IBM® DB2® OLAP Server™

Version 7

OLAP Database Administrator's Guide Volume I

P/N: SC27-0788-00

U.S. Patent Number: 5,359,724

Printed in the U.S.A.

Contents

Volume I

Chapter 4: Basic Architectural Elements

Chapter 5: Designing a Single-Server Application

Chapter 6: Designing Partitioned Applications

Part II: Building Hyperion Essbase Applications

Chapter 7: Creating Applications and Databases

Chapter 8: Creating and Changing Database Outlines

Chapter 9: Setting Dimension and Member Properties

Chapter 10: Working with Attributes

Chapter 11: Creating and Managing Aliases

Chapter 12: Linking Objects to Hyperion Essbase Data

Chapter 13: Introducing Dynamic Dimension Building

Chapter 14: Building Dimensions Using a Rules File

Chapter 15: Estimating Disk and Memory Requirements for a Database

Chapter 16: Building and Maintaining Partitions

Part III: Designing and Building a Security System

Chapter 17: Managing Security at Global and User Levels

Chapter 18: Controlling Access to Database Cells

Chapter 19: Security Examples

Part IV: Loading Data

Chapter 20: Introducing Data Loading

Chapter 21: Setting up a Rules File to Manipulate Records

Chapter 22: Manipulating Fields Using a Rules File

Chapter 23: Performing a Data Load

Chapter 24: Debugging and Optimizing Data Loads

Part V: Calculating Data

Chapter 25: Introduction to Database Calculations

Chapter 26: Developing Formulas

Chapter 27: Examples of Formulas

Chapter 28: Defining the Calculation Order

Chapter 29: Dynamically Calculating Data Values

Chapter 30: Calculating Time Series Data

Chapter 31: Developing Calc Scripts

Chapter 32: Examples of Calc Scripts

Chapter 33: Optimizing Calculations

Chapter 34: Using Intelligent Calculation to Optimize Calculation

Volume II

Part VI: Reporting on Data

Chapter 35: Quick Start to Report Scripts

Chapter 36: Developing Report Scripts

Chapter 37: Examples of Report Scripts

Chapter 38: Optimizing Your Reports

Chapter 39: Copying Data Subsets and Exporting Data to Other Programs

Part VII: Managing Multidimensional Hyperion Essbase Data Storage

Chapter 40: Introducing the Hyperion Essbase Kernel

Chapter 41: Specifying Hyperion Essbase Kernel Settings

Chapter 42: Ensuring Data Integrity

Part VIII: Designing and Building Currency Applications

Chapter 43: Designing and Building Currency Conversion Applications

Part IX: Maintaining Hyperion Essbase Applications

Chapter 44: Performing Interactive and Batch Operations Using ESSCMD

Chapter 45: Running Hyperion Essbase, Applications, and Databases

Chapter 46: Using Diagnostics to Monitor Performance

Chapter 47: Working with Hyperion Essbase Files and Cross-Platform Environments

Chapter 48: Backing Up Data and Recovering Databases

Appendix A: Performance-Related Settings

Glossary

Index

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

This guide is primarily for the following:

- Hyperion Essbase database administrators. A Hyperion Essbase database administrator is a person who installs, controls, and maintains a Hyperion Essbase system. A database administrator should have prior experience in networking, system administration, and financial software.
- People who need to use 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 you the following:

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

Use this guide to complete any of the following tasks:

- Learn about Hyperion Essbase architecture, philosophy, and concepts
- Design a multidimensional application
- Define users and security
- Perform data loads and outline updates
- Learn techniques for developing calc and report scripts for advanced applications
- Ensure data integrity
- Optimize a database
- Backup and restore data
- Manage large databases

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

- Volume II
  - Part VI, “Reporting on Data”
  - Part VII, “Managing Multidimensional Hyperion 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 the `DOCS` directory.
- Procedures for installing Hyperion Essbase OLAP Server. For this information, see the *Hyperion Essbase Installation Guide*.
- Basic data retrieval. For this information, you need to use the spreadsheet environment. See the *Hyperion Essbase Spreadsheet Add-in User’s Guide for Excel* or the *Hyperion Essbase Spreadsheet Add-in User’s Guide for 1-2-3*.

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 of the operating systems your clients use. For information on the supported operating systems, see the *Hyperion Essbase Start Here* booklet.
- An understanding of typical database administration requirements and tasks, including designing security, setting up user accounts, and maintaining the Hyperion 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 Hyperion Essbase with current data.
- Knowledge of your business’s data requirements, so you can apply Hyperion Essbase to your specific application.

However, you may simply want to understand the basics of how Hyperion Essbase works or use Hyperion Essbase 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. The examples are based on applications that are provided with the Hyperion Essbase server software and are called Sample, Demo, Samppart, and Sampeast. (Samppart and Sampeast are examples of partitioned applications. These applications are available only if your company has licensed Hyperion Essbase Partitioning.) The individual who installs the server is responsible for making the example applications available to end users:

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

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

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

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

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 the same font as the text.
`\ESSBASE`	Names of files, directories, and specific text you must type appear in `This Font`.
Hyperion Essbase System Login dialog box	Names of dialog boxes and their controls, such as buttons, text boxes, and list boxes, appear in the same font as the text.
Hyperion Essbase Installation Guide	Titles of books appear in *italics*. Italics also indicate important terms and special emphasis.
File > Open	The greater than symbol (>) indicates a menu followed by an individual menu command in that menu.

Additional Documentation Sources

In addition to the *Hyperion Essbase Database Administrator's Guide*, the Hyperion Essbase documentation set includes the following:

- The *Hyperion Essbase Start Here* booklet, which provides late-breaking information as well as the most important migration and feature information for this release of Hyperion Essbase.
- The *Hyperion Essbase Documentation Roadmap*, which lists all the Hyperion Essbase documentation and tells you how to access the online information.

- The *Hyperion Essbase Installation Guide*, which shows you how to install and configure the Hyperion Essbase server, Hyperion Essbase Spreadsheet Add-in, Hyperion Essbase Application Manager, Hyperion Essbase SQL Interface, Hyperion Essbase API, Runtime Client, and sample applications.
- The *Hyperion Essbase Quick Path Card (QPC)*, which provides an overview of the tasks involved in creating, using, and maintaining Hyperion Essbase databases.
- The Hyperion Essbase Application Manager online help file, which explains how to use the Application Manager user interface.
- The *Hyperion Essbase Quick Technical Reference*, which provides syntax for Hyperion Essbase functions, calculation commands, report commands, ESSCMD commands, and configuration file (`.CFG` file) settings.
- The online *Technical Reference* in the `DOCS` directory, which lists and describes Hyperion Essbase functions, calculation commands, report commands, ESSCMD commands, and configuration file (`.CFG` file) settings.
- The *Hyperion Essbase Spreadsheet Add-in User's Guide*, which explains how to use Hyperion Essbase Spreadsheet Add-in features with Microsoft Excel or 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 Reader (Version 3.0.1 or later). Adobe Acrobat Reader is provided on the Hyperion Essbase CD-ROM, and can also be downloaded from `WWW.ADOBE.COM`.
- The Hyperion Essbase Spreadsheet Add-in online help files, which explain how to use Hyperion Essbase Spreadsheet Add-in, and provide all the spreadsheet macros and VBA functions.
- The online *API Reference* in the `DOCS` directory, which lists and describes programmatic functions available through the Hyperion Essbase API, and provides information to help you start programming with these functions.
- The *Hyperion Essbase SQL Interface Guide*, which contains information on how to set up systems to load data by means of Hyperion Essbase SQL Interface. This guide is provided in the `\ESSBASE\DOCS\CLIENT` directory in `.PDF` format for online viewing and printing in Adobe Acrobat Reader (Version 3.0.1 or later). Adobe Acrobat Reader is provided on the Hyperion Essbase CD-ROM, and can also be downloaded from `WWW.ADOBE.COM`.

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, "Introduction to Hyperion Essbase," 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 Hyperion 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

Introduction to Hyperion Essbase

This chapter introduces Hyperion Essbase and describes the Hyperion Essbase environment. Hyperion Essbase is a multidimensional database server that is optimized for planning, analysis, and management reporting applications. You can access Hyperion 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 Hyperion Essbase.

This chapter contains the following sections:

About the Hyperion Essbase Product Family

The Hyperion Essbase product family includes the following feature sets:

- Hyperion Essbase Application Manager

  A graphical environment for developing and maintaining Hyperion Essbase applications. Tasks include building outlines and dimensions, performing data loads 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 Hyperion Essbase server manages analytical data models, data storage, calculations, and data security.

- Hyperion Essbase Spreadsheet Add-in

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

- Hyperion Essbase application tools

  A suite of tools for extending Hyperion Essbase applications. These tools include Hyperion Essbase Currency Conversion, Hyperion Essbase SQL Interface, Hyperion Essbase Spreadsheet Toolkit, and Hyperion Essbase API.

- Hyperion Essbase Partitioning

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

Multidimensional Development Features

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

- Hyperion 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 dimensions, change calculations, and modify hierarchies to reflect new business developments. The dynamic dimension builder automatically defines and dynamically loads large amounts of data. You can load spreadsheets, flat files, and SQL tables directly into the database.
- Hyperion Essbase contains over 100 analytical functions, all of which operate on very large data sets. You can perform key analytical functions, such as trend analysis, ratios, and allocations, without programming. You can define calculations quickly by using pre-defined functions for depreciation, variances, and other common formulas.
- Hyperion 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.
- Hyperion Essbase is easy to deploy and supports standard applications, operating systems, and networking protocols.

Architectural Features of Hyperion Essbase

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

Dynamic Dimensionality

The Hyperion Essbase server uses a method called dynamic dimensionality for storing and retrieving data and for optimizing analytical performance. This method separates data into sparse and dense dimensions. See Chapter 3, “Multidimensional Concepts” and Chapter 4, “Basic Architectural Elements” to learn how Hyperion Essbase defines and uses sparse and dense dimensions to optimize data access and to reduce index and storage requirements within the database.

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

Multithreaded Design and SMP

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

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

Multi-User Read and Write

The Hyperion 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 Hyperion Essbase client-server architecture supports enterprise analysis applications. The server runs Hyperion 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 Hyperion Essbase software.

Hyperion 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.

Hyperion 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.

Hyperion Essbase Server

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

The Hyperion Essbase server software runs on PC or UNIX servers. See the *Hyperion Essbase Start Here* booklet for information on the supported operating systems. See the *Hyperion Essbase Installation Guide* for specific information on server configuration requirements.

Hyperion Essbase Client

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

There are three types of Hyperion Essbase clients:

- The first client interface is Hyperion Essbase Spreadsheet Add-in, which provides users with seamless data access to the server. The *Hyperion Essbase Spreadsheet Add-in User's Guide* describes how to use the spreadsheet client interface.
- The second client interface is Hyperion Essbase Application Manager, which designs, develops, and maintains Hyperion Essbase applications. This manual provides task-oriented instructions for using Hyperion Essbase Application Manager.
- The third client interface is a custom application built with Hyperion Essbase Application Programming Interface (API). Hyperion Essbase API enables application developers to create custom interfaces to Hyperion Essbase quickly, using standard tools. You can use Hyperion Essbase 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 Hyperion Essbase API, although they are not specifically tested by software providers. The online *API Reference* in the `DOCS` directory provides a complete listing of functions, platforms, and supported compilers.

