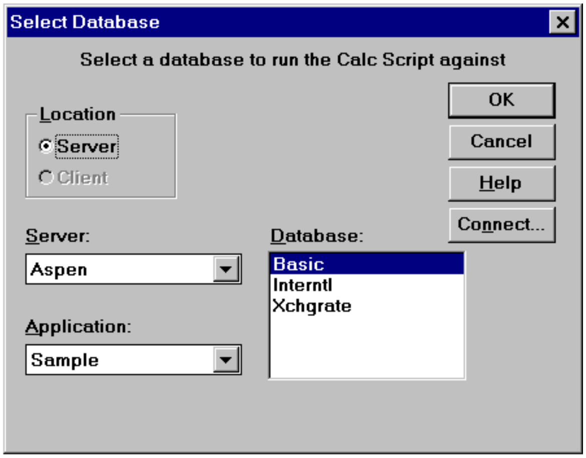
Databases:
(all dbs)
Basic
Interntl
Xchgrate

Calc Scripts:
CalcOne

☑ Lock file

Figure 30-24, Application Desktop Server Window

4. In the **Calc Scripts** list, select the calc script you want to run, and click the **Run** button. The **Select Database** dialog box appears.



Select Database

Select a database to run the Calc Script against

Location
● Server
○ Client

Server: Aspen

Application: Sample

Database:
Basic
Interntl
Xchgrate

Figure 30-25, Select Database Dialog Box

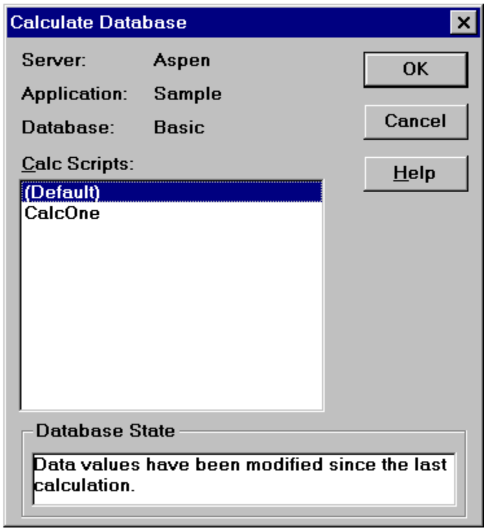
5. Select the Essbase server, application and database against which you want to run the calc script. If you are not currently connected to the server, click **Connect**.

6. Click OK.

Essbase runs the calc script against the database you selected.

To run a calc script from the menu:

1. In the Application Desktop Server window, select the application and database that contains the calc script you want to run.
2. Choose Database | Calculate from the Application Manager menu. The **Calculate Database** dialog box appears.



Calculate Database

Server: Aspen

Application: Sample

Database: Basic

Calc Scripts:

(Default)

CalcOne

Database State

Data values have been modified since the last calculation.

Figure 30-26, Calculate Database Dialog Box

3. In the **Calc Scripts** list, select the calc script you want to run and click OK. Only scripts to which you have security access appear in the list.

Essbase runs the calc script against the database you selected in the Application Desktop Server window.

You can use the RUNCALC command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

For information on running a calc script from your Spreadsheet Add-in, see your *Spreadsheet Add-in User's Guide*.

Printing a Calc Script

To print a calc script:

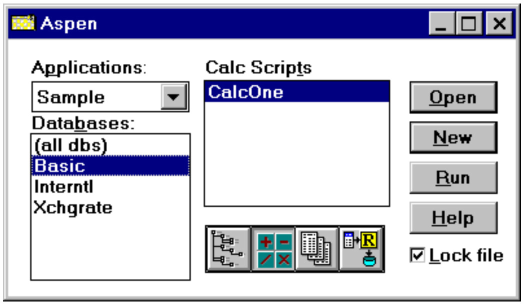
1. Open the calc script in the Calc Script Editor. For more information see "Opening the Calc Script Editor" on page 30-14.
2. Choose File | Print, or click the 🖨 button. The **Print Calc Script** dialog box appears.
3. If you want to print page numbers on your calc script, check **Page Numbers**.
4. Click **Print**.

Deleting a Calc Script

How you delete a calc script depends on where it is saved.

Deleting a Calc Script Saved as an Object On the Essbase Server or On Your Client Machine

1. In the Application Desktop Server or Client window, select the application and database with which the calc script is associated.
2. Click the ⊞ button. Essbase displays a list of all the calc scripts associated with the application and database you chose.



Aspen

Applications: Sample

Databases: (all dbs), Basic, Interntl, Xchgrate

Calc Scripts: CalcOne

☑ Lock file

Figure 30-27, Application Desktop Server Window

3. In the **Calc Scripts** list, select the calc script that you want to delete.
4. Choose File | Delete from the Application Manager menu. Essbase displays a **Confirm Delete** message box.
5. Click Yes to delete the calc script.

Deleting a Calc Script Saved in Your Client Machine's File System

You cannot delete the file using Application Manager. Delete the calc script file using your client machine's file system.

Undoing the Last Action

In the Calc Script Editor, select Edit | Undo, or click the button.

Using Formulas in Your Calc Script

You can place member formulas in a calc script. When you place formulas in a calc script, they override any conflicting formulas applied to members in the database outline.

In a calc script, you can:

- Calculate a member formula on the database outline
- Define a formula

To calculate a formula 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 the calc script, use the Calc Script Formula 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 calc script as follows:

```
Member = mathematical_operation;
```

where *Member* is a member name from your database outline, and *mathematical_operation* is any valid mathematical operation.

For example:

```
Margin = Sales - COGS;
```

cycles through the database subtracting the values in COGS from the values in Sales and placing the results in Margin.

```
Markup = (Retail - Cost) % Retail;
```

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 3, Multidimensional Concepts.

Conditional Equations

When you use an IF statement as part of a member formula in a calc script, you need to:

- Associate it with a single member
- Enclose it in parentheses

For example:

```
Profit (IF (Sales > 100)
      Profit = (Sales - COGS) * 2;
ELSE
      Profit = (Sales - COGS) * 1.5;
ENDIF)
```

Essbase cycles through the database, performing 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 it by 2, and places the result in Profit.
3. If Sales is less than, or equal to 100, Essbase subtracts the value in the COGS member from the value in the Sales member, multiplies it by 1.5, and places the result in the Profit member.

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 calc script, the same rules apply as for the IF statement. You need to:

- Associate the formula with a single member
- Enclose the formula in parentheses

Consider the interdependent formula we discussed earlier. If you place the formula in a calc script, you construct it as follows:

```
"Opening Inventory" (IF(NOT @ISMBR (Jan))"Opening Inventory" =
@PRIOR("Ending Inventory"));
ENDIF
"Ending Inventory" = "Opening Inventory" - Sales + Additions;)
```

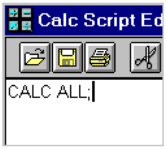
The whole of the formula is enclosed in parentheses and associated with the Opening Inventory member, `"Opening Inventory" (IF(...)...)`.

Inserting Text and Operators in Your Calc Script

You can type text and operators directly into the text area, or you can use the toolbar buttons to add the text and operators.

Typing Text

To type text, in the Calc Script Editor, click in the text area below the toolbar, and begin to type. Text appears at the cursor position as you type.



```
CALC ALL;
```

Figure 30-28, Calc Script Editor Showing Calc Script

Cutting, Copying, and Pasting Text

To cut text, in the Calc Script Editor, select the text that you want to cut, and choose Edit | Cut, or click the ✂ button. You can also press `Ctrl+X`.

To copy text, in the Calc Script Editor, select the text that you want to copy, and choose Edit | Copy, or click the ⎘ button. You can also press `Ctrl+C`.

To paste text, on the Calc Script Editor, select the text that you want to paste, and choose Edit | Paste, or click the 📋 button. You can also press `Ctrl+V`.

Finding and Replacing Text

To find and replace text:

1. In the Calc Script Editor, choose Edit | Find. Essbase displays the **Find** dialog box.

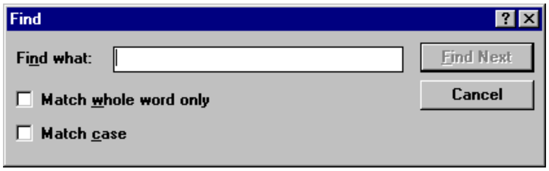


Figure 30-29, Calc Script Editor Find Text Dialog Box

2. In the **Find what** text box, type the characters that you want to search for, and click the **Find Next** button.

   Doing a Case-Sensitive Search:

   a. Check **Match case** in the **Find** dialog box. For example, to search for Margin but not margin, type `Margin` in the **Find what** text box and check **Match case**.

   b. Click **Find Next**.

   Searching For Whole Words Only:

   a. Check **Match whole word only** in the **Find** dialog box. For example, to search for Margin but not Margin%, type `margin` in the **Find what** text box and check **Match whole word only**.

   b. Click **Find Next**.

Inserting an Equal (=) Sign

In the Calc Script Editor, place the cursor where you want to insert the equal (=) sign, and type = or click the = button.

Inserting a Mathematical Operator (+, -, X, /, %)

In the Calc Script Editor, place the cursor where you want to insert the mathematical operator, and type the operator or click one of the following toolbar buttons:

+ − × / %

For example, to insert an addition (+) operator:

1. Place the cursor where you want to insert the addition (+) operator.
2. Type +, or click the + button.

Inserting the Cross-Dimensional Operator (->)

In the Calc Script Editor, place the cursor where you want to insert the cross-dimensional operator, and type a - (hyphen) followed by a > (greater than symbol), or click the → button.

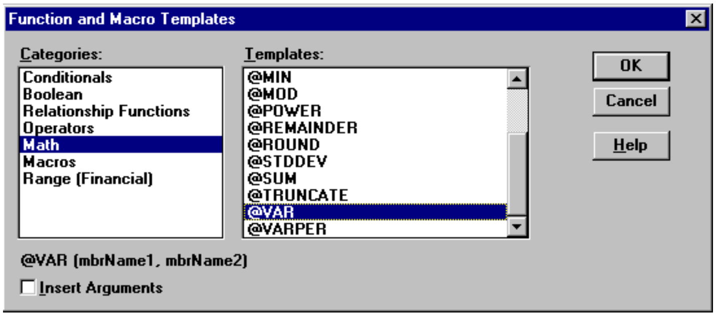
For more information on the cross-dimensional operator, see Chapter 25, Developing Formulas.

Inserting the Semicolon Formula End-Of-Line Character (;)

In the Calc Script Editor, place the cursor at the end of the formula, and type a ; (semicolon) or click the ; button.

Inserting a Function, Operator, or Macro in Your Calc Script

1. In the Calc Script Editor, place the cursor where you want to insert the function or macro, and choose Formula | Paste Function or click the f(x) button.
2. Select the function category in the **Categories** list. For example, to insert the @VAR function, select **Math**.
3. Select the required function, operator, or macro in the **Templates** list. For example, scroll down the list and select @VAR. Essbase displays the function, operator, or macro and the default arguments below the **Categories** list.

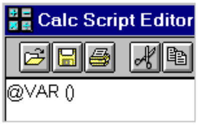


@VAR (mbrName1, mbrName2)

☐ Insert Arguments

Figure 30-30, Function and Macro Templates Dialog Box

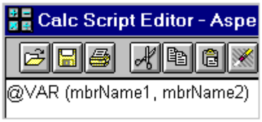
4. If required, check **Insert Arguments**, to insert default, temporary arguments in the function.
5. Click OK. Essbase inserts @VAR at the cursor position.



```
@VAR ()
```

Figure 30-31, Calc Script Editor With @VAR Function Inserted

If you checked **Insert Arguments**, Essbase inserts @VAR and default, temporary arguments. You can then type over these with the correct arguments.