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

Chapter 2

Steps for Implementing Hyperion Essbase

You have 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 Hyperion Essbase to integrate your data as efficiently as possible.

The following table describes the steps you need to complete to get up and running with Hyperion Essbase and tells you where you can find more information about each step.

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

Table 2-1: A Process Map

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

Table 2-1: A Process Map (Continued)

Process step	For information, see...
Analyze your data from a multidimensional perspective. Consider: • Where are your data sources? • What type is the data? Is it detailed, relational data or is it higher-level, hierarchical data that can be used for analysis? • In what format is the data? • How will you access the data? If you need to access relational data, you may need Hyperion Essbase SQL Interface or Hyperion Essbase Integration Server (a separately purchasable product).	• Chapter 3, “Multidimensional Concepts” • *Hyperion Essbase SQL Interface Guide* • ODBC drivers documentation • Hyperion Essbase Integration Server documentation
Learn the fundamentals of Hyperion Essbase and distributed OLAP.	• Chapter 1, “Introduction to Hyperion Essbase” • Chapter 3, “Multidimensional Concepts” • Chapter 4, “Basic Architectural Elements” • Chapter 7, “Creating Applications and Databases” • Chapter 10, “Working with Attributes” • Chapter 6, “Designing Partitioned Applications” • Attend a Hyperion Essbase training class; contact your software provider for details.
Design your application and database. Think about which dimensions you will designate as sparse and which as dense, which dimensions you will designate as time and which as account, and where you will include attribute dimensions.	Chapter 5, “Designing a Single-Server Application”
Estimate the size of your database.	Chapter 15, “Estimating Disk and Memory Requirements for a Database”

Table 2-1: A Process Map (Continued)

Process step	For information, see...
Allocate storage and specify Hyperion Essbase Kernel settings for your database.	Chapter 41, "Specifying Hyperion Essbase Kernel Settings"
Learn about Hyperion Essbase Partitioning. Think about whether your data can benefit from being decentralized into connected databases.	• Chapter 6, "Designing Partitioned Applications" • Chapter 16, "Building and Maintaining Partitions"
Learn about dynamic calculations and how they can greatly improve performance.	Chapter 29, "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 will introduce new members into the outline. If so, set up dynamic dimension building. If not, set up regular data loads.	• Chapter 13, "Introducing Dynamic Dimension Building" • Chapter 14, "Building Dimensions Using a Rules File"
Load your data.	Part IV, "Loading Data"
Calculate your database.	Part V, "Calculating Data"
Run a report.	Part VI, "Reporting on Data"
Use Hyperion Essbase Spreadsheet Add-in to retrieve data.	The *Hyperion Essbase Spreadsheet Add-in User's Guide* for your spreadsheet application
Link files or cell notes to data cells.	Chapter 12, "Linking Objects to Hyperion Essbase Data"
Assign alias names to your members.	Chapter 11, "Creating and Managing Aliases"
Copy or export data subsets.	Chapter 39, "Copying Data Subsets and Exporting Data to Other Programs"

Table 2-1: A Process Map (Continued)

Process step	For information, see...
Back up and restore your data.	Chapter 48, “Backing Up Data and Recovering Databases”
Design a currency application.	Chapter 43, “Designing and Building Currency Conversion Applications”
Fine-tune your database performance and storage settings.	• Chapter 46, “Using Diagnostics to Monitor Performance” • Chapter 40, “Introducing the Hyperion Essbase Kernel” • Chapter 41, “Specifying Hyperion Essbase Kernel Settings”
Automate routine operations by using ESSCMD.	Chapter 44, “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 that are described as online analytical processing (OLAP) applications. This chapter discusses multidimensional concepts and terminology. This chapter contains the following sections:

Introducing OLAP

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

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

- How did Product A sell last month? How does this figure compare to sales in the same month over the last five years? How did the product sell by branch, region, and territory?
- Did this product sell better in particular regions? Are there regional trends?
- Did customers return Product A last year? Were the returns due to product defects? Did the company manufacture the products in a specific plant?
- Did commissions and pricing affect how salespeople sold the product? Did particular salespeople do a better job of selling the product?

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

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

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

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

Introducing Dimensions and Members

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

Dimensions. Hyperion Essbase has two types of dimensions: *standard dimensions* and *attribute dimensions.*

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

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

Attribute dimensions must be associated with a base standard dimension. Hyperion Essbase does not store the data for attribute dimensions, Hyperion Essbase dynamically calculates the data when a user retrieves it. For more information about attribute dimensions, see Chapter 10, "Working with Attributes."

Members are the individual components of a dimension. For example, Product A, Product B, and Product C might be members of the Product dimension. Each member has a unique name. A dimension can contain an unlimited number of members. Hyperion Essbase can store the data associated with a member (referred to as a stored member in this chapter) or it can dynamically calculate the data when a user retrieves it. For more information, see Chapter 29, "Dynamically Calculating Data Values."

A dimension represents the highest consolidation level in the database outline. The database outline indents members below one another to indicate a consolidation relationship. For example, in Figure 3-1, 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 drill and pivot data.

Arranging Dimensions into Hierarchies

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

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

Hyperion 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 Hyperion Essbase how to consolidate the values of its individual members. Within the tree structure of the database outline, a *consolidation* is a group of members in a branch of the tree.

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

In the Sample Basic database, 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 consist of the members that represent the 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.

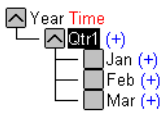


Figure 3-1: Hierarchical Structure

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

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

Defining Hyperion Essbase Terminology

Hyperion 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

Hyperion Essbase uses hierarchical and family history terms to describe the roles and relationships of the members in an outline. You can describe the position of the members of the branches in Figure 3-2 in several ways.

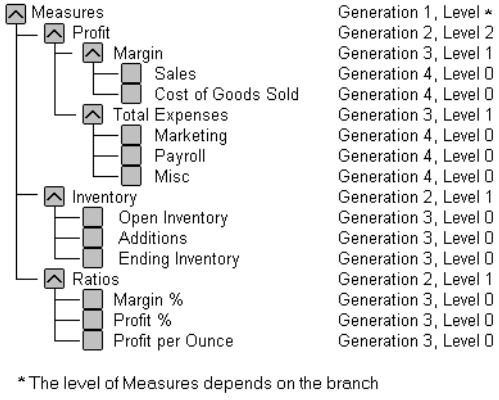


Figure 3-2: Member Generation and Level Numbers

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

Descendants and Ancestors

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

Descendants: Descendants are all the members in branches below a parent. For example, Profit, Inventory, and Ratios are descendants of Measures. The children of Profit, Inventory, and Ratios are also descendants of Measures.

Ancestors: Ancestors are all the members in branches above a member. For example, Margin, Profit, and Measures are ancestors of Sales.

Roots and Leaves

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

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

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 members.

Generations and Levels

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

Generations: Generation numbers refer to consolidation levels within a dimension. A root branch of the tree is generation 1. Generation numbers increase as you count from the root toward the leaf member. In Figure 3-2, 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.

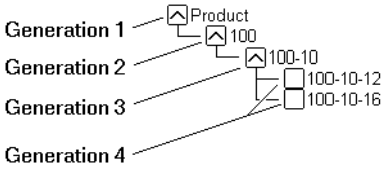


Figure 3-3: Generations

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

Figure 3-4 shows part of the Product dimension with its levels numbered.

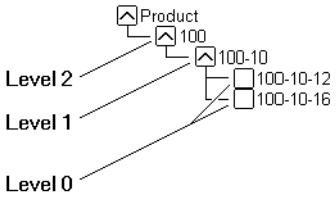


Figure 3-4: Levels

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

Identifying Values in a Multidimensional Database

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

Consider the simplified database in Figure 3-5:

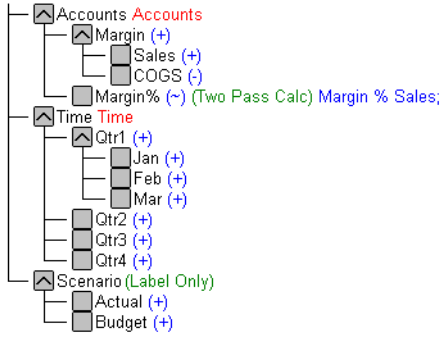


Figure 3-5: A Multidimensional Database Outline

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

- The Accounts dimension has four members: Sales, COGS, Margin, Margin%.
- The Time dimension has four quarter members. Figure 3-6 shows only the members in Qtr1.
- The Scenario dimension has two child members: Budget for budget values and Actual for actual values.

An intersection of members (one member from each dimension) represents a data value. The example in Figure 3-6 has three dimensions; thus, in Figure 3-6 the dimensions and data values in the database can be represented in a cube, as shown in Figure 3-6:

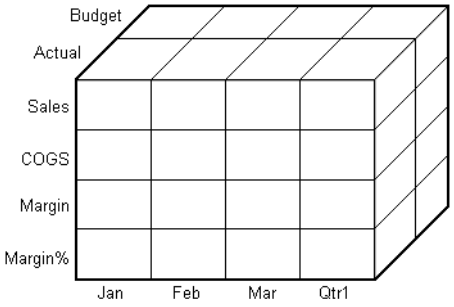


Figure 3-6: Three Dimensional Database

The shaded cells in Figure 3-7 illustrate that, when you refer to Sales, you are referring to a slice of the database containing eight Sales values:

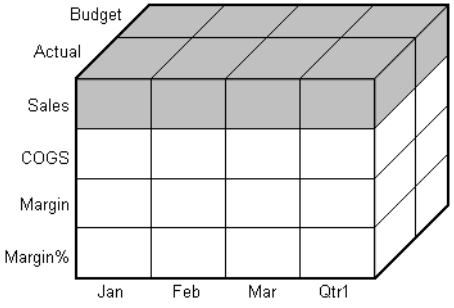


Figure 3-7: Sales Slice of the Database

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

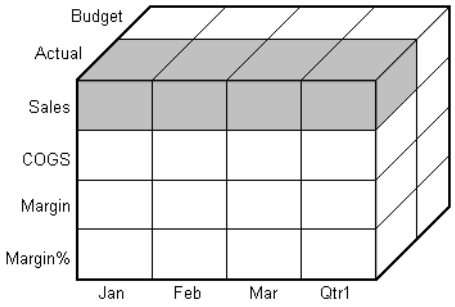


Figure 3-8: Actual, Sales Slice of the Database

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 3-9, the cell containing the data value for Sales, Jan, Actual is shaded. The data value can also be expressed using the cross-dimensional operator (->) as Sales->Actual->Jan.

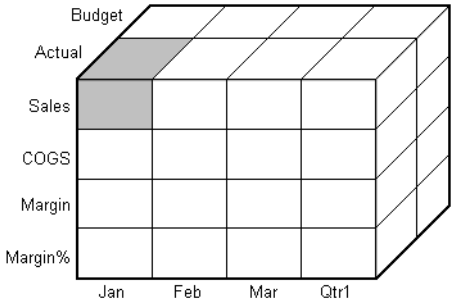


Figure 3-9: Sales, Jan, Actual Slice of the Database

Looking at Data from Different Perspectives

Slicing the database in different ways gives you different perspectives of the data. The slice in Figure 3-10, for example, shows data about the month of January:

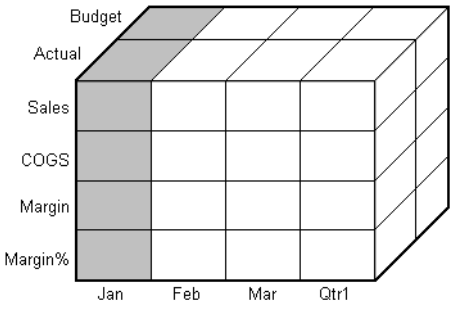


Figure 3-10: Data for January

The slice in Figure 3-11 shows data for the month of February:

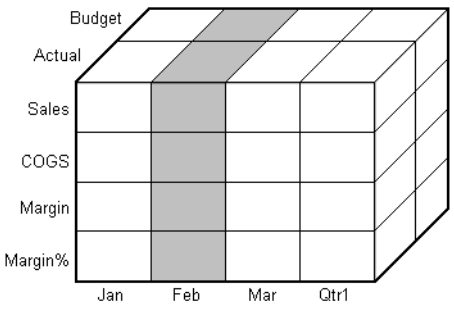


Figure 3-11: Data for February

The slice in Figure 3-12 shows data for profit margin:

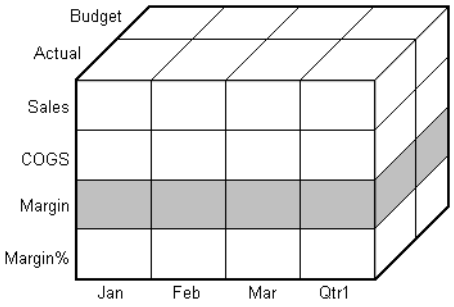


Figure 3-12: Data for Profit Margin

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 in Figure 3-10, 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 that you use when building a typical multidimensional database.

First, analyze a 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. Because all users have access to the server, they can retrieve data at will, regardless of the data's origin. To accommodate their needs, TCC chooses Hyperion Essbase.

Organizing Multidimensional Data

First, you will create a database outline for TCC. The outline defines the structure of the database, including the dimensions and members that it contains. It is important to remember that Hyperion Essbase stores the database outline separately from the data in the database. Each time you make a significant change to the database outline, Hyperion Essbase restructures the data 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 standard dimensions, attribute dimensions, and members. The members can be stored or they can be dynamically calculated upon retrieval. The following example uses only standard dimensions and stored members. For more information about attribute dimensions and dynamically calculated members, see “Introducing Dimensions and Members” on page 3-3. Your first job is to determine a logical structure for the data. Remember, standard dimensions often parallel a company’s organization. TCC has four standard dimensions: Time, Product, Market, and Measures.

Figure 3-13: TCC Simple with Four Standard Dimensions

Give each dimension two members:

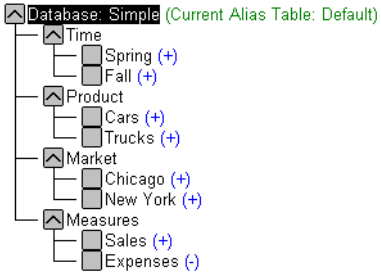


Figure 3-14: TCC Simple with Standard Dimensions and Members

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

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

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

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

Time	Product	Market	Measures	Data Value
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
Spring	Product	New York	Sales	1050
Spring	Product	New York	Expenses	350
Spring	Product	New York	Measures	700
Spring	Product	Market	Sales	2550
Spring	Product	Market	Expenses	1350
Spring	**Product**	**Market**	**Measures**	**1200**

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

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

A typical database contains associated formulas and a calc script to analyze the data. For example, you might want to calculate the variance between budget and actual expenses values. You calculate the variance by defining the appropriate formula on a Variance member. For more information on developing formulas and calc scripts, see Chapter 25, “Introduction to Database Calculations.”

Adding and Deleting Standard Dimensions and Stored Members

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

Adding Stored Members

In this example, you will add several stored members to the database under each standard dimension. Add two seasons under Time; one product called Motorcycles; a market, LA; and three 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.”

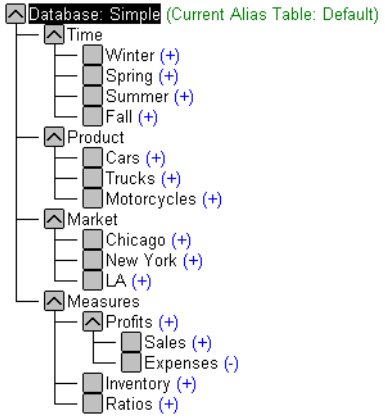


Figure 3-15: Adding a Stored Member

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

Adding a Standard Dimension

Now you will add a new standard dimension called Distribution Channel to the database. In addition, you will give the new dimension two stored members, Retail and Wholesale. For more information, see “Adding Dimensions and Members to Outlines” on page 8-15.

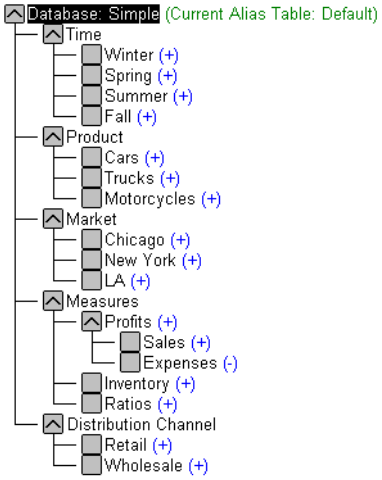


Figure 3-16: Adding a Standard Dimension

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

When you add a new standard dimension to an outline, you must associate any data in the database with one of the members of the new dimension. For example, in the Simple database, you would have to specify whether the existing data represented Retail or Wholesale. You would then need to load data and calculate the database. For more information, see Chapter 20, “Introducing Data Loading” and Chapter 25, “Introduction to Database Calculations.”

Removing a Standard Dimension

In this example, you will 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 standard dimension diminishes the overall size of the database. However, data for all members in the Market dimension still exists. If you delete a standard dimension from a database outline, the data associated with one member of the deleted standard dimension is retained. You must choose which member’s data to retain.

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

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

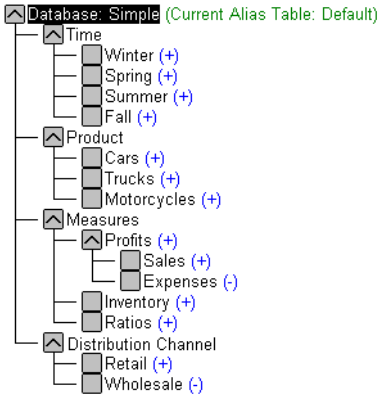


Figure 3-17: Deleting a Standard Dimension and Members

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:

Attribute Dimensions and Standard Dimensions

Hyperion Essbase has two types of dimensions: attribute dimensions and non-attribute dimensions (standard dimensions). An attribute dimension is a special type of dimension that is associated with a standard dimension. For more information about attribute dimensions, see Chapter 10, "Working with Attributes." This chapter primarily considers standard dimensions because Hyperion Essbase does not allocate storage for attribute dimension members. Instead it dynamically calculates the members when the user requests data associated with them.

Sparse and Dense Dimensions

Most data sets of multidimensional applications have two characteristics:

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

Hyperion Essbase maximizes performance by dividing an application's standard dimensions into two types: dense dimensions and sparse dimensions. This division allows Hyperion Essbase to cope with data that is not smoothly distributed, without losing the advantages of matrix-style access to the data. Hyperion 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, as shown in Figure 4-1, 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. For example, in the Sample Basic database, accounts data exists for almost all products in all markets, so Measures

is chosen as a dense dimension. Year and Scenario are also chosen as dense dimensions. Year represents time in months, and Scenario represents whether the accounts values are budget or actual values.

Note: Caffeinated, Intro Date, Ounces, and Pkg Type are attribute dimensions that are associated with the Product dimension. Population is an attribute dimension that is associated with the Market dimension. Members of attribute dimensions describe characteristics of the members of the dimensions with which they are associated. For example, each product has a size in ounces. Attribute dimensions are always sparse dimensions and must be associated with a sparse standard dimension. Hyperion Essbase does not store the data for attribute dimensions, Hyperion Essbase dynamically calculates the data when a user retrieves it. For more information about attribute dimensions, see Chapter 10, “Working with Attributes.”

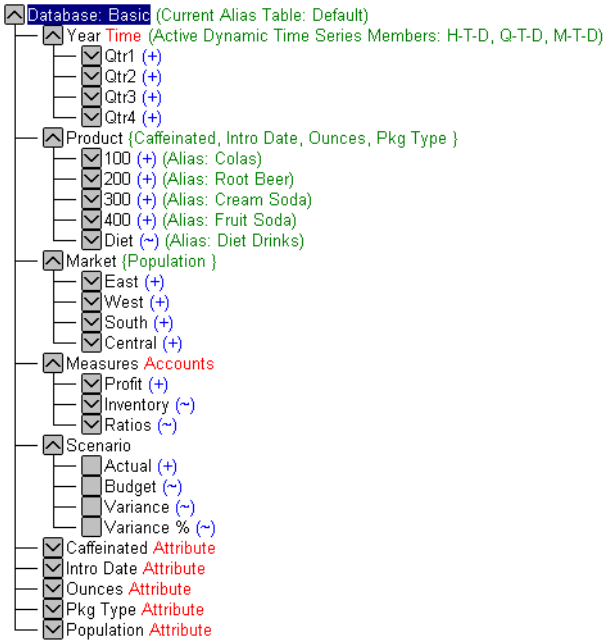


Figure 4-1: Sample Basic Database Outline

Data Blocks and the Index System

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

Hyperion Essbase creates a data block for each unique combination of sparse standard 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 its combination of sparse dimension members.

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

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

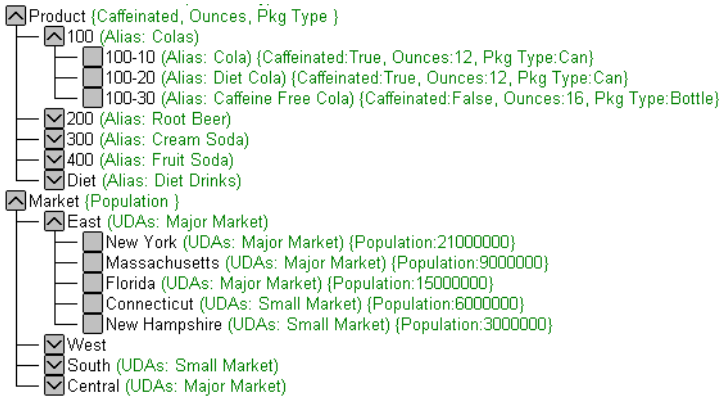


Figure 4-2: Product and Market Dimensions from the Sample Basic Database

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

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

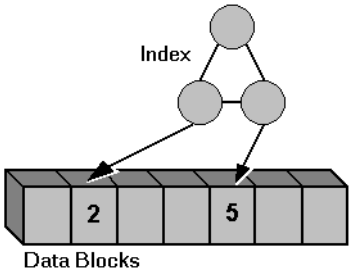


Figure 4-3: Simplified Index and Data Blocks

Each unique data value can be considered to exist in a cell in a data block. When Hyperion Essbase searches for a data value, it uses the index to locate the appropriate 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 includes only 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).