@VAR (mbrName1, mbrName2)

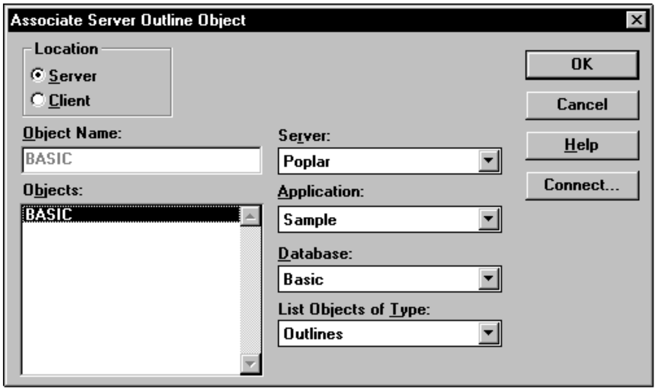
Figure 30-32, Calc Script Editor With @VAR Function and Arguments Inserted

Associating Your Calc Script with a Database

If you want to insert member names in your calc script by selecting them from in the Calc Script Editor, you need to associate the calc script with the database outline containing the members.

To associate your calc script with a database outline:

1. Click the button or choose Options | Associate Outline from the Calc Script Editor menu. Essbase displays the **Associate Client Outline** or **Associate Server Outline** dialog box.



Location
● Server
○ Client

Object Name: BASIC

Objects: BASIC

Server: Poplar

Application: Sample

Database: Basic

List Objects of Type: Outlines

Figure 30-33, Associate Server Outline Object Dialog Box

2. Under **Location**, select **Client** to associate an outline saved on your client machine. Select **Server** to associate an outline saved on the Essbase server machine.
3. In the **Server**, **Application**, and **Database** lists, choose the server, application, and database containing the outline that you want to associate with your calc script.

4. In the **Objects** list, choose the database outline.
5. Click OK. Essbase displays the associated outline's dimension names in the **Dimensions** list. You can now insert members from this list. See "Inserting Members in Your Calc Script" on page 30-39.

Essbase associates your calc script with the database outline only while you are editing the calc script. When you close the Calc Script Editor, Essbase cancels the association. If you want to insert members from this database outline in the future, you need to re-associate the outline with your calc script.

Inserting Members in Your Calc Script

To insert members in your calc script:

1. Associate the database outline containing the dimensions and members that you want to insert. See "Associating Your Calc Script with a Database" on page 30-38.
2. In the Calc Script Editor, 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 your formula.

   The dimension member name appears in the **Members** list. If a ☑ button appears to the left of the dimension member name, then the dimension has children. The following shows the Scenario dimension in the Sample Basic database.

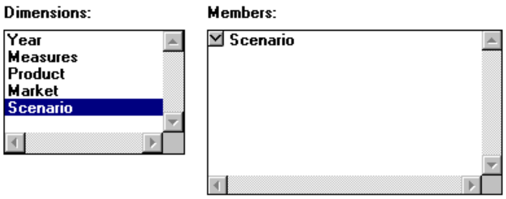
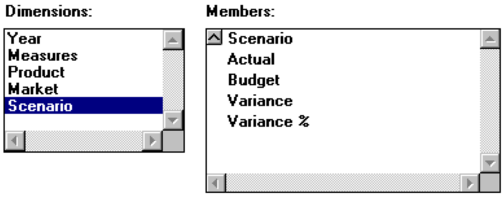


Figure 30-34, Inserting Dimensions and Members In a Calc Script

   If you want to insert the dimension member name in your formula, click the dimension name in the **Members** list. Essbase inserts the dimension member name at the cursor position.

Expanding 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.



Dimensions:

Year
Measures
Product
Market
Scenario

Members:

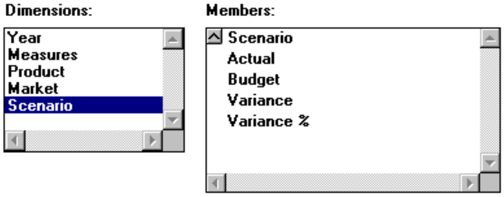
⌃ Scenario
Actual
Budget
Variance
Variance %

Figure 30-35, Expanding a Member Branch

The ⌄ button changes to a ⌃ button. You can click the ⌄ button to collapse the member branch.

Collapsing a Member Branch

In the **Members** list, double-click the ⌄ button next to collapse the member branch.



Dimensions:

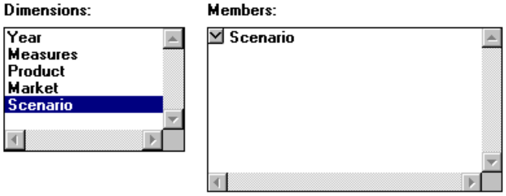
Year
Measures
Product
Market
Scenario

Members:

⌃ Scenario
Actual
Budget
Variance
Variance %

Figure 30-36, Collapsing a Member Branch

Essbase does not display the member’s children:



Dimensions:

Year
Measures
Product
Market
Scenario

Members:

⌄ Scenario

Figure 30-37, Collapsing a Member Branch

The ⌄ button has changed to a ⌃ button. Click the ⌄ button to expand the member branch.

Searching for Members

1. In the **Dimensions** list, select the dimension in which you want to search for the member. For example, select the Measures dimension from the Sample Basic database.
2. Click the **Find Member** button. Essbase displays the **Find** dialog box.

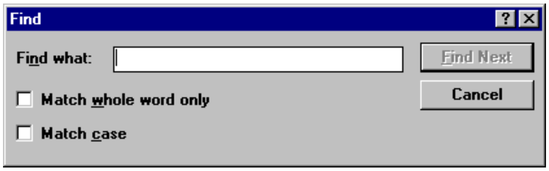
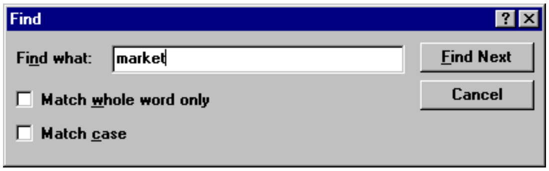


Figure 30-38, Searching For Members

3. In the **Find what** text box, type the characters that you want to search for. If you want to make the search case-sensitive, check **Match case**. For example to search for the Marketing member in the Measures dimension, enter `market`.



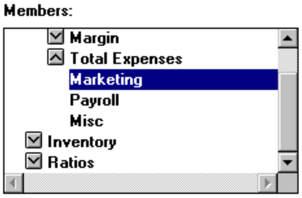
Find what: market

□ Match whole word only

□ Match case

Figure 30-39, Searching For Members

4. Click the **Find Next** button. Essbase finds and selects the Marketing member.


Members:

- Margin
- Total Expenses
  - Marketing
  - Payroll
  - Misc
- Inventory
- Ratios

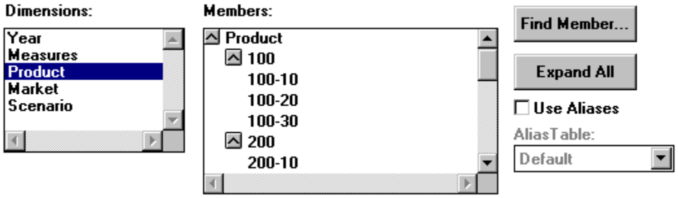
Figure 30-40, Searching For Members

To search for whole words only:

1. Check **Match whole word only** in the **Find** dialog box. For example to search for Margin, but not Margin% in the Sample Basic database, type `margin` in the **Find what** text box, and check **Match whole word only**.
2. Click **Find Next**.

Expanding a Dimension to Display All the Members

1. In the Calc Script Editor **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.

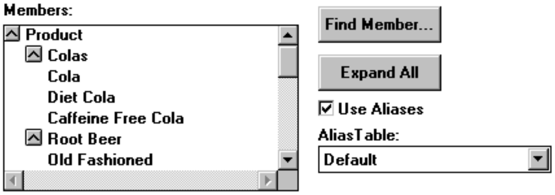


Dimensions:
Year
Measures
Product
Market
Scenario

Members:
Product
100
100-10
100-20
100-30
200
200-10

Figure 30-41, Expanding a Dimension

2. Click the **Expand All** button. In the **Members** list, Essbase displays all the members in the dimension.

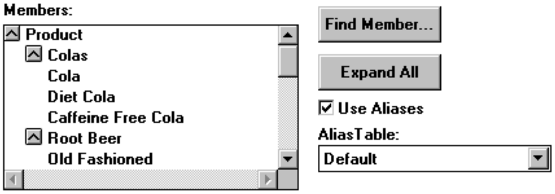


Members:
Product
Colas
Cola
Diet Cola
Caffeine Free Cola
Root Beer
Old Fashioned

Figure 30-42, Expanding a Dimension

Displaying and Inserting Alias Names

1. In the Calc Script Editor, check **Use Aliases**. Essbase displays the alias names for the members. The following example shows the Product dimension from the Sample Basic database.



Members:

Product
- Colas
  - Cola
  - Diet Cola
  - Caffeine Free Cola
- Root Beer
  - Old Fashioned

☑ Use Aliases

AliasTable: Default

Figure 30-43, Displaying and Inserting Alias Names

2. To choose a different alias table, choose the table from the **Alias Table** list box.

   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

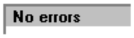
Essbase includes a syntax checker that tells you about any syntax errors in your calc script. For example, Essbase tells you if you have mistyped a function name.

The syntax checker cannot tell you about semantic errors in your calc script. Semantic errors occur when your calc script does not work as you expect. To find semantic errors, always run your calculation, and check the results to ensure they are as you expect.

To check the syntax of your calc scripts:

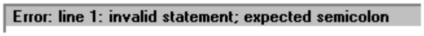
- In the Calc Script Editor, choose Syntax | Check or click the ✓ button. Essbase displays the syntax checker results at the bottom of the Calc Script Editor window.

  If Essbase finds no syntax errors, it displays the following message:

  No errors

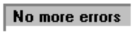
  If Essbase finds one or more syntax errors, it displays the line number, which includes the error and a brief description. For example, if you do not include a semicolon end-of-line character at the end of a calc script command, Essbase displays a message similar to the following:

  Error: line 1: invalid statement; expected semicolon

Stepping Through Syntax Errors

- Choose 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.

Controlling Intelligent Calculation Using a Calc Script

If you have a formula on a sparse dimension member and the formula contains:

- Index functions (for example, @PRIOR or @NEXT)
- Financial functions (for example, @NPV or @INTEREST)

Essbase always recalculates the data block containing the formula, even if it is marked as *clean* for the purposes of Intelligent Calculation. For more information, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

Grouping Formulas and Calculations

You may achieve significant calculation performance improvements by carefully grouping formulas and dimensions in your calc script. For more information and examples, see "Calculating a Series of Member Formulas" on page 30-45, and "Calculating a Series of Dimensions" on page 30-46.

When you run a calc script, Essbase automatically displays the calculation order of the dimensions for each pass through the database. This tells how many times Essbase has cycled through your database during the calculation.

Essbase displays these information messages in the ESSCMD window and in the Event Log file. To display the Event Log file, choose 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.

```
Profit = (Sales - COGS) * 1.5;
Market = East + West;
```

In the same way:

```
Qtr1;
Qtr2;
Qtr3;
```

or

```
(Qtr1;
Qtr2;
Qtr3;)
```

cause Essbase to cycle through the database only once, calculating the formulas on the members Qtr1, Qtr2, and Qtr3.

However,

```
(Qtr1;
Qtr2;)
Qtr3;
```

causes Essbase to cycle through the database twice. It cycles through once calculating the formulas on the members Qtr1 and Qtr2. It cycles through a second time calculating the formula on Qtr3.

Calculating a Series of Dimensions

When you calculate a series of dimensions, you can optimize performance by grouping the dimensions together wherever possible.

For example:

```
CALC DIM(Year, Measures);
```

causes Essbase to cycle through the database only once.

However,

```
CALC DIM(Year);
CALC DIM(Measures);
```

causes Essbase to cycle through the database twice. It cycles through once for each CALC DIM command.

Using Substitution Variables

You can use substitution variables in your calc scripts. This is useful, for example, when you frequently reference information or lists of members that change.

When you include a substitution variable in a calc script, Essbase replaces it with the value you have specified for the substitution variable.

You create and specify values for substitution values in the Application Manager. For more information, see Chapter 7, Creating Applications and Databases.

You can create variables at the server, application, and database level. When you use a substitution variable in a calc script, it must be available to the calc script. For example, if you create a substitution variable at the database level, it is only available to calc scripts within the database. However, if you create a variable at the server level, it is available to any calc script on the server.

The & command prefaces a substitution variable in your calc script. Essbase treats any string that begins with a leading & as a substitution variable; substituting these variables with their values before it parses the calc script.

For example:

```
&CurQtr;
```

becomes:

```
Qtr1;
```

if you have given the substitution variable `&CurQtr` the value `Qtr1`.

Consider the following example, in which you want to calculate Sample Basic data for the current quarter. You can use the following calc 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 calc script.

Clearing Data

You can use the CLEARDATA and CLEARBLOCK commands to remove data values and data blocks from your 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. This includes all the dense dimension members. Essbase removes the entire block, regardless of any FIX command on members within the block.

The following examples are based on the Sample Basic database. If the Scenario dimension is dense:

```
FIX(Actual)
CLEARBLOCK NONINPUT;
ENDFIX
```

removes all the data blocks that do not contain input data values. Essbase ignores the FIX command.

If the Scenario dimension is sparse:

```
FIX(Actual)
CLEARBLOCK NONINPUT;
ENDFIX
```

removes only the blocks whose Scenario dimension member is Actual. The other blocks remain.

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:

```
CLEARDATA Actual->Colas;
```

clears all the Actual data values for Colas.

You can use the FIX command with the CLEARDATA command to clear a subset of your database.

If you want to clear your entire database, you can also choose the Clear Data command from the Database Menu in the Application Manager.

For more information on the CLEARBLOCK and CLEARDATA commands, see the online *Technical Reference* in your `DOCS` directory.

Copying Data

You can use the DATACOPY command to copy a range of data cells to another range in your database. The two ranges must be the same size.

For example in the Sample Basic database:

```
DATACOPY Actual TO Budget;
```

copies Actual values to Budget values in your database.

You can use the FIX command to copy a subset of values.

Note: For currency conversions, you can only use the DATACOPY command on members of the dimension tagged as CURPARTITION. You cannot use the FIX command to copy a subset of values. For more information, see “Writing Calc Scripts for Application Partitions” on page 30-51.

Calculating a Subset of Your Database

You can calculate a subset of your database, which means that you can calculate separate sections of your database using different formulas.

To calculate a subset of your database, you can use:

- Macro functions to calculate lists of members.
- The FIX ... ENDFIX command to calculate a range of values.

For more information, see “Calculating Lists of Members” on page 30-49 and “Using the FIX Command” on page 30-49.

Note: When you have Intelligent Calculation turned on, the newly calculated data blocks are not marked as *clean* after a partial calculation of your database. When you calculate a subset of your database, you can ensure that the newly calculated blocks are marked as clean using the SET CLEARUPDATESTATUS AFTER command.

This ensures that Essbase recalculates your database as efficiently as possible using Intelligent Calculation. For more information on Intelligent Calculation, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation. For more information on the SET CLEARUPDATESTATUS command, see the online *Technical Reference* in your `DOCS` directory.

Calculating Lists of Members

You can use macro functions to generate lists of members based on a member you specify. For example, you can use the @IDESCENDANTS function to generate a list of all the descendants of a member.

In the Sample Basic database `@IDESCENDANTS("Total Expenses");` generates the following list of members: Total Expenses, Marketing, Payroll, Misc.

When you use a macro function in a formula, Essbase generates the list of members before calculating the formula.

For detailed information on these and other macro functions, see the online *Technical Reference* in your `DOCS` directory.

Using the FIX Command

The FIX ... ENDFIX command is particularly useful to calculate a carefully defined subset of the values in your database. For example, the following calc script calculates only the Budget values for descendants of East in the Sample Basic database:

```
FIX(Budget,@Descendants(East))
CALC DIM(Year, Measures, Product);
ENDFIX
```

In the Sample Basic database, this calculates the Budget values for New York, Massachusetts, Florida, Connecticut, and New Hampshire.

This second example fixes on member combinations for the children of East that have a User-Defined Attribute (UDA) of New Mkt. For information on defining User-Defined Attributes, see Chapter 8, Creating and Changing Database Outlines.

```
FIX(@CHILD(East) AND @UDA(Market,"New Mkt"))
CALC DIM(Year, Measures, Product, Market);
ENDFIX
```

The following example uses a wildcard match to fix on member names that end in the characters -10. In Sample Basic, this fixes on the members 100-10, 200-10, 300-10, and 400-10.

```
FIX(@MATCH(Product, "???-10"))
CALC DIM(Year, Measures, Product);
ENDFIX
```

When you use the FIX command on a dense dimension, Essbase retrieves the entire block containing the required value or values for the member or members you specify. This means that I/O is not affected, but the calculation performance time is improved.

When you use the FIX command on a sparse dimension, Essbase retrieves the block for the sparse dimension member or members specified. This means that I/O may be greatly reduced.

Essbase cycles through your database once for each FIX command that you use on dense dimension members. When possible, combine FIX blocks to improve your calculation performance. For example:

```
FIX(Actual,Budget)
CALC DIM(Year, Measures);
ENDFIX
```

causes Essbase to cycle through the database only once calculating both the Actual and the Budget values.

However,

```
FIX(Actual)
CALC DIM(Year, Measures);
ENDFIX
FIX(Budget)
CALC DIM(Year, Measures);
ENDFIX
```

causes Essbase to cycle through the database twice. It cycles through once calculating the Actual data values. It cycles through a second time calculating the data values for Budget.

For detailed information on using the FIX command, see the online *Technical Reference* in your `DOCS` directory.

Writing Calc Scripts for Application Partitions

A Hyperion Essbase OLAP Server partitioned application can span multiple servers, processors, or computers. For more information on partitioning, see Chapter 6, Designing Partitioned Applications and Chapter 15, Building and Maintaining Partitions.

You can achieve significant calculation performance improvements by partitioning your applications and running separate calculations on each partition.

However, when you use partitioning, you need to:

- Consider carefully the performance impact on your overall database calculation. You might choose to:
    - Redesign your overall calculation to avoid referencing remote values, which are in a transparent partition in a remote database.
    - Dynamically calculate the value in the remote database. See Chapter 28, Dynamically Calculating Data Values.
    - Replicate the value in the database containing the formula. See Chapter 6, 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 the referenced value is up-to-date when Essbase retrieves it. You can do this by choosing one of the above, or by calculating the referenced database before calculating the formula.

Calculating Multiple Databases

You need to calculate your databases in a specific order to ensure that Essbase calculates the required results. For example, consider the following application partitions in which you view information from the West, Central, and East databases transparently from a central Corporate database.

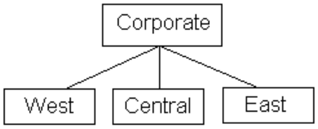


Figure 30-44, Calculating Application Partitions

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. However, you do need to calculate Corporate to calculate the budgeted values.

When Essbase calculates Corporate, it needs to take the latest values from West, Central, and East. To calculate the required results, you need to calculate West, Central, and East before you calculate Corporate.

Chapter 31 Examples of Calc Scripts

The examples in this chapter illustrate different types of calc scripts, which you may want to adapt for your own use.

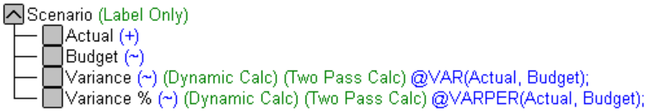
This chapter includes the following examples:

- "Calculating Variance" on page 31-1
- "Calculating a Subset of Your Database" on page 31-3
- "Loading New Budget Values" on page 31-4
- "Calculating Product and Market Share Values" on page 31-5
- "Allocating Costs Across Products" on page 31-6
- "Goal Seeking Using the LOOP Command" on page 31-8

For more examples that use the Intelligent Calculation commands SET UPDATECALC and SET CLEARUPDATESTATUS in calc scripts, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

Calculating Variance

The Sample Basic database includes a calculation of the percentage of variance between Budget and Actual values.



```
Scenario (Label Only)
 ├─ Actual (+)
 ├─ Budget (~)
 ├─ Variance (~) (Dynamic Calc) (Two Pass Calc) @VAR(Actual, Budget);
 └─ Variance % (~) (Dynamic Calc) (Two Pass Calc) @VARPER(Actual, Budget);
```

Figure 31-1, Calculating Variance and Variance %

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 Essbase dynamically calculates Variance% values when you retrieve them. This 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 calc 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), the following calc script performs a default calculation and then recalculates the formula on Variance %:

```
CALC ALL;

SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
"Variance %";
```

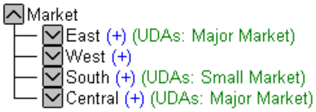
Essbase performs the following calculations:

1. 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 calc script.
2. The SET UPDATECALC OFF command turns off Intelligent Calculation.
3. The CLEARUPDATESTATUS AFTER command tells 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. Essbase cycles through the database calculating the formula for Variance %.

For more information on using a calc script for Two-Pass calculations, see Chapter 32, Optimizing Your Calculations. For more information on developing formulas, see Chapter 25, Developing Formulas.

Calculating a Subset of Your 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.



```
Market
    East (+) (UDAs: Major Market)
    West (+)
    South (+) (UDAs: Small Market)
    Central (+) (UDAs: Major Market)
```

Figure 31-2, Market Dimension from the Sample Basic Database

The following calc script is used by the Marketing manager of the region East to calculate the data values for East. It calculates the Year, Measures, and Products dimensions for each child of East.

```
/* Calculate the Budget data values the descendants of East */
FIX(Budget, @DESCENDANTS(East))
CALC DIM(Year, Measures, Product);
ENDFIX
/* Consolidate East */
FIX(Budget)
@DESCENDANTS(East);
ENDFIX
```

Essbase performs the following calculations:

1. 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. Essbase fixes on the Budget values for all members on the other dimensions.
4. Essbase aggregates the descendants of East and places the result in East.

The following three calc scripts are used by the Marketing managers of the other regions:

```
/* Calculate the Budget data values 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 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 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

The following example loads budget values into the Sample Basic database and recalculates the database:

```
/* Recalculate all the Budget values */
FIX(Budget)
CALC DIM(Year, Product, Market, Measures);
ENDFIX

/* Recalculate the Variance and Variance% formulas, which
      require two passes */
Variance;
"Variance %";
```

Essbase performs the following calculations:

1. Essbase fixes on the Budget values.
2. It calculates all the Budget values. The CALC DIM command is used to calculate all the dimensions except for the Scenario dimension, which contains Budget.
3. Essbase calculates the formula applied to Variance in the database outline.
4. 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);
```

Essbase performs the following calculations:

1. 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. Essbase cycles through the whole 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). For more information on using the cross-dimensional operator (->), see Chapter 25, Developing Formulas.
3. 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. 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. 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
```

Essbase performs the following calculations:

1. Essbase creates a one-dimensional array called ALLOCQ. The size of ALLOCQ is based on the number of members in the Year dimension. 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. The default is set in the `ESSBASE.CFG` file.

   - Essbase fixes on the Actual values.
   - 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 Calc Scripts.

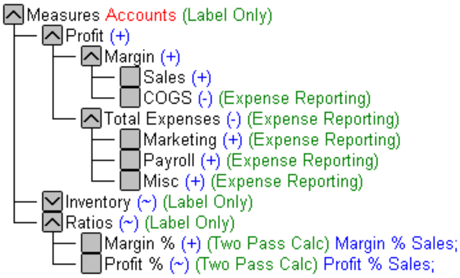
3. Essbase calculates and consolidates the Measures dimension.

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. 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.