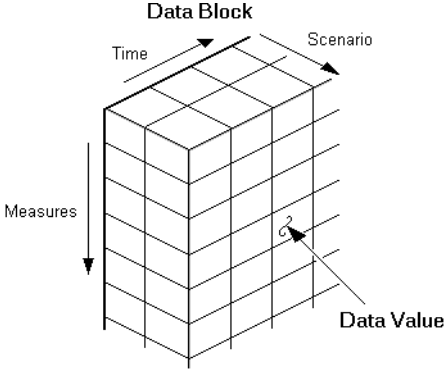


Figure 4-4: Part of a Data Block for the Sample Basic Database

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.

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

Figure 4-5 references the following database outline:

```
A (Dense)
   a1
   a2
B (Dense)
   b1
       b11
       b12
   b2
       b21
       b22
C (Dense)
   c1
   c2
   c3
D (Sparse)
   d1
   d2
       d21
       d22
E (Sparse)
   e1
   e2
   e3
```

The block in Figure 4-5 represents the three dense dimensions within the combination of the sparse members d22 and e3. In Hyperion Essbase, member combinations are denoted by the cross-dimensional operator. The symbol for the cross-dimensional operator is ->. So d22, e3 is written d22->e3. A, b21, c3 is written A->b21->c3.

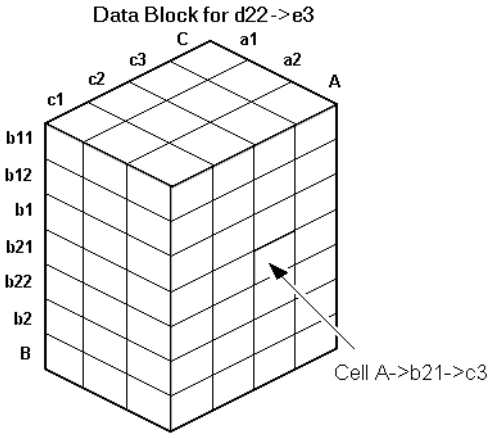


Figure 4-5: Data Block Representing Dense Dimensions for d22->e3

Hyperion 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. Hyperion 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 41-27.

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

Selection of Sparse and Dense Dimensions

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

When you create a database outline in Outline Editor, Hyperion Essbase automatically suggests which of the dimensions should be sparse and which dense. Hyperion Essbase, for example, consider 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 Properties.” 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 only an estimate. It cannot take into account the nature of the data you will load into your database or multi-user considerations.

You can associate attribute dimensions with *sparse* standard dimensions only. Attribute dimensions themselves are always sparse dimensions.

You can turn off auto-configuration and manually choose your sparse and dense dimensions. To help you select sparse and dense dimensions, Hyperion Essbase provides data storage information in the Application Manager Data Storage dialog box. To open this dialog box, open the database outline and choose Settings > Data Storage. The information that is provided includes data block size and density.

This information helps you choose the optimal configuration for your database. For more information, see Chapter 41, “Specifying Hyperion Essbase Kernel Settings.”

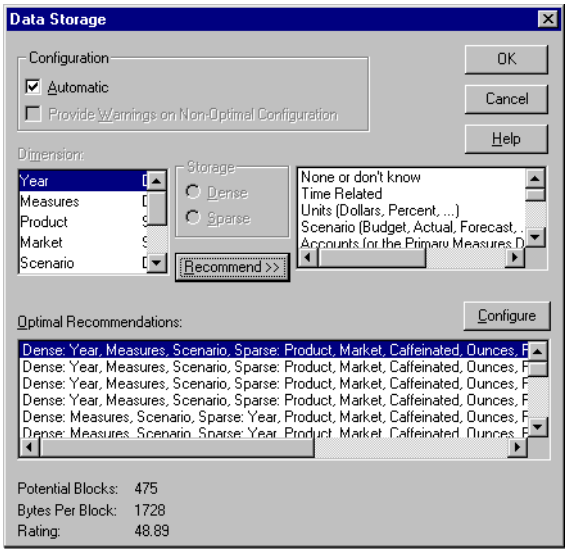


Figure 4-6: Data Storage Dialog Box

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

TBC does not sell every product in every market; therefore, the data set is reasonably sparse. Data values do not exist for many combinations of members on the Product and Market dimensions. For example, if Caffeine Free Cola is not sold in Florida, then data values do not exist for the combination Caffeine Free Cola (100-30)->Florida. So, Product and Market are sparse dimensions. Therefore, if no data values exist for a specific combination of members on these dimensions, Hyperion 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 some member combinations on these dimensions. For example, data values exist for the member combination Sales->January->Actual because at least some products are sold in January. Thus, Year and, similarly, Measures and Scenario are dense dimensions.

In configuration of the Sample Basic database may be summarized as follows:

- Sparse standard dimensions: Product and Market
- Dense standard dimensions: Year, Measures, and Scenario

Hyperion 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 probably exists for February and for 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.

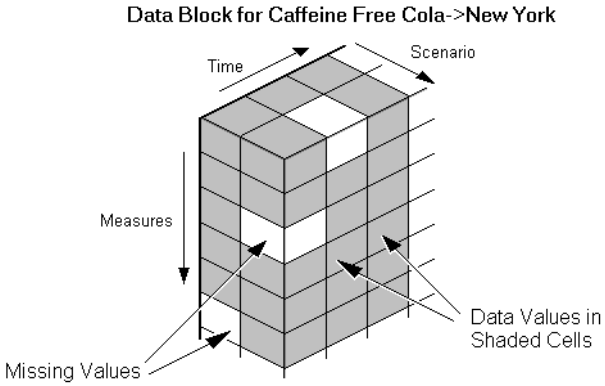


Figure 4-7: Dense Data Block for Sample Basic Database

Dense and Sparse Selection Scenarios

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

Scenario 1: A Database Consisting Entirely of Sparse Standard Dimensions

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

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

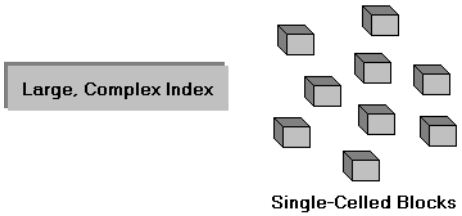


Figure 4-8: Database with All Sparse Standard Dimensions

Scenario 2: A Database Consisting Entirely of Dense Standard Dimensions

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

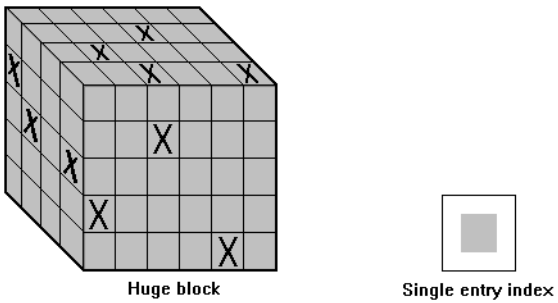


Figure 4-9: Database with All Dense Standard Dimensions

Scenario 3: An Ideal Configuration of Dense and Sparse Standard Dimensions

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

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

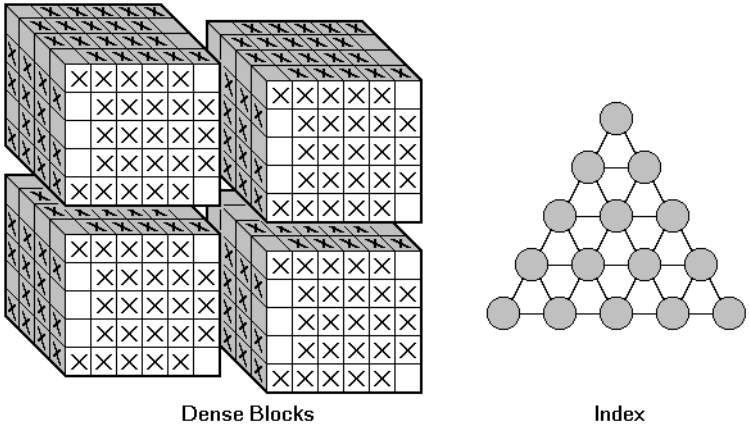


Figure 4-10: An Ideal Configuration

Scenario 4: A Typical Multidimensional Problem

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

The two-dimensional data blocks 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.

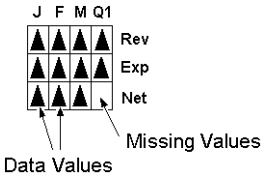


Figure 4-11: Two-dimensional Data Block for Time and Accounts

Hyperion Essbase creates data blocks for combinations of members on the sparse standard dimensions (providing at least one data value exists for the member combination). The sparse dimensions are Region and Product. The members of the Region dimension are East, West, South, and Total US. The members 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, Hyperion Essbase has not created data blocks for these member combinations. The data blocks that Hyperion Essbase has created have very few empty cells.

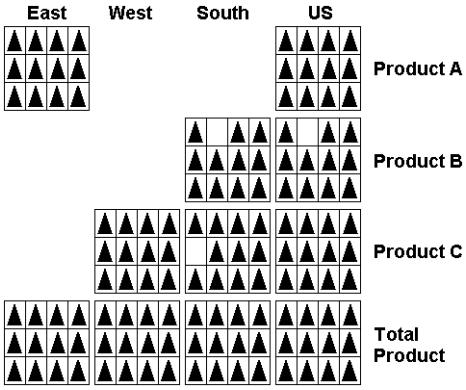


Figure 4-12: Data Blocks Created for Sparse Members on Region and Product

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

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

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

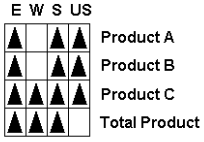


Figure 4-13: Two-Dimensional Data Block for Region and Product

Hyperion Essbase creates data blocks for combinations of members on the sparse standard dimensions (providing at least one data value exists for the member combination). The sparse standard 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, Hyperion 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.

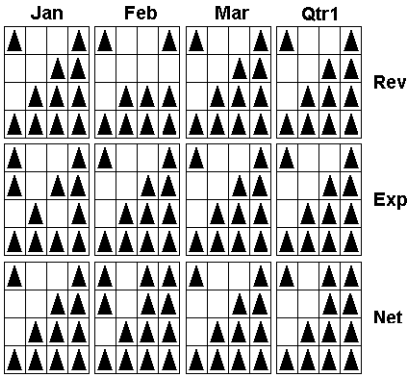


Figure 4-14: Data Blocks Created for Sparse Members on Time and Accounts

The Hyperion Essbase Solution

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

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

For more information on:

- Planning the development of your multidimensional database, see Chapter 5, “Designing a Single-Server Application.”
- Selecting dense and sparse dimensions, see “Sparse and Dense Dimensions” on page 4-2.
- Loading data, see Chapter 20, “Introducing Data Loading.”
- Compressing data and optimizing your database, see “Specifying Data Compression” on page 41-27
- Calculating your database, see Chapter 25, “Introduction to Database Calculations.”

Chapter 5

Designing a Single-Server Application

To implement a multidimensional database, first you install Hyperion Essbase OLAP Server, and then you design and create an application. You analyze data sources and define requirements very carefully and then decide whether a single-server application or a partitioned, distributed approach best serves your needs. For criteria that you can review to decide whether to partition an application, see Chapter 6, “Designing Partitioned Applications.”

Using a case study, this chapter provides an overview of the database planning process and discusses working rules that you can follow to design a single-server, multidimensional database solution for your organization. For detailed information about building applications and databases, see Chapter 7, “Creating Applications and Databases.”