```
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;
```

Figure 31-3, Measures Dimension from the Sample Basic Database

Assume that the data values before running the goal-seeking calc 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

The calc script is as follows:

```
/* 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
```

Essbase performs the following calculations:

1. It declares the required temporary variables using the VAR command. Where appropriate, the initial values are set.
2. Essbase declares a one-dimensional array called Rollback. The size of Rollback is based on the number of members in the Measures dimension. Essbase uses Rollback to store the Budget values.
3. 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. Essbase cycles through the member combinations, performing the following calculations:

6. Essbase places the Budget, Sales value in the Rollback temporary array variable.

7. 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.

8. It retains the current Sales value. It places the Sales value for the current member combination in the PriorVar temporary variable.

9. It retains the current Profit value. It places the Profit value for the current member combination in the PriorTar temporary variable.

10. 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.

11. Essbase calculates and consolidates the Measures dimension.

12. It subtracts the PriorTar value from the PriorVar value and places the result in the DependencyCheck temporary variable.

13. The IF command checks that DependencyCheck is not 0 (zero).

14. If DependencyCheck is not 0, then 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, Essbase calculates the Slope, NxtVar, and PctNewVarChange temporary variables.

    If it is not greater than AcceptableError, Essbase breaks the LOOP command by setting the value of Quit to 1. The calculation continues after the ENDLOOP command.

15. If DependencyCheck is 0, Essbase places the value in the Rollback array into Budget. Essbase breaks the LOOP command by setting the value of Quit to 1. The calculation continues after the ENDLOOP command.

16. Essbase calculates and consolidates the Measures dimension.

The results are shown in the following 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

Chapter 32 Optimizing Your Calculations

This chapter provides some information on how to optimize the performance of your Hyperion Essbase calculations. In addition, you can significantly optimize your overall database calculations using:

- Intelligent Calculation, as discussed in Chapter 33, Optimizing Your Calculation Using Intelligent Calculation
- Dynamic Calculations, as discussed in Chapter 28, Dynamically Calculating Data Values
- Partitioning; you can achieve significant calculation performance improvements by partitioning your applications and running separate calculations on each partition. See Chapter 6, Designing Partitioned Applications and Chapter 30, Developing Calc Scripts.

This chapter includes the following sections:

Designing for Calculation Performance

You can configure your database to optimize calculation performance.

The optimal configuration is highly dependent on the nature and size of your database. The following sections provide guidelines only.

Block Size and Block Density

Generally speaking, you do not want the data block size to be too small. A data block size of 8 K to 64 K provides optimal performance in most cases.

If the data blocks are very small, then the index is likely to be very large. This causes a performance overhead as Essbase writes and retrieves the index from disk. However, if the data blocks are too large, then Intelligent Calculation does not work effectively. For more information on Intelligent Calculation, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

To optimize calculation performance and data storage, you need to balance the data block density and size. You can do this by rearranging your database's dense and sparse dimension configuration:

- Aim to achieve a data block size of between 8K and 64K with as high a block density as possible.
- Run test calculations on the most promising configurations of a database that contains representative data. Check the results to determine the optimal configuration for calculation performance.

You can display information on your database, including the potential and actual number of data blocks and the data block size. Choose Database | Information in Hyperion Essbase Application Manager to display the **Database Information** dialog box. Choose the **Statistics** tab to display data block statistics.

Order of Sparse Dimensions

You may achieve a performance improvement by changing the order of the sparse dimensions in your database outline. Order the sparse dimensions by the number of members, starting with the dimension that contains the fewest members.

To achieve the maximum performance benefit from the calculator cache, make the largest sparse dimension the last dimension in your database outline. This provides approximately a 10% performance improvement if you have a database outline with a very large dimension (for example, containing more than 1000 members).

For more information on the calculator cache, see “Setting the Memory Cache Sizes” on page 32-11.

Incremental Data Loading Considerations

Many people load data incrementally. For example, they load data on a month-by-month basis. Each month they load new data for that month.

To optimize calculation performance when you load data incrementally, consider making the Time dimension a sparse dimension. If the Time dimension is sparse, the data for each time period is contained in a separate data block. When you load data, you are loading it into the required data blocks only. This means that if you have Intelligent Calculation enabled, only the data blocks marked as *dirty* are recalculated.

For example, if you load data for March, then 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 their dependent parents.

Note: Making the Time dimension sparse when it is naturally dense may significantly increase the size of the index.

Intelligent Calculation can still provide some benefits when the Time dimension is dense 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 only recalculates the blocks affected by the partial load, even though Time is dense (with Intelligent Calculation enabled).

For more information on Intelligent Calculation, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

Performance for Database Outlines with Two or More Flat Dimensions

If your database outline has two or more flat dimensions, then your calculation performance might be affected. A *flat dimension* is a dimension in which there are very few parents, each with many (thousands) of children.

If this is the case, you can achieve a performance improvement by doing one of the following:

- Adding intermediate levels to your database outline.
- Using the SET CALCHASHTBL command in a calc script to optimize the calculation of large, flat database outlines.

  You can specify the default setting for this option in the configuration (`ESSBASE.CFG`) file, using the CALCOPTCALCHASHTBL setting. You can set the maximum amount of memory that you want the Calculator hash table to use by using the CALCHASHTBLMEMORY setting in the configuration file. For more information, see the online *Technical Reference* in your `DOCS` directory.

Monitoring and Tracing Calculations

You can display information about how Essbase is calculating your database using the following commands and settings in a calc script:

- SET MSG SUMMARY
- SET MSG DETAIL
- SET NOTICE

SET MSG SUMMARY and DETAIL

You can use these commands in a calc script to display calculation settings (for example, whether Completion Notice Messages are enabled) and provide statistics on the number of:

- Data blocks created, read, and written
- Data cells calculated

In addition, the SET MSG DETAIL command provides a detailed information message every time Essbase calculates a data block.

SET MSG DETAIL is useful for seeing the calculation order of data blocks, and for testing intelligent recalculations.

The SET MSG DETAIL command causes a high processing overhead.

SET MSG SUMMARY causes a processing overhead of approximately 1% to 5%, depending on the database size.

For more information on these commands, see the online *Technical Reference* in your `DOCS` directory.

SET NOTICE

You can use this setting to display calculation completion notices, informing you what percentage of your database has been calculated.

You can use the SET MSG SUMMARY command together with the SET NOTICE command to show the calculation progress between completion notices.

Completion notices do not significantly reduce the calculation performance, except when used with a very small database.

For more information on this command, see the online *Technical Reference* in your `DOCS` directory.

Using Formulas

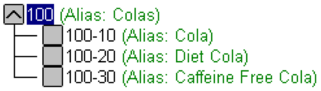
You may achieve significant calculation performance improvements by careful use of formulas in your database calculation. For example, you may achieve improved calculation performance by placing formulas on members in your database outline instead of placing the formulas in a calc script. For more information, see Chapter 25, Developing Formulas.

Consider the following questions to ensure that you use formulas in a way that optimizes calculation performance.

Could You Use a Consolidation on the Database Outline?

Using the database outline to roll up values is always more efficient than using a formula to calculate values.

For example, the following consolidation on the Sample Basic database outline:



```
100 (Alias: Colas)
    100-10 (Alias: Cola)
    100-20 (Alias: Diet Cola)
    100-30 (Alias: Caffeine Free Cola)
```

Figure 32-1, Consolidation on Sample Basic Outline

is more efficient than applying the following formula to the Colas member:
100-10 + 100-20 + 100-30

Are You Using a Simple Formula?

A *simple formula* is, for example, a ratio or a percentage. A simple formula does *not*:

- Reference values from a different dimension (sparse or dense); for example, Product->Jan.
- Use the mathematical range functions: @AVGRANGE, @MAXRANGE, @MINRANGE, @STDDEVRANGE, or @SUMRANGE.
- Use index or financial functions; for example, @ANCESTVAL, @NEXT, @PARENTVAL, @PRIOR, or @SHIFT. For a complete list of index and financial functions, see the online *Technical Reference* in your DOCS directory.

If you are using a simple formula, you can place it on either a sparse or a dense dimension, without significantly affecting calculation performance. However, consider that the bigger your block size, the more impact simple formulas have on your calculation performance. See “Block Size and Block Density” on page 32-2.

Are You Using a Complex Formula?

A complex formula:

- References a member or members on a different dimension (sparse or dense); for example, Product->Jan.
- Uses one or more of the mathematical range functions: @AVGRANGE, @MAXRANGE, @MINRANGE, @STDDEVRANGE, or @SUMRANGE.
- Uses index or financial functions; for example, @ANCESTVAL, @NEXT, @PARENTVAL, @PRIOR, or @SHIFT. For a complete list of index and financial functions, see the online *Technical Reference* in your `DOCS` directory.

If you are using a complex formula, applying your formula to a member on a *sparse* dimension can significantly lower calculation performance.

If you apply a complex formula to a member on a sparse dimension, Essbase checks each possible sparse member combination (possible data block) to see if it needs calculating. This evaluation causes a significant calculation overhead. If your complex formula uses index or financial formulas, Essbase evaluates each possible sparse member combination, which causes an even greater calculation overhead.

The lower the ratio of existing data blocks to possible data blocks, the higher the calculation performance overhead.

If you are using a complex formula, consider:

- If possible, applying the formula to a member on a *dense* dimension.
- Using the FIX command in a calc script to calculate only the minimum, required data blocks. See Chapter 30, Developing Calc Scripts.
- Increasing the density of your database (ratio of existing data blocks to possible data blocks). See “Block Size and Block Density” on page 32-2.

Optimizing Formulas on Sparse Dimensions in Large Database Outlines

You can use the SET FRMLBOTTOMUP calculation command to optimize the calculation of formulas on sparse dimensions in large database outlines by turning on the bottom-up sparse formula calculation method. This means that you can efficiently use CALC ALL and CALC DIM commands to calculate the database. For more information, see the SET FRMLBOTTOMUP command and the CALCOPTFRMLBOTTOMUP configuration setting in the online *Technical Reference* in your `DOCS` directory.

Assigning Constants to Members on a Sparse Dimension

When you assign a constant to a member on a sparse dimension, Essbase automatically creates a data block for every combination of sparse dimension members containing that member.

For example, a member or calc script formula contains the expression

```
California = 120;
```

where California is a member on a sparse dimension and 120 is a constant value.

Essbase automatically creates all the combinations of data blocks for California and assigns the value 120 to all the data cells. Potentially, this may create many thousands of data blocks.

When assigning constants on sparse dimension members, use the FIX command to ensure that Essbase creates only the required data blocks. For example:

```
FIX(Colas,Misc,Actual)
California = 120;
ENDFIX;
```

assigns the value 120 to California (on the Market dimension), Actual (on the Scenario dimension), Misc (miscellaneous expenses on the Measures dimension) values for Colas (on the Product dimension), and for all members on the Year dimension.

In the Sample Basic database, Colas is a member on the Product dimension, which is a sparse dimension. Actual is a member on the dense Scenario dimension and Misc is a member on the dense Measures dimension. Essbase creates new data blocks for all combinations of the sparse dimensions, California and Colas. It leaves other Measures and Scenario values set to `#MISSING` within these blocks.

For more information on the FIX command, see the online *Technical Reference* in your `DOCS` directory.

When you assign a constant to a member on a sparse dimension, you do not need to enable **Create Blocks on Equations**. However, you still need to enable **Create Blocks on Equations** in Application Manager when you assign anything other than a constant to a member on a sparse dimension, for which a data block does not already exist.

To enable **Create Blocks on Equations** in Application Manager:

1. Choose Database | Settings. Essbase displays the **Database Settings** dialog box.
2. Check **Create Blocks on Equations**.
3. Click OK.

For example, you would need to enable **Create Blocks on Equations** for the following formula:

```
West = California + 120;
```

Using the Cross-Dimensional Operator (->)

Use caution when using the cross-dimensional operator (->) in the following situations:

- On the left-hand side of an equation
- In equations on a dense dimension
- When doing a Two-Pass formula calculation on a dimension tagged as Accounts

On the Left Side of an Equation

It is possible to use the cross-dimensional operator on the left side of an equation in the database outline or in a calc script, when you associate it with a member in the database outline. However, it is considerably more efficient to use the FIX command in a calc script. Consider an example in which you want to increase the Jan->Sales values by 5% in the Sample Basic database. You place the following formula on the Sales member in your database outline. As Essbase cycles through the database, it calculates the formula for every combination of members. This means that Essbase recalculates the same formula for every member on the Time dimension, causing redundant calculations.

```
Sales(Sales->Jan = Sales->Jan * .05;)
```

You can use the FIX command in a calc script to achieve the same result more efficiently:

```
FIX(Jan)
      Sales = Sales * .05;
ENDFIX
```

When you use the FIX command, Essbase calculates the formula for member combinations for Jan only.

For more information on calc scripts and the FIX command, see Chapter 30, Developing Calc Scripts, and the online *Technical Reference* in your `DOCS` directory.

In Equations on a Dense Dimension

When you use the cross-dimensional operator in an equation on a dense dimension, Essbase does not automatically create the required blocks if the resultant values are from a dense dimension and the operand or operands are from a sparse dimension.

`result = member->operand`

Where:

`result` is from a dense dimension and *`operand`* is from a sparse dimension.

Consider the following example from Sample Basic in which you want to create budget sales and expenses data from existing actual data.

Sales and Expenses are members on the dense Measures dimension. Budget and Actual are members on the sparse Scenario dimension.

```
FIX(Budget)
      (Sales = Sales->Actual * 1.1;
       Expenses = Expenses->Actual * .95;)
ENDFIX
```

The above calc script does *not* create the required data blocks. Budget data values are not calculated for any blocks that do not already exist. The resultant values are dense dimension members (Sales and Expenses). The operand is a sparse dimension member (Actual).

You can solve the problem 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
```

Essbase then copies the data and creates the required blocks. Essbase creates blocks containing the Budget values for each corresponding Actual block that already exists.

Alternatively, you can avoid copying the data 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 containing the dense member equations:

```
FIX(Sales, Expenses)
Budget (Sales = Sales->Actual * 1.1;
      Expenses = Expenses->Actual * .95;)
ENDFIX
```

Calc Script Techniques

You may achieve significant calculation performance improvement by carefully grouping formulas and dimensions in your calc script. For more information, see Chapter 30, Developing Calc Scripts. In this way, you can ensure that Essbase cycles through the data blocks in your database as few times as possible during a calculation. For more information on calculation passes, see Chapter 27, Defining the Calculation Order.

Aim to make your database calculation as simple as possible. If possible, consider applying all your formulas to the database outline and using a default calculation (CALC ALL). This may improve calculation performance.

Setting the Memory Cache Sizes

When calculating your database, Essbase uses approximately 30 bytes of memory per member in the database outline. So if your database has 5,000 members, Essbase needs approximately 150K of memory to calculate the database.

When running concurrent calculations, each calculation uses separate memory space. For example, if you are running two calc scripts concurrently, Essbase requires 60 bytes per member. Essbase needs 30 bytes per member for each calculation.

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 your database outline. However, you can ensure that the memory caches sizes allow Essbase to optimize the calculation.

Essbase uses three memory caches to coordinate memory usage:

- The calculator cache
- The index cache
- The data cache

Ensure that the calculator cache is large enough to optimize your calculation performance. For more information, see the following sections.

Calculator Cache

Essbase uses the calculator cache to create and track data blocks during calculation. Using the calculator cache significantly improves your calculation performance, particularly if you are calculating your database for the first time, or when the data in your database is very sparse. The size of the performance improvement depends on the configuration of your database.

You can use the default calculator cache size, or you can set the size of the calculator cache within a calc script for the duration of the calc script. For more information, see the calc script SET CACHE command and the CALCCACHE configuration command in the online *Technical Reference* in your `DOCS` directory.

The maximum calculator cache size that you can specify is 200,000,000 bytes. The default, if you do not set the calculator cache, is 200,000 bytes.

The size of the calculator cache you choose depends on the amount of memory your system has available and the configuration of your database.

Essbase uses the calculator cache providing that:

- Your database has at least two sparse dimensions.
- You calculate at least one, full sparse dimension (unless you specify the SET CACHE ALL option in a calc script; in which case, Essbase uses a calculator cache, even when you do not have at least one, full sparse dimension).

Calculator Cache Options

For the calculator cache, Essbase separates your database's sparse dimensions into two groups:

Calculator Cache "Anchoring" dimensions	The last one or the last two sparse dimensions in your database outline.
Bitmap dimensions	The remaining sparse dimensions in your database outline. Each member combination of these sparse dimensions occupies 1 bit of memory.

The calculator cache then uses one of two methods:

Single Bitmap cache	Uses the least memory, but is less efficient than the Multiple Bitmap method. The calculator cache uses this method if there is more than one calculator Cache "Anchoring" dimension or if the calculator cache is not large enough to support the Multiple Bitmap cache method.
Multiple Bitmap cache	The optimal method. It uses more memory, but is more efficient than the Single Bitmap method. The performance improvement is particularly high when you are calculating your database for the first time. The calculator cache can use this method only if there is a single calculator Cache "Anchoring" dimension. The number of bitmaps used is determined by the maximum number of dependent parents for any of the members on the calculator Cache "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 have more than two dependent parents on this dimension. Therefore, if Product is the calculator Cache "Anchoring" dimension, the maximum number of dependent parents is 2.

Depending on the calculator cache size you specify, Essbase chooses one of three options:

Option	Performance	Method
1	1	Single calculator Cache “Anchoring” dimension Multiple Bitmap cache
2	2	Single calculator Cache “Anchoring” dimension Single Bitmap cache
3	3	Multiple calculator Cache “Anchoring” dimensions Single Bitmap cache

Essbase chooses the optimal performance method for your database calculation, according to the size of the calculator cache available.

If the calculator cache size is too small for any of the above options, Essbase does not use a calculator cache. Your calculation performance may be significantly impaired.

Calculating the Required Calculator Cache for Your Database

The size of the calculator cache you choose depends on the amount of memory your system has available. It also depends on the nature and configuration of your database.

You can calculate the calculator cache size required for each of the above three options.

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

The calculator cache size required for each of the three options can be calculated as follows:

Option 1:	
Bitmap dimensions	S1, S2, S3, S4
Calculator Cache “Anchoring” dimension	S5
Dependent parents on calculator Cache “Anchoring” dimension	3

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 on the calculator Cache “Anchoring” dimension + 2 constant Bitmaps
	=	3 + 2
	=	5
Calculator cache	=	Bitmap size * Number of Bitmaps
	=	125,000 * 5
	=	**625,000 bytes**

Option 2:	
Bitmap dimensions	S1, S2, S3, S4
Calculator Cache “Anchoring” dimension	S5
Dependent parents on calculator Cache “Anchoring” dimension	Not applicable

Bitmap size in bytes	=	(S1 * S2 * S3 * S4)/8
	=	(20 * 20 * 50 * 50)/8
	=	125,000 bytes
Number of Bitmaps	=	Single Bitmap cache
	=	1
Calculator cache	=	Bitmap size * Number of Bitmaps
	=	125,000 * 1
	=	**125,000 bytes**

Option 3:	
Bitmap dimensions	S1, S2, S3
Calculator Cache “Anchoring” dimension	S4, S5
Dependent parents on calculator Cache “Anchoring” dimension	Not applicable

Bitmap size in byte	=	(S1 * S2 * S3)/8
	=	(20 * 20 * 50)/8
	=	20,000 bytes
Number of Bitmaps	=	Single Bitmap cache
	=	1
Calculator cache	=	Bitmap size * Number of Bitmaps
	=	20,000 * 1
	=	**20,000 bytes**

Conclusion

In the above example, if you specify a calculator cache of at least:

625,000 bytes	Essbase uses Option 1, which provides optimal calculator cache performance.
125,000 bytes	Essbase uses Option 2.
20,000 bytes	Essbase uses Option 3.

If you specify a calculator cache size of less than 20,000 bytes, Essbase does not use a calculator cache during the calculation. The calculation performance may be significantly impaired.

You can check which calculator cache option Essbase is able to use on your database using the SET MSG SUMMARY command in a calc script. Run the following calc script on your empty database.

```
SET MSG SUMMARY;
CALC ALL;
```

Essbase displays the calculator cache setting in the ESSCMD window or in the Event Log file. For more information, see “Monitoring and Tracing Calculations” on page 32-4.

The size of the calculator, index, and data caches generally has more effect on performance if your database calculation is based mainly on aggregations rather than formula calculations.

Calculating Your Database for the First Time

If you are calculating your database for the first time, the size of the calculator cache is particularly significant for calculation performance.

If possible, ensure that the calculator cache size is large enough for Essbase to use the optimal calculator cache option. For more information, see “Calculator Cache Options” on page 32-13.

Optimizing Use of the Calculator Cache for Large, Flat Database Outlines

You can use the SET CALCHASHTBL command to optimize how Essbase uses the calculator cache when calculating large, flat database outlines (for example, where one member has more than 5000 children).

When you enable this feature, Essbase uses a hash table to optimize use of the calculator cache for large, flat databases. A large, flat database is, for example, a database in which one or more members has over 5000 children. Using this feature may significantly improve the performance of a CALC ALL of the database or 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 online *Technical Reference* in your `DOCS` directory.

Index and Data Caches

The index cache needs to be large enough to reduce the need to keep writing the index pages to the disk. If your database is large, the default index cache is not large enough to provide optimum calculation performance.

For information on sizing your index and data caches, see Chapter 14, Sizing Your Database.

Block Locking During Calculation

When a block is calculated, Essbase gets addressability to the block along with the blocks containing its children. Essbase calculates the block and then releases it along with the blocks containing its children.

By default, Essbase allows the Calculator to get addressability to up to 100 blocks concurrently when calculating a block. This is sufficient for most database calculations. If you are calculating a formula on a sparse dimension, Essbase works most efficiently if it can get addressability to all the required children concurrently.

Therefore, when calculating a formula on a sparse dimension, you may want to set a number higher than 100 if you are consolidating very large numbers of children in a calculation (for example, more than 100 children).

By increasing the number, you ensure that Essbase can get addressability to all the required blocks when calculating a data block, and that performance is not impaired.

You can use the SET LOCKBLOCK command in a calc script and the CALCLOCKBLOCK setting in the `ESSBASE.CFG` file to specify the maximum number of blocks that Essbase can get addressability to concurrently when calculating a block. For more information, see the online *Technical Reference* in your `DOCS` directory.

Note: For aggregations on a sparse dimension, this is not a consideration because Essbase does not need to get addressability to all the children concurrently.

Essbase locking behavior depends on the Isolation Level setting. See Chapter 41, Ensuring Data Integrity, for more information.

Multi-User Considerations

Essbase uses a block locking system to provide concurrent access to users. This system ensures that only one user can update or calculate a particular data block at any time. How Essbase handles block locking and committing data depends on the Storage Manager Isolation Level setting. For more information, see Chapter 41, Ensuring Data Integrity.

When Essbase calculates a data block, it gets addressability to the block with an exclusive lock. This can mean that 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.