This chapter includes the following sections:

Designing an Application

As illustrated in Figure 5-1, designing an application is a cyclic process that moves from a planning stage to verification stage.

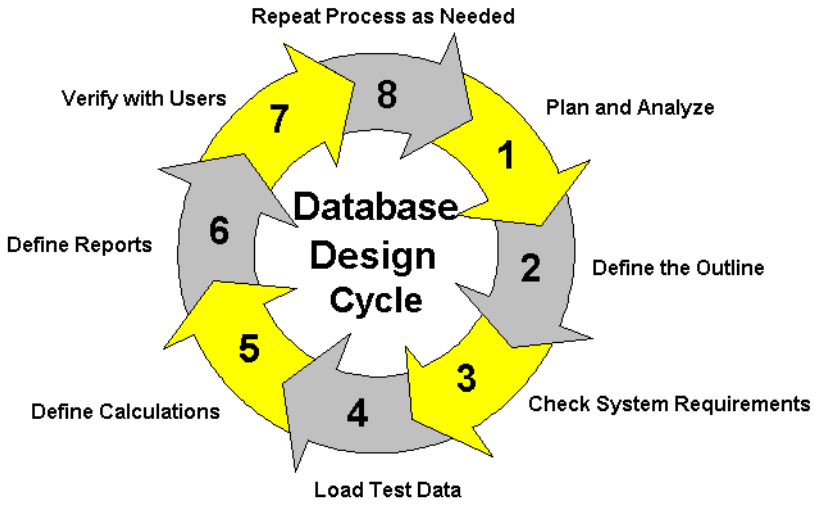


Figure 5-1: The Database Design Cycle

The database design process includes the following basic steps:

1. **Analyze business needs and plan the database.** In this step you determine the information needs that the application and database must satisfy. You identify source data, user information, and access needs.
2. **Define the database outline.** The *outline* determines the structure of the database—what information is stored and how different pieces of information relate to one another.
3. **Check system requirements.** How you meet system requirements and define system parameters affects the efficiency and performance of the database. Make sure that you have allocated enough disk space for the database and that the index and data file caches in memory are of adequate size (see Chapter 15, “Estimating Disk and Memory Requirements for a Database” and Chapter 41, “Specifying Hyperion Essbase Kernel Settings”).

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

Case Study: The Beverage Company

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

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

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

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

Planning and Analyzing

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

The planning and analysis phase involves three tasks:

- Identifying the data that you want to include in the database
- Identifying users' requirements
- Creating and testing a business model for the data

When designing a multidimensional application, consider these factors:

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

Before you create a database and build its outline, you must create a Hyperion Essbase application to contain it. Applications that use the optional Currency Conversion module generally consist of a main database and a separate currency database (see Chapter 43, "Designing and Building Currency Conversion Applications").

Identifying Source Data

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

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

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

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

Consider who updates the database and how frequently. Do the individuals who need to update data have access to the data?

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

Checklist: Select Data for the Hyperion Essbase Database

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

Identifying User Requirements

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

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

Creating a Business Model

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

Most businesses choose to analyze include the following areas:

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

Identifying Analysis Objectives

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

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

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

Determining Dimensions and Members

You can represent each of the business views as a separate dimension in the database. If you need to analyze a business area by classification or attribute, such as by the size or color of products, you can use attribute dimensions to represent the classification views. For information about attribute dimensions, see Chapter 10, “Working with Attributes.”

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

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

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

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

In other words, the analyst may want to examine information from three different perspectives: time, account, and scenario. The sample database shown in Figure 5-2 represents these three perspectives as three dimensions, with one dimension represented along each of the three axes:

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

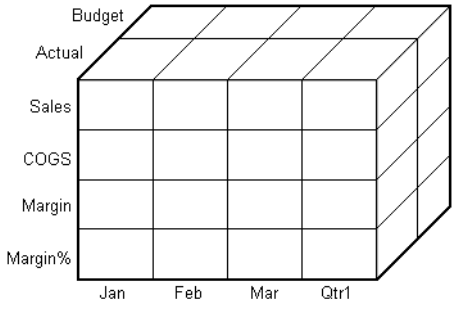


Figure 5-2: Cube Representing Three Database Dimensions

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

Table 5-1 shows a summary of TBC's business areas that the planner determined would be dimensions. These are the major business areas to be analyzed. The planner created three columns, with the dimensions in the left column and members in the two right columns. The members in column 3 are subcategories of the members in column 2. In some cases, members in column 3 are broken into another level of subcategories; for example, the Margin and Total Expenses members of the Measures dimension.

Table 5-1: TBC Sample Dimensions

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

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

Table 5-2: TBC Sample Attribute Dimensions

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

Checklist: Create a Business Model

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

Analyzing Database Design

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

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

Examine dimension relationships

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

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

```
             Cust A  Cust B  Cust C

New York     100     N/A     N/A
Illinois     N/A     150     N/A
California   N/A     N/A     30
```

Cust A is only in New York, Cust B is only in Illinois, and Cust C is only in California. In this situation, the company can define the data in one standard dimension:

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

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

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

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

```
             Cust A  Cust B  Cust C

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

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

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

Examine dimension combinations

Break each combination of two dimensions into a two-dimensional matrix. For example, proposed dimensions at TBC (as listed in Table 5-1) include the following combinations:

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

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

To help visualize each dimension, you can draw a matrix and include a few of the first generation members. Figure 5-3 shows a simplified set of matrixes for three dimensions.

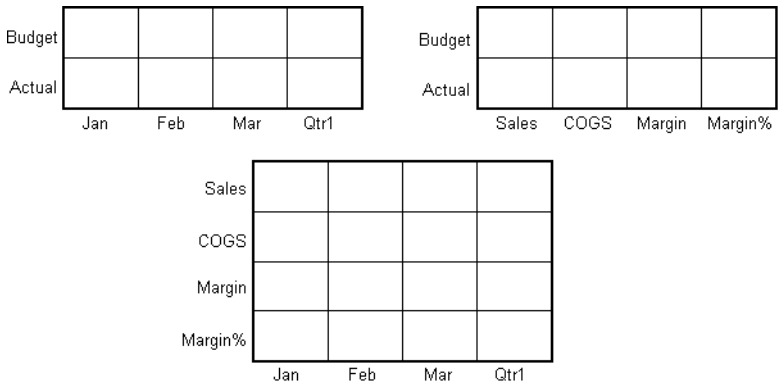


Figure 5-3: Analyzing Dimensional Relationships

For each combination of dimensions, ask three questions:

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

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

Avoid repetition in the outline

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

Repetition

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

No Repetition

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

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

As a way to analyze members by their characteristics or attributes, one approach to outline design repeats elements as shared members. The first example uses shared members to analyze diet beverages. You can avoid the repetition of the first example and simplify the design of the outline by creating a Diet attribute dimension, as shown in the second example.

Repetition

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

No Repetition

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

Diet Attribute (Type: Boolean)
- True
- False

Attribute dimensions also provide additional analytic capabilities. For guidelines on when to use attribute dimensions, see “Attribute Design Considerations” on page 10-12.

Avoid interdimensional irrelevance

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

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

TBC considered separating employees into a separate dimension. Table 5-3 shows an example of how TBC analyzed the proposed Employee dimension for interdimensional irrelevance. Members of the proposed Employee dimension are compared with members of the Measures dimension. Only the Salary measure is relevant to individual employees.

Table 5-3: Interdimensional Irrelevance Example

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

Split databases if necessary

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

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

Checklist: Analyze the Database

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

Planning for Security in a Multi-User Environment

The time to think about the type of security privileges you plan to issue for a Hyperion Essbase database is when you consider user information needs. As a Hyperion Essbase administrator, you frequently load external data into the database. Determine who else should have privileges for these operations. End the plan with a list of users and privileges. See Chapter 17, “Managing Security at Global and User Levels” to learn more about assigning user privileges.

Checklist: Plan for Security

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

Drafting an Outline

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

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

- **Year**—TBC needs to collect data monthly and summarize the monthly data by quarter and year. Monthly data, stored in members such as Jan, Feb, and Mar, consolidates to quarters. Quarterly data, stored in members such as Qtr1 and Qtr2, consolidates into Year.
- **Measures**—Sales, Cost of Goods Sold, Marketing, Payroll, Miscellaneous, Opening Inventory, Additions, and Ending Inventory are standard measures. Hyperion Essbase can calculate Margin, Total Expenses, Profit, Total Inventory, Profit %, Margin %, and Profit per Ounce from these measures. TBC needs to calculate Measures on a monthly, quarterly, and yearly basis.
- **Product**—The Product codes are 100-10, 100-20, 100-30, 200-10, 200-20, 200-30, 200-40, 300-10, 300-20, 300-30, 400-10, 400-20, and 400-30. Each product consolidates to its respective family: 100, 200, 300, and 400.

  TBC wants to enable analysis by size and package by associating each product with members of the Ounces and Pkg Type attribute dimensions.
- **Market**—Several states make up a region, and four regions make up a market. The states are New York, Massachusetts, Connecticut, Florida, New Hampshire, Oregon, Washington, California, Utah, Nevada, Texas, Louisiana, New Mexico, Oklahoma, Illinois, Ohio, Wisconsin, Missouri, Iowa, and Colorado. Each state consolidates into its respective region: East, West, South, and Central. Each region consolidates into Market.
- **Scenario**—TBC derives and tracks budget data versus actual data. Managers must monitor and track budgets and actuals, as well as the variance and variance percentage between them.
- **Pkg Type**—TBC wants to see the effect that product packaging has on sales and profit. Establishing the Pkg Type attribute dimension enables users to analyze product information based on whether a product is packaged in bottles or cans.
- **Ounces**—TBC sells products in different sizes in ounces in different market areas. Establishing the Ounces attribute dimension helps users to monitor which sizes sell better in which markets.

Building an Outline

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

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

The following section introduces many of these terms and explains how to use them.

Defining Dimension and Member Properties

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

As shown in Figure 5-4, the Year dimension is tagged as time, the Measures dimension is tagged as accounts and label only, and the Scenario dimension is tagged as label only. Ounces and Pkg Type are attribute dimensions. The Products dimension is associated with two attribute dimensions, Ounces and Pkg Type.

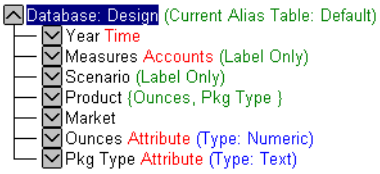


Figure 5-4: TBC Dimensions and Related Properties

A dimension type is a property that Hyperion Essbase provides that adds special functionality to a dimension. The most commonly used dimension types are time, accounts, and attribute. Table 5-4 defines each of the Hyperion Essbase dimension types.

Table 5-4: Dimension Types

Dimension Types	Description
None	Specifies no particular dimension type.
Time	Defines the time period for which you report and update data. You can tag only one dimension as time. The time dimension enables several accounts dimension functions, such as first and last time balances.
Accounts	Contains items that you want to measure, such as profit and inventory and makes Hyperion Essbase built-in accounting functionality available. (For example, see "Accounts Dimension Calculation" on page 5-29.) Only one dimension can be defined as accounts.
Attribute	Contains members that can be used to classify members of another, associated dimension. For example, the Pkg Type attribute dimension contains a member for each type of packaging, such as bottle or can, that applies to members of the Product dimension.
Country	Contains data about where business activities take place. In a country dimension, you can specify the type of currency used in each member. For example, Canada has three markets: Vancouver, Toronto, and Montreal. They use the same currency type, Canadian dollars.
Currency partition	Separates local currency members from a base currency defined in the application. This dimension type is used only in the main database and is used for currency conversion applications. If the base currency for analysis is US dollars, the local currency members contain values that are based on the currency type of the region.

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

As appropriate, you can change a member consolidation property to one of the following operators: +, -, *, /, %, and ~ (no consolidation). For more information on consolidation properties, see “Consolidation Paths” on page 5-26.

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

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

Table 5-5 describes Hyperion Essbase data storage properties.

Table 5-5: Hyperion Essbase Data Storage Properties

Storage Properties	Effects on Members
Store data	The member stores data. Store data is the default storage property.
Dynamic Calc	The data associated with the member is not calculated until requested by a user. The calculated data is not stored; it is discarded after the request is completed.
Dynamic Calc and Store	The data associated with the member is not calculated until it is requested by a user. The calculated data is then stored.
Shared member	The data associated with the member comes from another member with the same name.

Table 5-5: Hyperion Essbase Data Storage Properties (Continued)

Never share	The data associated with the member is duplicated with its parent or child if an implied shared relationship exists.
Label only	Although a label only member has no data associated with it, it can still display a value. The label only tag groups members and eases navigation and reporting. Typically, label only members have a no consolidation property. (See “Consolidation Ordering” on page 5-27.)

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

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

- Chapter 7, “Creating Applications and Databases”
- Chapter 8, “Creating and Changing Database Outlines”
- Chapter 9, “Setting Dimension and Member Properties”
- Chapter 10, “Working with Attributes”

Checklist: Define Dimension and Member Properties

- Can you identify a time dimension?
- Can you identify an accounts dimension?
- Does the data include foreign currencies? If so, did you identify a currency partition dimension?
- Can you identify qualities or characteristics of dimensions that should be defined as separate attribute dimensions?
- What members require special data storage properties?

Designing an Outline to Optimize Performance

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

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

Optimizing Query Performance

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

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

The outline shown in Figure 5-5 is designed for optimum query performance:

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

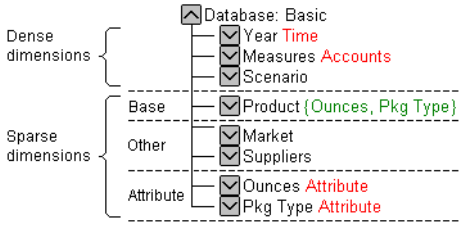


Figure 5-5: Designing an Outline for Optimized Query Times

Optimizing Calculation Performance

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

For more information about factors that affect calculation performance, see “Designing for Calculation Performance” on page 33-2.

The outline shown in Figure 5-6 is designed for optimum calculation performance:

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

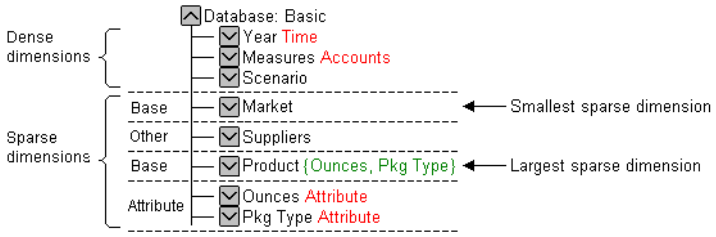


Figure 5-6: Designing an Outline for Optimized Calculation Times

Meeting the Needs of Both Calculation and Retrieval

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

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

Loading Test Data

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

Detailed instructions for loading data are in the following chapters:

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

When you are satisfied with a database design, you should test the loading of the complete set of real data with which you will populate the final database. To load the actual data, you may be able to use the data load rules files that you created for the test data. This final test may reveal problems with the source data that you did not anticipate during earlier phases of the database design process.

Defining and Testing Calculations

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

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

- Consolidation paths
- Consolidation order
- Calculations of time and accounts dimensions
- Formulas

Consolidation Paths

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

TBC's application has several consolidation paths:

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

Unary operators define how Hyperion Essbase rolls up data for each member in a branch to the parent. Hyperion Essbase gives members a default operator of addition (+), meaning that it adds the value of a member to the values of other members in the branch. For example, it adds 100-10, 100-20, and 100-30 and stores the result in their parent, 100, as shown in Figure 5-7.

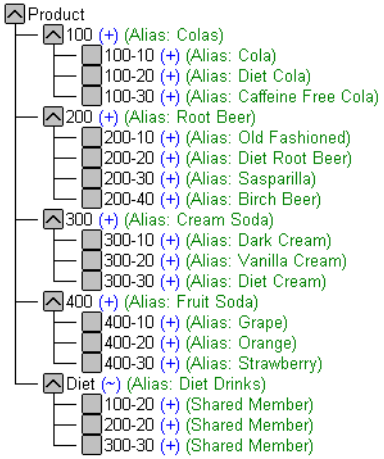


Figure 5-7: TBC Product Dimension

The Product dimension contains mostly (+), operators, which indicate that each group of members is added together and rolled up to the parent. Diet has a tilde (~), which indicates that Hyperion Essbase does not include the Diet member in the consolidation to the parent, Product. The Diet member consists entirely of

members it shares or duplicates. The TBC product management group wants to be able to isolate Diet drinks in reports, so TBC created a separate Diet member that does not impact the overall consolidation.

For more information on consolidation and calculations, see Chapter 9, “Setting Dimension and Member Properties.”

Consolidation Ordering

Hyperion Essbase calculates the data in a branch in top-down order. For example, if you have, in order, two members tagged with an addition symbol (+) and a third member tagged with a multiplication symbol (*). Hyperion Essbase adds the first two and multiplies the sum by the third. Be aware that Hyperion Essbase always begins with the top member when it consolidates, so the order and the labels of the members is very important. Table 5-6 shows the Hyperion Essbase unary operators. For more information, see “Introducing Member Consolidation Properties” on page 9-16.

Table 5-6: Unary Operations

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

Shared Members and Consolidation

Shared members also affect consolidation paths. The shared member concept lets two members with the same name share the same data. The shared member stores a pointer to data contained in the other member, so Hyperion 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 the outline?
- Did you tag each member with the proper unary operator?
- Did you specify a shared member tag for designated members?
- Would shared members be more efficient if designed as an attribute dimension?

Time and Accounts Calculations

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

Take a moment to look closely at the Measures dimension tags defined by TBC (in Figure 5-8). You see that you already know about many of the properties in the Measures dimension. You have 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. Some members of Measures have a Dynamic Calc tag. Dynamic calculations are discussed in “Dynamic Calculations” on page 5-33.

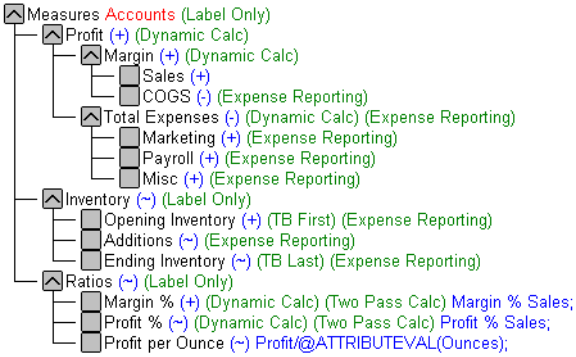


Figure 5-8: TBC Measures Dimension

For details on Hyperion Essbase calculations, see Chapter 25, “Introduction to Database Calculations” through Chapter 34, “Using Intelligent Calculation to Optimize Calculation.”

Accounts Dimension Calculation