Multi-Users and Calculation Performance

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, then the calculation waits for one of the following, depending on your Storage Manager Isolation Level setting:

- For the data block to be released (Uncommitted Access Isolation Level setting)
- For the calculation to complete (Committed Access Isolation Level setting)

Essbase does not provide a message to say that the calculation is waiting for the data block to be released.

You can prevent the calculation delays caused by waiting for locked blocks by using Essbase security options to:

- Deny access to other users
- Disconnect users from Essbase

For more information on these security options, see Part III, Designing and Building a Security System. For information on how Essbase handles locks and transactions, see Chapter 41, Ensuring Data Integrity.

Note: When Essbase gets addressability to a data block for calculation, it does not put an exclusive lock on the dependent child blocks. This means that another user could update values in the child blocks. If necessary, you can use the above security options to prevent this happening.

Using Two-Pass Calculation

In some cases, you can achieve a significant performance improvement by tagging an Accounts dimension member formula as Two Pass in the database outline rather than repeating the formula in a calc script. In other cases, you may need to use a calc script to calculate some formulas twice.

Note: You can achieve a performance improvement by tagging Two-Pass members as Dynamic Calc. You can tag these members as Two Pass, even if the members are *not* on the Accounts dimension. For more information, see Chapter 28, Dynamically Calculating Data Values.

Some member formulas need to be calculated twice to produce the required value. For example, consider the calculation required for Profit%, where `Profit% = Profit % Sales`.

The following example 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).

For detailed information on the cell calculation order depending on your database configuration, see Chapter 27, Defining the Calculation Order.

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.

Measures/Year	**Jan**	**Feb**	**Mar**	**Qtr1**
Profit	75	50	120	**5**
Sales	150	200	340	**6**
Profit%	**1**	**2**	**3**	**4/7**

The calculation order is as follows:

1. Essbase calculates the formula `Profit % Sales` for Profit%->Jan, Profit%->Feb, Profit%->Mar, and Profit%->Qtr1.
2. Essbase calculates Profit->Qtr1, and Sales->Qtr1 by adding the values for Jan, Feb, and Mar.
3. Essbase calculates Profit%->Qtr1 by adding the values for Profit%->Jan, Profit%->Feb and Profit%->Mar. This addition of percentages does not produce the correct result.

Measures/Year	Jan	Feb	Mar	Qtr1
Profit	75	50	120	**245**
Sales	150	200	340	**690**
Profit%	**50%**	**25%**	**50%**	**125%**

4. When Profit% is tagged as Two Pass in the database outline, Essbase recalculates the Profit% values using the `Profit % Sales` formula to produce the correct results.

Measures/Year	Jan	Feb	Mar	Qtr1
Profit	75	50	120	**245**
Sales	150	200	340	**690**
Profit%	**50%**	**25%**	**50%**	**36%**

Notes:

- You can only tag a member as Two Pass if it is on a dimension tagged as Accounts, unless it is a Dynamic Calc or Dynamic Calc And Store member. You can tag Dynamic Calc and Dynamic Calc And Store members as Two Pass, even if the members are *not* on the Accounts dimension. For more information, see Chapter 28, Dynamically Calculating Data Values.
- When you perform a default calculation on your database, Essbase automatically recalculates any formulas tagged as Two Pass on the dimension tagged as Accounts in your database outline.

- In some situations, you may need to use a calc script to calculate the Two-Pass formulas. For example, if your database configuration means that Essbase uses Scenario A, see “Scenario A” on page 32-24 and the formula references values from another data block. For more information, see “Using a Calc Script for Two-Pass Calculations” on page 32-26.

  In other situations, you may want to use a calc script to ensure efficient use of Intelligent Calculation. For more information, see “Using Two Pass on a Default Calculation” on page 32-22.
- When you use a calc script, Essbase does not automatically recalculate Two-Pass formulas. You need to use the `CALC TWOPASS;` command. If you are using Intelligent Calculation, consider the implications of marking data blocks as *clean*. For more detailed information, see “Using a Calc Script for Two-Pass Calculations” on page 32-26.

Using Two Pass on a Default Calculation

When you perform a default calculation on your database, with Two-Pass calculation enabled (the default), Essbase automatically attempts to calculate any formulas tagged as Two Pass on the dimension tagged as Accounts in your database outline. This is true even if you have customized your default calc script.

You can perform a default calculation in:

- ESSCMD
- Application Manager (by selecting Database | Calculate and choosing Default).

To enable Two-Pass Calculation in Application Manager:

1. Choose Database | Settings. Essbase displays the **Database Settings** dialog box.
2. Check **Two Pass Calculation** on the **General** page.
3. Click OK.

However, in some situations, you may need to use a calc script to calculate the Two-Pass formulas. For more information, see “Using a Calc Script for Two-Pass Calculations” on page 32-26.

Essbase recognizes formulas on members tagged as Two Pass only on a dimension tagged as Accounts, unless the member is a Dynamic Calc or Dynamic Calc And Store member. You can tag Dynamic Calc and Dynamic Calc And Store members as Two Pass, even if the members are *not* on the Accounts dimension. For more information, see Chapter 28, Dynamically Calculating Data Values.

When you are doing a default calculation of your database, you need to ensure that the Two-Pass Calculation option is selected in the **Database Settings** dialog box in Application Manager. Essbase then does the Two-Pass calculation as part of the default calculation.

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. This means that Essbase does not need to do an extra calculation pass through your database. However, in some situations, Essbase needs to calculate the Two-Pass formulas on an extra calculation pass through your database. For information on calculation passes, see Chapter 27, Defining the Calculation Order.

How Essbase calculates the Two-Pass formulas depends on whether you have a dimension tagged as Time as well as a dimension tagged as Accounts. It also depends on the dense/sparse configuration of these dimensions.

This table summarizes two different scenarios, which are described in detail in the following sections. If you are using Intelligent Calculation, consider the scenario that matches your database configuration; it tells you how to ensure that Essbase accurately calculates Two-Pass formulas.

The following information assumes that you understand the concepts of Intelligent Calculation. For more information, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

Dimension Tagged As...		
Accounts	**Time**	**Scenario**
Dense	None	A
Sparse	None	B
Dense	Dense	A
Dense (and is the only dense dimension)	Sparse	A
Dense (and there are other dense dimensions)	Sparse	B
Sparse	Sparse	B
Sparse	Dense	B

Scenario A

In summary:

- No extra calculation pass for Two-Pass formulas
- All data blocks marked as *clean*

☞ Scenario A is efficient; place formulas in your outline and tag as Two Pass.

No Extra Calculation Pass for Two-Pass Formulas
Essbase calculates the Two-Pass formulas while it is calculating the data block. This means that Essbase does not need to do an extra calculation pass through your database.

All Data Blocks Marked As *Clean*
After the calculation Essbase has marked all the data blocks as *clean* for the purposes of Intelligent Calculation. For more information, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

If you have changed your default calculation from the default `CALC ALL;` the data blocks may not all be marked as *clean* after a default calculation. For more information, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation. You can check the default calculation setting by selecting Database | Set Default Calc in Application Manager.

If your database configuration means that Essbase can use Scenario A, you can achieve a performance improvement by tagging a member formula as Two Pass in the outline rather than repeating the formula in a calc script. 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 calc script, Essbase has to read and write the data blocks to memory in order to recalculate the formula.

Scenario B

In summary:

- Extra calculation pass for Two-Pass formulas
- Data blocks representing Two-Pass formulas are *not* marked as *clean*

☞ Consider using a calc script to calculate Two-Pass formulas

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 the data blocks are marked as *clean* after the first database calculation, Essbase ignores the *clean* status on the blocks representing the Two-Pass formulas, and recalculates these blocks.

Data Blocks Representing Two-Pass formulas Are *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 this is a partial calculation of the database, Essbase does not mark these blocks as *clean*. When you recalculate your database with Intelligent Calculation turned on, these data blocks may be recalculated unnecessarily.

If you have changed your default calculation from the default `CALC ALL;` the data blocks may not be marked as *clean* after the first calculation. For more information, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation. You can check the default calculation setting by selecting Database | Set Default Calc in Application Manager.

If your database configuration means that Essbase uses Scenario B, consider using a calc script to perform your Two-Pass formula calculations. If you use a calc 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 “Using a Calc Script for Two-Pass Calculations” on page 32-26.

Using a Calc Script for Two-Pass Calculations

You need to use a calc script to calculate a formula twice if your database configuration means that Essbase uses Scenario A, as described in "Using Two Pass on a Default Calculation" on page 32-22, and the formula references values from another data block.

You may want to use a calc script to calculate Two-Pass formulas if your database configuration means that Essbase uses Scenario B, as described in "Using Two Pass On a Default Calculation."

Consider an example in which your dimension tagged as Accounts is dense. You want to calculate sales for each product as a percentage of sales for all products. The formula might be `Sales % Sales->Product`.

When Essbase calculates the data block for each product, it has not yet calculated the value Sales->Product so the results are incorrect.

If you have Intelligent Calculation turned on (the default), Essbase calculates only those data blocks that are not marked as *clean*. When you perform a default calculation of your database with Intelligent Calculation turned on, all the data blocks are marked as *clean*. Therefore, before you recalculate the formula, you need to turn off Intelligent Calculation.

To calculate the correct results for Sales % you can choose one of two options, depending on whether you want to obtain the performance benefits of Intelligent Calculation when performing the first, full calculation of your database.

Option 1: Using Intelligent Calculation

If your index is quite large, and you want to have the benefit of using Intelligent Calculation, you can:

- **Use a calc script to calculate the formula:** Run a calc script to perform a full calculation of the database with Intelligent Calculation turned on. This marks all the data blocks as *clean*. Then, in the calc script, turn off Intelligent Calculation, tell Essbase to mark the calculated data blocks as *clean*, and calculate the formula.

```
SET UPDATECALC ON;
CALC ALL;
SET UPDATECALC OFF;
SET CLEARUPDATESTATUS AFTER;
"Share of Sales" = Sales % Sales->Product;
```

- **Tag the member as Two Pass, and use a calc script to calculate Two-Pass members:** Place the formula in the database outline and tag it as Two Pass. Place the formula on the Share of Sales member on the dimension tagged as Accounts. In this case, the required calc script is:

  ```
  SET UPDATECALC ON;
  CALC ALL;
  SET UPDATECALC OFF;
  SET CLEARUPDATESTATUS AFTER;
  CALC TWOPASS;
  ```

- **Do a default calculation and then use a calc script to calculate the formula:** Perform the full calculation in Application Manager by choosing Database|Calculate from the Application Manager menu, and choosing Default (providing that you have not customized the default calc script), or from ESSCMD. Ensure that Intelligent Calculation is turned on when you calculate the database. Then, use the second part of the calc script to calculate the formula, as follows:

  ```
  SET UPDATECALC OFF;
  SET CLEARUPDATESTATUS AFTER;
  "Share of Sales" = Sales % Sales->Product;
  ```

  Or:

  ```
  SET UPDATECALC OFF;
  SET CLEARUPDATESTATUS AFTER;
  CALC TWOPASS;
  ```

Essbase performs the following calculations during these calc scripts:

1. The SET UPDATECALC ON command turns on Intelligent Calculation.
2. The CALC ALL commands calculates the database and marks the data blocks as *clean*.
3. The SET UPDATECALC OFF command turns off Intelligent Calculation.
4. The SET CLEARUPDATESTATUS AFTER command tells Essbase to mark the calculated blocks as *clean*, even though this is a partial calculation of the database. (If you do not use the SET CLEARUPDATESTATUS AFTER command, Essbase marks data blocks as *clean* only after a full calculation of the database.)
5. Essbase cycles through the database calculating the formula for Share of Sales or calculating only those formulas tagged as Two Pass in the database outline.

Note: For more information on Intelligent Calculation, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation. For more information on developing formulas and calc scripts, see Chapter 25, Developing Formulas, and Chapter 30, Developing Calc Scripts.

Option 2: Using Intelligent Calculation

If your index is small, and you want to have the benefit of using Intelligent Calculation, you can run a calc script to calculate your database, but tell Essbase not to mark the calculated data blocks as clean. Then mark all the data blocks as *clean*, but do *not* calculate the data blocks.

```
SET CLEARUPDATESTATUS OFF;
CALC ALL;
CALC TWOPASS;
SET CLEARUPDATESTATUS ONLY;
CALC ALL;
```

Essbase performs the following calculations:

1. The SET CLEARUPDATESTATUS OFF command tells Essbase not to mark the calculated data blocks as clean.
2. The CALC ALL command causes Essbase to cycle through the database calculating all the *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. Essbase cycles through the database recalculating the those formulas tagged as Two Pass on 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 these calculated 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 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.

Option 3: Without Using Intelligent Calculation

Use a calc script to turn off Intelligent Calculation, perform a full calculation, and repeat the formula at the end of a calc script as follows:

```
SET UPDATECALC OFF;
CALC ALL;
"Share of Sales" = Sales % Sales->Product;
```

Calculating #MISSING Values

If no data value exists for a unique combination of dimension members, then Essbase gives the combination a `#MISSING` value. This is different from a zero (0) value.

Essbase treats `#MISSING` values differently from 0 values. It calculates them in the following way:

- Multiplying or dividing a `#MISSING` value returns a `#MISSING` value.
- Adding or subtracting a `#MISSING` value returns a non-missing value, unless all the values in the equation are `#MISSING` values.

By default Essbase does not aggregate `#MISSING` values. The AGGMISSG setting is turned off. This is important when you need to load data at parent levels. For more information, see "Loading Data at Parent Levels" on page 32-30.

You can change the way Essbase aggregates `#MISSING` values in Application Manager or in a calc script.

In Application Manager:

1. Choose Database | Settings. Essbase displays the **Database Settings** dialog box.
2. Check **Aggregate Missing Values**.
3. Click OK.

In a calc script, use the SET AGGMISSG command. For more information, see the online *Technical Reference* in your `DOCS` directory.

If you never load data at parent levels, then you can achieve a performance improvement by aggregating `#MISSING` values. To do this, turn on the AGGMISSG setting. The amount of performance improvement you achieve depends on the ratio of Upper Level and Input blocks in your database. For more information see Chapter 27, Defining the Calculation Order.

Note: Turning on the AGGMISSG setting changes the cell calculation order within a data block. For more information, see Chapter 27, Defining the Calculation Order.

When the AGGMISSG setting is turned off, 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 on 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 your database configuration or how you load data.

Loading Data at Parent Levels

Sometimes you may need to load data into cells representing parent members of one or more dimensions.

For example, in the Sample Basic database you might want to load Miscellaneous Expenses (Misc) for each product at the consolidated product level. You would load Miscellaneous Expenses for Colas (100), Root Beer (200), Cream Soda (300) and Fruit Soda (400) rather than for each individual product.

When you do this, you do not want the Miscellaneous Expenses data to be overwritten when Essbase aggregates the `#MISSING` values in the children of each product type. By default Essbase does not aggregate `#MISSING` values and so the data is not overwritten.

However, if you always load data at level 0 and never at parent levels, then you should choose to aggregate missing values by turning on the AGGMISSG setting. This provides a calculation performance improvement of between 1% and 30%. The performance improvement varies, depending on your database size and configuration.

For more information on aggregating `#MISSING` values, see "Calculating #MISSING Values" on page 32-29, Chapter 27, Defining the Calculation Order, and the online *Technical Reference* in your `DOCS` directory.

Warning: Ensuring that the AGGMISSG setting is off (the default) protects parent-level data only if the child member combinations have `#MISSING` values. If the child member combinations have any other values, including zeros (0), then Essbase aggregates the child values and overwrites the parent values.

Chapter 33 Optimizing Your Calculation Using Intelligent Calculation

This chapter provides information on how to optimize the performance of your Hyperion Essbase calculations using Intelligent Calculation. For additional information on optimizing your overall database calculations, see:

- Chapter 32, Optimizing Your Calculations
- Chapter 28, Dynamically Calculating Data Values.
- Chapter 6, Designing Partitioned Applications

This chapter includes the following sections:

Why Use Intelligent Calculation?

If you are doing a full calculation of your database, in most cases you can obtain improved calculation performance by turning on Intelligent Calculation (the default). For more information, see “Turning Intelligent Calculation On and Off” on page 33-15.

If you cannot use Intelligent Calculation for all of your database calculation, you may be able to use it for part of the calculation. For example, consider a case in which you calculate your database by doing a default consolidation and then an allocation of data. In this case, turn on Intelligent Calculation for the default consolidation and then turn off Intelligent Calculation for the allocation to significantly improve your calculation performance.

Assuming that Intelligent Calculation is turned on (the default), use a calc script to:

1. Do a CALC ALL of your database with Intelligent Calculation turned on.
2. Turn off Intelligent Calculation for the duration of the calc script using the SET UPDATECALC command.
3. Allocate Headquarters Costs across all products.

For more information on turning on and turning off Intelligent Calculation, see “Turning Intelligent Calculation On and Off” on page 33-15.

If you want to use Intelligent Calculation when calculating a subset of your database, or when performing multiple calculation passes through your database, consider carefully the implications of marking data blocks as *clean*.

Carefully maintain the *clean* and *dirty* status of the data blocks, to ensure that Essbase recalculates your database as efficiently as possible using Intelligent Calculation.

For example, when you calculate a subset of your 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* using the SET CLEARUPDATESTATUS AFTER command in a calc script.

For more information see “Using the SET CLEARUPDATESTATUS Command” on page 33-6, and the online *Technical Reference* in your `DOCS` directory.

Introducing Intelligent Calculation

When you do a full calculation of your database, Essbase tracks which data blocks it has calculated. This means that on subsequent calculations, after you have loaded a small subset of data, you can choose to calculate *only* those data blocks that Essbase has *not* calculated or that require recalculating. This makes the calculation more efficient. Essbase uses Intelligent Calculation to do this.

Intelligent Calculation is designed to provide significant performance benefits for a full calculation (CALC ALL), or when you use a calc script that calculates all members in *one* CALC DIM command. For more information, see "How Intelligent Calculation Works" on page 33-3.

If you want to use Intelligent Calculation when calculating a subset of your database, or to perform multiple calculation passes through your database, consider carefully the implications of how Essbase marks data blocks as *clean*. To ensure accurate calculation results, you need to consider the information in "Using the SET CLEARUPDATESTATUS Command."

By default, Intelligent Calculation is turned on. You can change this default setting in the `ESSBASE.CFG` file. 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 33-15. For more information on the `ESSBASE.CFG` file, see the online *Technical Reference* in your `DOCS` directory.

How Intelligent Calculation Works

The data blocks in your database have a calculation status of either *clean* or *dirty*.

Essbase marks a data block as *clean* when it calculates the data block on a full calculation (CALC ALL).

More precisely, Essbase marks data blocks as *clean* in the following calculations:

- A full calculation (CALC ALL) of your database. Unless you have changed it, CALC ALL is the default calculation for your database. You can check the default calculation setting by choosing Database | Set Default Calc in Hyperion Essbase Application Manager.

- A calc script that calculates all the dimensions in *one* CALC DIM statement. For example, the following calc script calculates all the members in the Sample Basic database:

  ```
  CALC DIM(Measures, Product, Market, Year, Scenario);
  ```

  However, note that the following calc script calculates all the members, but 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*:

  ```
  CALC DIM(Measures, Product);
  CALC DIM(Market, Year, Scenario);
  ```

  When a calc script causes Essbase to do two or more calculation passes through the database, you need to consider carefully the effects of Intelligent Calculation. For more information, see "Using the SET CLEARUPDATESTATUS Command" on page 33-6. For information on calculation passes, see Chapter 27, Defining the Calculation Order.

Essbase does *not* mark calculated data blocks as *clean* in any calculations other than the situations described above, unless you use the `SET CLEARUPDATESTATUS` command in your calc script. For more information, see "Using the SET CLEARUPDATESTATUS Command" on page 33-6.

Essbase marks a data block as *dirty* when:

- It calculates a data block. If the data block is part of a full calculation, or you are using the SET CLEARUPDATESTATUS AFTER command, Essbase then marks the data block as *clean*.
- You load data into a data block.
- You restructure your database (for example, by adding a member to a dense dimension).
- You copy data to the data block.

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 in a data block, the whole data block is marked as *dirty*.

Using Intelligent Calculation for a Default, Full Calculation

Intelligent Calculation generally provides significant performance benefits when you do a full calculation (CALC ALL) of your database. If you do a full calculation of your database, you can leave Intelligent Calculation turned on (the default) to take advantage of the performance benefits it provides.

Unless you have changed it, a full calculation (CALC ALL) is the default calculation for your database. You can check the default calculation setting by choosing Database | Set Default Calc in Application Manager.

Warning: When using Intelligent Calculation, note the information in "Limitations" on page 33-16.

Calculating for the First Time

When you do a full calculation of your 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 your database with Intelligent Calculation turned on, Essbase checks each block to see if it is marked as *clean* or *dirty*. Checking the data blocks has a 5% to 10% performance overhead. For more information on *clean* and *dirty*, see "How Intelligent Calculation Works" on page 33-3.

During most recalculations, this small overhead is much less than the performance gained by turning on Intelligent Calculation.

However, if you recalculate a database where more than approximately 80% of the values have been changed, the overhead of Intelligent Calculation may outweigh the benefits. In this case, you can turn off Intelligent Calculation. For more information, see "Changing a Formula or Time Series Attribute" on page 33-16.

Using Intelligent Calculation for a Partial, Calc Script Calculation

Essbase marks a data block as *clean* when it calculates the data block on a full calculation (CALC ALL) or when Essbase calculates all dimensions in one CALC DIM command. For more information, see "How Intelligent Calculation Works" on page 33-3.

In any other calculations, Essbase does *not* mark calculated data blocks as *clean*, unless you use the SET CLEARUPDATESTATUS command in your calc script. For example, if you calculate a subset of your database or calculate your 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);
```

Using the SET CLEARUPDATESTATUS Command

The SET CLEARUPDATESTATUS command has three parameters: AFTER | ONLY | OFF. When you use:

SET CLEARUPDATESTATUS AFTER;	Essbase marks calculated data blocks as *clean*, even if you are calculating a subset of your database.
SET CLEARUPDATESTATUS ONLY;	Essbase marks the specified data blocks as *clean*, but does not actually calculate the data blocks. This does the same as AFTER, but disables calculation.
SET CLEARUPDATESTATUS OFF;	Essbase does calculate 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 your database. The existing *clean* or *dirty* status of the calculated data blocks remains unchanged.

Warning: When you use the SET CLEARUPDATESTATUS command to mark calculated data blocks as *clean,* consider carefully the following questions:

Which Data Blocks are Calculated?	Only *calculated* data blocks are marked as *clean*. For more information, see “Calculating Data Blocks” on page 33-8.
Will Concurrent Calculations Affect the Same Data Blocks?	Do not use the SET CLEARUPDATESTATUS AFTER command with concurrent calculations unless you are certain that the different 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 turned on, Essbase may not recalculate the data blocks, because they are already marked as *clean*. For more information, see “Concurrent Calculations” on page 33-10.
Will Essbase Recalculate the Same Data Blocks on a Second Calculation Pass Through Your Database?	If you calculate data blocks on a first calculation pass through your database, Essbase marks them as *clean*. If you then attempt to calculate the same data blocks on a subsequent pass with Intelligent Calculation turned on, Essbase does not recalculate the data blocks, because they are already marked as *clean*. For more information, see “Multiple-Pass Calculations” on page 33-11. For information on Two Pass calculations.

Examples Using SET CLEARUPDATESTATUS

The following examples are based on the Sample Basic database in which the sparse dimensions are Market and Product. The examples assume that Intelligent Calculation is turned on (the default). For more information, see “Changing a Formula or Time Series Attribute” on page 33-16.

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Product);
ENDFIX
```

New York is a member on the sparse Market dimension. 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 based on the Product dimension and marks them as *clean*. 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.

```
SET CLEARUPDATESTATUS ONLY;
FIX("New York")
CALC DIM(Product);
ENDFIX
```

New York is a member on the sparse Market dimension. 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). It marks them as *clean*. It does *not* calculate the data blocks. It does not mark the Level 0 data blocks as *clean* because they are not calculated. For example, if "New York"->100-10 is *dirty*, it remains *dirty*.

```
SET CLEARUPDATESTATUS OFF;
CALC ALL;
CALC TWOPASS;
SET CLEARUPDATESTATUS ONLY;
CALC ALL;
```

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 those members tagged as Two Pass on 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 the 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. 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.