This section discusses two forms of calculations for a dimension tagged as accounts: time balance and variance reporting.

Time Balance Properties

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

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

Ending Inventory data represents the inventory that TBC carries at the end of each month. The quarterly value for Ending Inventory is equal to the ending value for the quarter. Ending Inventory requires the time balance tag, TB last. Table 5-7 shows the time balance tags for the accounts dimension.

Table 5-7: Accounts Member Tags

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

Table 5-8 shows how consolidation in the time dimension is affected by time balance properties in the accounts dimension.

Table 5-8: TBC Consolidations Affected by Time Balance Properties

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

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

For more information on time balance tags, see “Introducing Time Balance Properties” on page 9-4.

Variance Reporting

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

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

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

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

Formulas

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

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

The operators include the following:

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

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

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

The Measures dimension uses the following formulas:

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

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

For a complete list of operators, functions, and syntax, see the online *Technical Reference* in the `DOCS` directory. For a discussion of how to use formulas, see Chapter 26, "Developing Formulas."

Dynamic Calculations

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

As shown in Figure 5-9, TBC's Measures dimension contains several members that are tagged as Dynamic Calc: Profit, Margin, Total Expenses, Margin %, and Profit %.

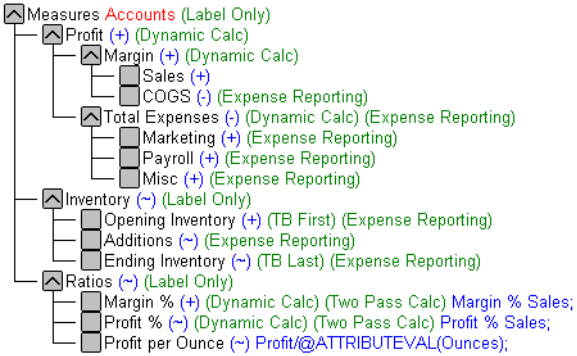


Figure 5-9: TBC Measures Dimension, Dynamic Calc Tags

When an overall database calculation is performed, these members and their corresponding formulas are not calculated. Rather, these members are calculated when a user requests them, for example, from Hyperion Essbase Spreadsheet Add-in. Hyperion Essbase does not store the calculated values; it recalculates the values for any subsequent retrieval. However, you can choose to store dynamically calculated values after the first retrieval. For more information, see "Understanding Dynamic Calc Members" on page 29-2.

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

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

For more information about dynamic calculations, see Chapter 29, “Dynamically Calculating Data Values.”

Two-Pass Calculations

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

Hyperion Essbase loads data into the system as follows:

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

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

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

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

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

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

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

Checklist: Define the Calculations

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

Defining Reports

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

Hyperion Essbase provides several tools that can help you during the design process to display and format the data quickly and test whether the database design meets user needs. You can use the Report Writer of Hyperion Essbase Application Manager to write report scripts quickly. Those familiar with spreadsheets can use the spreadsheet add-ins for Microsoft Excel and Lotus 1-2-3. See Chapter 35, “Quick Start to Report Scripts” and the appropriate spreadsheet user’s guide.

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

- Grouping and sequencing of data. Do the intersections enabled by the design provide the data that users need?
- Levels of totals. In spreadsheets this term refers to the consolidation levels that are available to a user who drills down and up through the hierarchy of the outline design.
- Attribute reporting. Does the database design facilitate an analysis that is based on the characteristics or attributes of specific dimensions or members? For example, do you need to compare sales by specific combinations of size and packaging, such as comparing the sales of 16-ounce bottled colas with the sales of 32-ounce bottled colas? How can you take advantage of the Attribute Calculations dimension that Hyperion Essbase provides for use with attribute dimensions? For more information about attributes, see Chapter 10, “Working with Attributes.”

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

Verifying the Design

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

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

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

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

Chapter 6

Designing Partitioned Applications

An 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:

CAUTION: 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 Hyperion 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. Hyperion 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. Hyperion Essbase tracks changes made to the outlines of partitioned databases and provides tools for updating related outlines.
- Allow users to navigate between databases with differing dimensionality. When users drill across to the new database, they can drill down to more detailed data.

Based on user requirements, you can decide to:

- Partition applications from the top down. *Top-down partitioning* allows you to split a database onto multiple processors, servers, or computers. Top-down partitioning can improve the scalability, reliability, and performance of your databases. To achieve the best results with top-down partitioning, create a separate application for each partitioned database.
- Partition applications from the bottom up. *Bottom-up partitioning* allows you to manage the flow of data between multiple related databases. Bottom-up partitioning can improve the quality and accessibility of the data in your databases.
- Partition database according to attribute values associated with base dimensions (a base dimension is a standard dimension associated with one or more attribute dimensions). Partitioning a base dimension according to its attributes enables the user to extract data based on the characteristics of a dimension such as flavor or size. For more information on attributes, see Chapter 10, “Working with Attributes.”

Data Sources and Data Targets

Partitioned databases contain at least one *data source*, the primary site of the data, and at least one *data target*, the secondary site of the data. A single database can serve as both the data source for one partition and the data target for another. When you define a partition, you map cells in the data source to their counterparts in the data target.

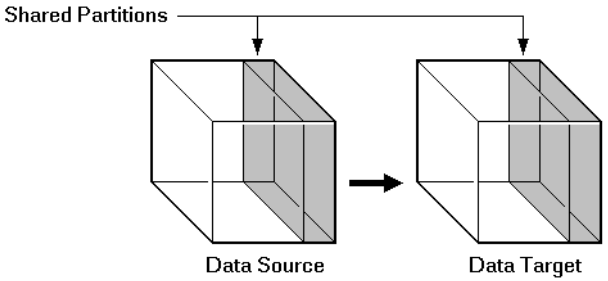


Figure 6-1: Data Source and Data Target

A Hyperion Essbase database can contain many different partitions as well as data that is not shared with any other Hyperion Essbase database. You can define partitions between:

- Different databases in different applications, as long as each database uses the same language (for example, German).
- Different databases in different applications on different processors or computers, as long as each database uses the same language (for example, German).
- Different databases in one application. This is not recommended, because you cannot reap the full benefits of partitioning your databases unless each database is in a separate application.

You can only define one partition of each type between the same two databases. For example, you can only create one replicated partition between the Sampeast East database and the Samppart Company database. The East or Company databases can, however, contain many replicated partitions that connect to other databases.

A single database can serve as the data source or data target for multiple partitions. To share data among many databases, create multiple partitions, each with the same data source and a different data target.

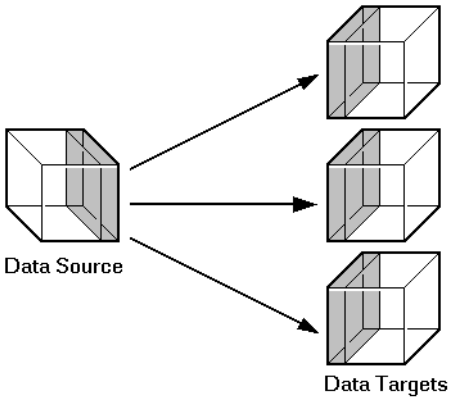


Figure 6-2: Data Shared at Multiple Targets

Overlapping Partitions

An overlapping partition occurs when similar data from two or more databases serve as the data source for a single data target in a partition. For example, `IDESC East, Sales` from database1 and `Boston, Sales` from database 2 are mapped to `IDESC East, Sales` and `Boston, Sales` in database 3. Because `Boston` is a member of the dimension `East`, the data for `Boston` mapped to database 3 from database 1 and database 2, overlap. This data overlap results in an overlapping partition.

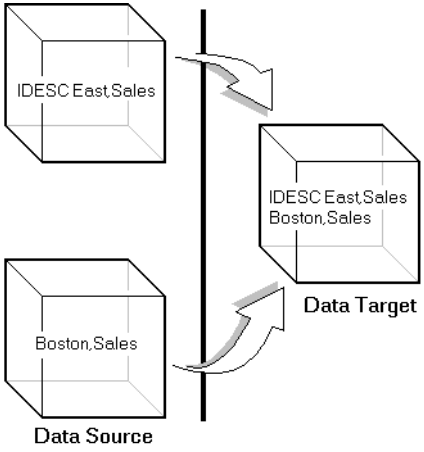


Figure 6-3: Overlapping Partitions

An overlapping partition is allowed in linked partitions, but is invalid in replicated and transparent partitions and will generate an error message during validation.

What Is a Partition?

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

- 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 Hyperion 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. Hyperion 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.

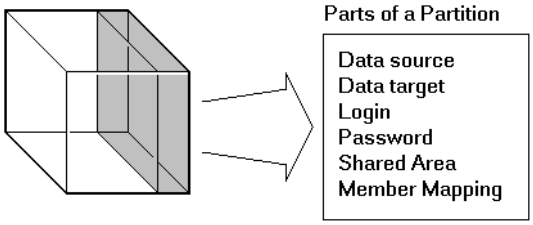


Figure 6-4: Parts of a Partition

Workflow for Partitioning a Database

Here is the suggested workflow for partitioning a database.

1. Design the partitioned databases. See "Determining Which Data to Partition" on page 6-9 and "Deciding Which Type of Partition to Use" on page 6-10.
2. Edit existing applications that use the individual databases.
   - Edit the outlines 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 6-30.
3. Create the partitions. See Chapter 16, "Building and Maintaining Partitions."
4. Load data into the partitions. Load the data into the new databases. See Chapter 20, "Introducing Data Loading."
5. If necessary, calculate the new databases. See Chapter 25, "Introduction to Database Calculations."
6. Test and maintain the partitions. Test your partitioned databases, including updates, data loads, calc scripts, report scripts, and database restructures. For more information on maintaining partitioned databases, see Chapter 16, "Building and Maintaining Partitions."

When to Partition a Database

Review the following list of questions. If you find yourself answering yes to many of them, or answering yes to some that are very important to you, partitioning your databases may solve your problems.

- Should the data be closer to the people who are using it? Is the network being stressed because users are accessing data that is far away?
- Would a single failure be catastrophic? If everyone is using a single database for mission-critical purposes, what happens if the database goes down?
- Does it take too long to perform calculations after new data is loaded? 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? Hyperion 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 6-33.

Note: You cannot partition attribute dimensions.

Deciding Which Type of Partition to Use

You can create the following types of partitions:

- A *replicated partition* is a copy of a portion of the data source that is stored in the data target. For more information, see “Replicated Partitions” on page 6-10.
- 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 Hyperion Essbase database, or on another Hyperion Essbase server. For more information, see “Transparent Partitions” on page 6-15.
- 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 6-23.

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 Hyperion 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 Hyperion Essbase, see “Scenario 1: Partitioning an Existing Database” on page 6-33.

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, Hyperion Essbase overwrites their changes when the database administrator updates the replicated partition.

The database administrator can prevent the data in the replicated portion of the data target from being updated. This setting (choose Settings from the Connect page of the Partition Wizard) takes precedence over access provided by security filters and is also honored by batch operations such as dataload and 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 16-5.

Rules for Replicated Partitions

Replicated partitions must follow these rules:

- You must be able to map the shared replicated areas of the data source and data target outlines even though the shared areas do not have to be identical. This means that you must tell Hyperion 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.

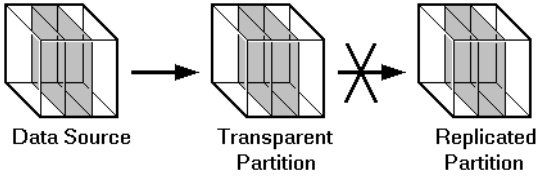


Figure 6-5: Invalid Replicated Partition

- The cells in the data target of a replicated partition cannot come from two different data sources; the cells in one partition must come from just one database. If you want to replicate cells from more than one database, create a different partition for each data source.

  The cells in a data target can be the data source for a different replicated partition. For example, if the Samppart Company database contains a replicated partition from the Sampeast East database, you could replicate the cells in the Sampeast East database into a third database, such as the Sampwest West database.

- You cannot use attribute dimension members to define a replicated partition. For example, associated with the Market dimension, the Market Type attribute dimension members are Urban, Suburban, and Rural. You cannot define a partition on Urban, Suburban, or Rural because a replicated partition contains dynamic data, not stored data. Hence, an attempt to map attributes in replicated partitions results in an error message. However, you can use the WITHATTR command to replicate attribute data in the Areas tab of the Partition Wizard. For example:

For information on using Dynamic Time Series members in replicated partitions, see “Using Dynamic Time Series Members in Partitions” on page 30-16.

Advantages of Replicated Partitions

Replicated partitions can solve many database problems. Following are the advantages of using a replicated partition:

- Replicated partitions can decrease network activity, because the data is now stored closer to the end users, in the data target. Decreased network activity results in improved retrieval times for the users.
- The data is more easily accessible to all users. Some users access the data at the data source, others at the data target.
- Failures are not as catastrophic. Because the data is in more than one place, if a single database fails, only the users connected to that database are unable to access the information. It is still available at and can be retrieved from the other sites.

- Local database administrators can control the down time of their local databases. For example, because users in the eastern region are accessing their own replicated data instead of the Company database, the database administrator can bring down the Company database without affecting the users in the eastern region.