The following information assumes that you are familiar with the concepts of Upper, Level 0 and Input data blocks. 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 without using a FIX command, Essbase calculates at least some of the data values in every data block in your database. For example, the following calc script is based on the Sample Basic database:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Year);
```

This calculates the Year dimension, which is a dense dimension. Because Year is dense, every data block in the database represents members in the Year dimension. Therefore, Essbase calculates data values in every data block. Because you are using the SET CLEARUPDATESTATUS AFTER command, Essbase marks all the data blocks as *clean*.

Calculating a Sparse Dimension

When you calculate a sparse dimension, Essbase might not need to calculate every data block in your database. For example, the following calc script is based on the Sample Basic database:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product);
```

This calculates the Product dimension, which is a sparse dimension. Because it is sparse, separate data blocks exist for each member on the Product dimension. For example, one data block exists for "New York"->Colas and another for "New York"->100-10. The data block "New York"->100-10 is a Level 0 block, which means that it does not represent a parent member on either sparse dimension. 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. It does not mark it as *clean*, even though you are using the SET CLEARUPDATESTATUS AFTER command.

The upper level data block "New York"->Colas represents Colas, which is a parent level member on the Product dimension. Essbase needs to calculate values for Colas, so Essbase calculates this data block. Because you are using the SET CLEARUPDATESTATUS AFTER command, Essbase marks this data block as *clean*.

When Essbase calculates a sparse dimension, it does not calculate the Level 0 data blocks (unless a formula is applied to one of the sparse Level 0 members). Because it does not calculate these Level 0 data blocks, they are *not* marked as *clean*, even when you are using the SET CLEARUPDATESTATUS AFTER command.

When Essbase calculates a sparse dimension, it recalculates an upper level data block if the block is dependent on one or more child blocks that are *dirty*.

If you load data into your 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, these Level 0 blocks remain *dirty*, because Essbase does not calculate them. Therefore, when you recalculate only a sparse dimension or dimensions, Essbase recalculates all the 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*.

You can avoid this by ensuring that you calculate at least one dense dimension; calculating a dense dimension without using the FIX command calculates data values in every data block, including the Level 0 blocks. So the Level 0 blocks are marked as *clean*.

Concurrent Calculations

If concurrent calculations attempt to calculate the same data blocks, with Intelligent Calculation 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 different 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. If User One runs this first calc script:

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York", Actual)
CALC DIM(Product, Year);
ENDFIX
```

Essbase calculates the Actual values for all data blocks that represent "New York". Essbase marks the calculated data blocks as *clean*, even though all the data values in each calculated block have not been calculated. For example, the Budget values have not yet been calculated.

If User Two runs this second calc script to calculate the Budget values for "New York":

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York", Budget)
CALC DIM(Product, Year);
ENDFIX
```

Essbase does not recalculate the specified data blocks, because they are already marked as *clean*. The calculation results are not correct.

One way to solve this problem is to make the Scenario dimension sparse; then the Actual and Budget values would be in different data blocks. For example, "New York"->Colas->Actual and "New York"->Colas->Budget. In this case, the second calc script would correctly calculate the data blocks.

Multiple-Pass Calculations

Whenever possible, Essbase calculates your 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 your database, the number of calculation passes Essbase performs depends upon your calc script. For more information, see “How Intelligent Calculation Works” on page 33-3, and for more information on grouping formulas and calculations, see Chapter 30, Developing Calc Scripts.

Consider a situation in which you calculate data blocks on a first calculation pass through your database, and Essbase marks them as *clean*. If you then attempt to calculate the same data blocks on a subsequent pass with Intelligent Calculation turned on, Essbase does not recalculate the data blocks, because they are already marked as *clean*.

The following examples describe situations in which you would obtain incorrect calculation results. They provide solutions to each of the situations. These examples are based on the Sample Basic database and assume that Intelligent Calculation is turned on.

Example 1

Consider the following calc script:

```
CALC ALL;
CALC TWOPASS;
```

Essbase calculates the *dirty* data blocks in your database and marks all the data blocks as *clean*. Essbase then needs to recalculate those members tagged as Two Pass on 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.

You can calculate the correct results by turning off Intelligent Calculation for the Two-Pass calculation.

Example 2

This calc script calculates data values for "New York" based on the Product dimension:

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Product);
ENDFIX
CALC TWOPASS;
```

Essbase performs the following calculations:

1. The SET CLEARUPDATESTATUS AFTER command tells Essbase to mark the calculated blocks as *clean*, even though this is a partial calculation of the database.
2. Essbase cycles through the database calculating the *dirty* data blocks that represent "New York". It calculates based on the Product dimension. This means that it calculates only those blocks representing a parent member on the Product dimension (for example, "New York"->Colas, "New York"->"Root Beer" and "New York"->"Fruit Soda"). It calculates only the aggregations and formulas for the Product dimension. Essbase marks the calculated data blocks as *clean*, even though all the data values in each calculated block have not been calculated.
3. Essbase needs to recalculate those members tagged as Two Pass on 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.

You can calculate the correct results by turning off Intelligent Calculation for the Two-Pass calculation. For detailed information on using Two-Pass calculations, see Chapter 27, Defining the Calculation Order.

Example 3

This calc script calculates the database based 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);
```

Essbase performs the following calculations:

1. The SET CLEARUPDATESTATUS AFTER command tells Essbase to mark the calculated blocks as *clean*, even though this is a partial calculation of the database.
2. Essbase cycles through the database calculating the *dirty* data blocks based on the Product dimension, as in Example 2. Essbase marks the calculated data blocks as *clean*, even though all the data values in each calculated block have not been calculated.
3. Essbase needs to recalculate the data blocks based on the Year dimension. 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.

You can calculate the correct results by using one CALC DIM command to calculate the Product and Year dimensions. Essbase then calculates both dimensions in one calculation pass through the database. The following calc script would calculate 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

Consider another example, which calculates data values for "New York", but calculates based on two different dimensions. The first calc script calculates the Product dimension:

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Product);
ENDFIX
```

Essbase calculates the data blocks that represent "New York". It calculates based on the Product dimension. This means that it calculates only those *dirty* blocks representing a parent member on the Product dimension (for example, "New York"->Colas, "New York"->"Root Beer" and "New York"->"Fruit Soda"). It calculates only the aggregations and formulas for the Product dimension. Essbase marks the calculated data blocks as *clean*, even though all the data values in each calculated block have not been calculated.

The second calc script is intended to calculate the Year dimension:

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Year);
ENDFIX
```

Essbase calculates the data blocks that represent "New York". It calculates based on the Year dimension. Year is a dense dimension, which means that Essbase needs to calculate all data blocks that represent "New York". Within each of these data blocks it calculates based on the Year dimension. It calculates only the aggregations and formulas for the Year dimension.

However, some of the data blocks for "New York" are already marked as *clean* from the previous calculation. Essbase does not recalculate these data blocks because they are already marked as *clean*. The calculation results are not correct.

You could calculate the correct results by telling Essbase *not* to mark the calculated data blocks as *clean*. The following calc script would calculate 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 tells Essbase to calculate the *dirty* data blocks, but *not* to mark them as *clean* after the first calc script. The SET CLEARUPDATESTATUS AFTER command tells Essbase to mark the data blocks as *clean* after the second calc script.

This assumes that the data blocks are *not* already marked as *clean* from a previous partial calculation of your database.

You can ensure that all data blocks are calculated, irrespective of their *clean* or *dirty* status, by turning off Intelligent Calculation. The following calc script calculates all the 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 the calculated data blocks as *clean*.

Turning Intelligent Calculation On and Off

By default, Intelligent Calculation is turned on. You can change this default using the UPDATECALC setting in the `ESSBASE.CFG` file.

You can turn Intelligent Calculation on and off for the duration of a calc script using the SET UPDATECALC command in a calc script. Turning on Intelligent Calculation means that Essbase calculates only *dirty* blocks and their dependent parents. Turning off 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 online *Technical Reference* in your `DOCS` directory.

Limitations

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 in a data block, the whole data block is marked as *dirty*.
- Changing a formula on the database outline, or changing a Time Series attribute on the database outline, does not cause Essbase to restructure the database. Therefore Essbase does not mark the blocks as *dirty*. You must recalculate the appropriate data blocks. For more information, see "Changing a Formula or Time Series Attribute" on page 33-16.
- Whenever possible, Essbase calculates formulas tagged as Two Pass on the dimension tagged as Accounts as part of the main calculation of your database. However, you may need to use a calc script to calculate some formulas twice. When you use a calc script, you need to turn off Intelligent Calculation before recalculating formulas.

Changing a Formula or Time Series Attribute

Changing a formula on the database outline, or changing a Time Series attribute on the database outline, does not cause Essbase to restructure the database. This means that the data blocks affected by the change are not marked as *dirty*.

Warning: When you subsequently recalculate your database using a default calculation with Intelligent Calculation turned on, then the changes *will not* be calculated.

You must recalculate the appropriate data blocks. You can do this by using a calc script to:

- Turn off Intelligent Calculation and calculate the member formula that has changed.
- Turn off Intelligent Calculation and use the FIX command to calculate the appropriate subset of your database.
- Turn off Intelligent Calculation and perform a default CALC ALL on your database.

For more detailed information, see "Changing a Formula or Time Series Attribute" on page 33-16, and Chapter 30, Developing Calc Scripts.

Considering the Effects of Intelligent Calculation

Using Intelligent Calculation may have implications for the way you administer your database. This section discusses the implications of:

- Index and financial functions
- Restructuring
- Copying and clearing data
- Converting currencies

Index and Financial Functions

If you use index 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 containing the formula, even if it is marked as *clean*.

For more information on index functions and financial functions, see the online *Technical Reference* in your `DOCS` directory.

Restructuring

When you restructure your database (for example, by adding a member to a dense dimension), all the data blocks potentially need recalculating. Therefore, Essbase marks all the data blocks as *dirty*. When you calculate the restructured database, all the blocks are calculated.

Note: Changing a formula on the database outline, or changing a Time Series attribute on the database outline, does not cause Essbase to restructure the database. You must be sure to recalculate the appropriate data blocks. For more information, see “Changing a Formula or Time Series Attribute” on page 33-16.

Copying and Clearing Data

When you copy values to a data block using the DATACOPY command, the resulting data block is marked as *dirty*. Essbase calculates the block when you recalculate your database.

When you clear data values 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 online *Technical Reference* in your `DOCS` directory.

Converting Currencies

When you convert currencies using the CCONV command, the resulting data blocks are marked as *dirty*. Essbase calculates all the converted blocks when you recalculate your database.

For more information on this command, see the online *Technical Reference* in your `DOCS` directory.

Index

This index spans two volumes. Chapters 1–33 are in Volume I; Chapters 34–47 and an Appendix are in Volume II. Examples of entries: 33-10 means Chapter 33, p. 10; A-2 means Appendix A, p. 2.