- Because only the relevant data is kept at each site, databases can be smaller. For example, users in the eastern region can replicate just the eastern budget information, instead of accessing a larger company database containing budget information for all regions.

Disadvantages of Replicated Partitions

Replicated partitions are not always the ideal partition type. Following are the disadvantages of using a replicated partition:

- You need more disk space, because you are storing the data in two or more locations.
- The data must be refreshed regularly by the database administrator, so it is not up-to-the-minute.

If these disadvantages are too serious, consider using transparent or linked partitions instead.

Performance Considerations for Replicated Partitions

To improve the performance of replicated partitions, consider the following when replicating data.

- Not replicating members that are dynamically calculated in the data source can greatly reduce replication time, because Hyperion 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 29, “Dynamically Calculating Data Values.”
- Not replicating derived data from the data source can greatly reduce replication time. Instead, try to replicate the lowest practical level of each dimension and perform the calculations on the data target after you complete the replication.

For example, to replicate along the Market dimension:

- Define the shared area as the lowest level members of the Market dimension that you care about, for example, East, West, South, and Central and the level 0 members of the other dimensions.
- After you complete the replication, calculate the values for Market and the upper level values in the other dimensions at the data target.

Sometimes you cannot calculate derived data at the data target. In that case, you must replicate it from the data source. For example, you cannot calculate derived data at the data source if the data:

- Requires data outside the replicated area to be calculated.
- Requires 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 Hyperion 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, Hyperion 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. Hyperion Essbase does not load or replicate into Dynamic Calc and Dynamic Calc And Store members. Hyperion 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 29, “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.

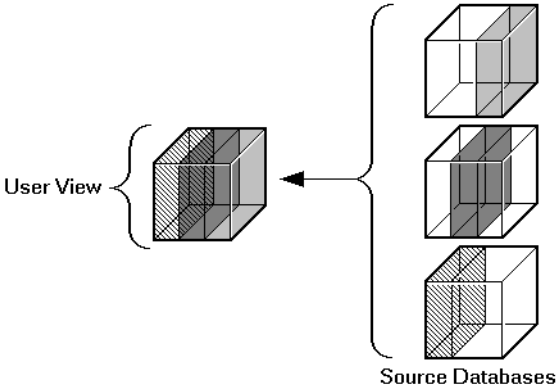


Figure 6-6: Transparent Partitions

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 6-33.

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 Hyperion Essbase how each dimension and member in the data source maps to each dimension and member in the data target.
- The data source and data target outlines for the non-shared areas do not have to be mappable. *Exception:* attribute associations should be identical. Otherwise, users can get incorrect results for some retrievals. For example, if product 100-10-1010 is associated with the Grape Flavor attribute on the source, but product 100-10-1010 is not associated with Grape on the target, the total of sales for all Grape flavors in New York would be incorrect.

- You cannot use attribute dimensions or members to define a transparent partition. For example, associated with the Market dimension, the Market Type attribute dimension has members Urban, Suburban, and Rural. You cannot define a partition on Urban, Suburban, or Rural.
- You can create a transparent partition on top of a replicated partition. In other words, you can create a transparent partition target using a replicated partition source.

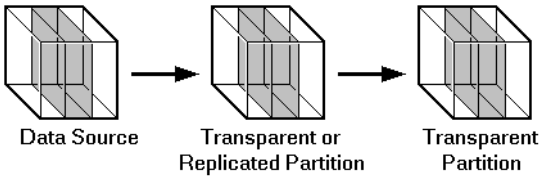


Figure 6-7: Valid Transparent Partition

- 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.

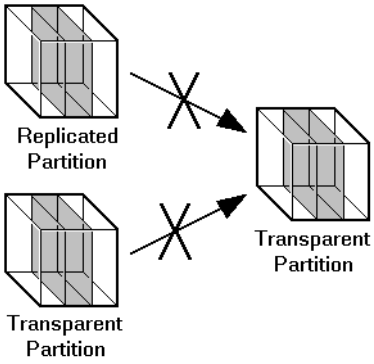


Figure 6-8: Invalid Transparent Partition

- Carefully consider any formulas you assign to members in the data source and data target. For more information, see “Transparent Partitions and Member Formulas” on page 6-22.

For more information on using Dynamic Time Series members in transparent partitions, see “Using Dynamic Time Series Members in Partitions” on page 30-16.

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, Hyperion 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 Hyperion 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, Hyperion 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, Hyperion Essbase performs the calculation using the current values of the local data and transparent dependents. Hyperion 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.

  Hyperion Essbase does not recalculate the values of transparent dependents because the outlines for the data source and the data target may be so different that such a calculation would not be accurate.

  For example, suppose that the data target outline contained a Market dimension with East, West, South, and Central members and the data source outline contained an East dimension with New York and New Jersey members. If you tried to calculate the data target outline, you would assume that East was a level 0 member. In the data source, however, East is derived by adding New York and New Jersey. Any calculations at the data target, however, would not know this and could not reflect any changes made to New York and New Jersey in the data source. To perform an accurate calculation, therefore, you must first calculate East in the data source and then calculate the data target.

  For more information on transparent calculations, see “Calculating Transparent Partitions” on page 6-21.

- Formulas assigned to members in the data source may produce calculated results that are inconsistent with formulas or consolidations defined in the data target, and vice versa. For more information, see “Transparent Partitions and Member Formulas” on page 6-22.

If these disadvantages are too serious, consider using replicated or linked partitions instead.

Performance Considerations for Transparent Partitions

To improve the performance of transparent partitions, consider the following facts when creating the partition:

- Partitioning along dense dimensions in a transparent partition can greatly slow performance. This is because dense dimensions are used to determine the structure and contents of data blocks. If a database is partitioned only along a dense dimension at the target, Hyperion Essbase must compose data blocks by performing network calls for the remote data in the transparent partition in addition to the disk I/O for the local portion of the block. To improve performance, consider including one or more sparse dimensions in the area definition so that the number of blocks required is limited to combinations with the sparse members.
- Basing transparent partitions on the attribute values of a dimension could increase retrieval time, because attributes are associated with sparse dimensions. In such cases, partitioning at a level higher than the level that is associated with attributes improves retrieval time. For example, in the Product dimension of the Sample Basic database, if children 100-10, 200-10, and 300-10 (level 0) are associated with attributes, then partition their parents 100, 200, and 300 (level 1) for better retrieval performance.
- Loading data into the data source from the data target can greatly slow performance. If possible, load data into the data source locally.
- Retrieval time is slower because users access the data over the network.
- Partitioning base dimensions can greatly slow performance.
- For calculation-related performance considerations, see “Guidelines for Transparent Calculations” on page 6-21.

When to Use a Transparent Partition

Use a transparent partition when you want to:

- Show users the latest version of the data.
- Allow users at the data target to update data.
- Decrease disk space.

Calculating Transparent Partitions

When you perform a calculation on a transparent partition, Hyperion Essbase performs the calculation using the current values of the local data and transparent dependents. When calculating local data that depends on remote data, Hyperion Essbase performs a bottom-up calculation. The bottom-up calculation can be done only when the calculator cache on the target database is used properly. For complete information on bottom-up calculations, see "Understanding Bottom-Up Versus Top-Down Calculation" on page 33-12. For information on the calculator cache, see "Using the Calculator Cache" on page 33-15.

Increasing the amount of memory assigned to the calculator cache greatly improves calculation performance with transparent partitions. When a calculation is started, a message in the server event log indicates whether or not the calculator cache is enabled or disabled on the target database. Using the calculator cache on the target database reduces the number of blocks that are requested from the data source during calculation. This, in turn, reduces the amount of network traffic that is generated by transferring blocks across the network. For information on estimating the size of the calculator cache, see "Using the Calculator Cache" on page 33-15.

Guidelines for Transparent Calculations

- Calculating data on the data target can greatly slow performance in some cases. Keep in mind that:
  - Hyperion Essbase must perform a top-down calculation on any portion of the data target that contains top-down member formulas. When the data target contains no top-down member formulas, it can perform a bottom-up calculation on the data target, which is much faster. When Hyperion Essbase performs the calculation on the data source, it can always perform a bottom-up calculation. For more information on top-down and bottom-up calculations, see "Understanding Bottom-Up Versus Top-Down Calculation" on page 33-12.
  - Hyperion 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. Hyperion Essbase performs the calculation only when users request it.
  - A replicated layer between the low-level transparent data and high-level local data.

- Keep the partition fully within the calculator cache area (see "Using the Calculator Cache" on page 33-15). Keeping a partition fully within the calculator cache means that any sparse members in the partition definition must be contained within the calculator cache. For example, in the Sample Basic database, if a partition definition includes @IDESC(East), all descendants of East must be within the calculator cache.
- Enable the calculator cache, and assign a sufficient amount of memory to it. For more information, see "Using the Calculator Cache" on page 33-15.
- Do not use complex formulas on any members that define the partition. For example, in Sample Basic, assigning a complex formula to New York or New Jersey (both children of East) forces Hyperion Essbase to use the top-down calculation method. For more information, see "Understanding Bottom-Up Versus Top-Down Calculation" on page 33-12.

Transparent Partitions and Member Formulas

If the data target and data source outlines are identical except for different member formulas, make sure that your partition definition will produce the desired calculation results.

For example, suppose that the data source and data target outlines both contain a Market dimension with North and South members, and children of North and South. On the data target, Market is calculated from the data for the North and South members (and their children) on the data source. If any of these members on the data source contain member formulas, these formulas are calculated, thus affecting the calculated value of Market on the data target. These results may be different from how the Market member would be calculated from the North and South members on the data target, where these formulas may not exist.

Make sure that any formulas you assign to members in the data source and data target will produce the desired results.

Transparent Partitions and Port Usage

One port is used for every unique user and machine combination. If a user defines several transparent partitions on one server, using the same user name, then only one port is occupied.

In a transparent partition, when a user (user1) drills into an area in the target that accesses source data, user1 is using the user name declared in the partition definition (partition user) to access the data from the source database. This causes the use of an additional port because different users (user1 and partition user) are connecting to the application.

If a second user (user2) connects to the target database and drills down to access source data, user2 also uses the user name declared in the Partition Wizard (partition user) to access the source database. Because the partition user is already connected to the source database, an additional port is not needed for the partition user, as long as user2 is accessing the same source database.

Linked Partitions

A linked partition connects two different databases with a data cell. When you click the linked cell in the data source, you drill across to a second database, the data target, and view the data there. If you are using 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 6-30.

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

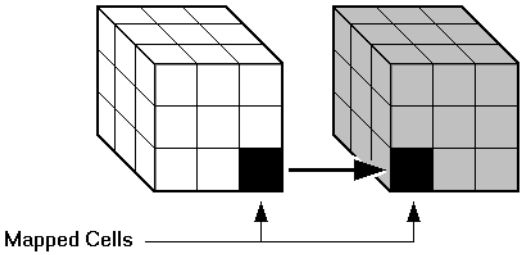


Figure 6-9: Linked Partition

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 6-38.

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 target to the data source.

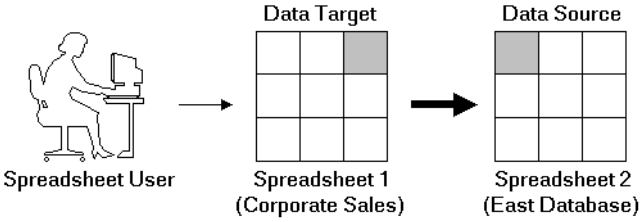


Figure 6-10: Source and Target for Linked Partition

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 6-30.
- Users must access the linked database using Hyperion Essbase Release 5-aware tools. If you have custom built your tools, you must extend them using the Hyperion Essbase Release 5 Grid API.

When to Use a Linked Partition

Use a linked partition when you want to connect databases with different dimensionality.

Linked Partitions and Port Usage

When accessing a linked partition, Hyperion 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, Hyperion 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
Do you want to map attributes associated with base dimensions?	Transparent or 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		x	x
Reduced network traffic	x		x
Reduced disk space		x	x
Increased calculation speed	x		
Smaller databases		x	x
Improved query speed	x		x
Invisible to end users	x	x	
Access to databases with different dimensionality			x
Easier to recover	x		
Less synchronization required			x
Ability to query data based on its attributes		x	x

Using Attributes in Partitions

You can use attribute functions for partitioning on attribute values. But you cannot partition an attribute dimension. Use attribute values to partition a database when you want to access members of a dimension according to their characteristics.

For example, in the Sample Basic database, you cannot partition the Pkg Type attribute dimension. But you can create a partition that contains all the members of the Product dimension that are associated with either or both members (Bottle and Can) of the Pkg Type dimension. If you create a partition that contains members associated with Can, you can access data only on Product members that are packaged in cans; namely, 100-10, 100-20, and 300-30.

You can use the @ATTRIBUTE command and the @WITHATTR command to define partitions.

For example, to extract data on all members of the Product dimension that are associated with the Caffeinated attribute dimension, you can create a partition such as @ATTRIBUTE (Caffeinated). But you cannot partition the Caffeinated attribute dimension.

Based on the previous example, the following partition is incorrect:

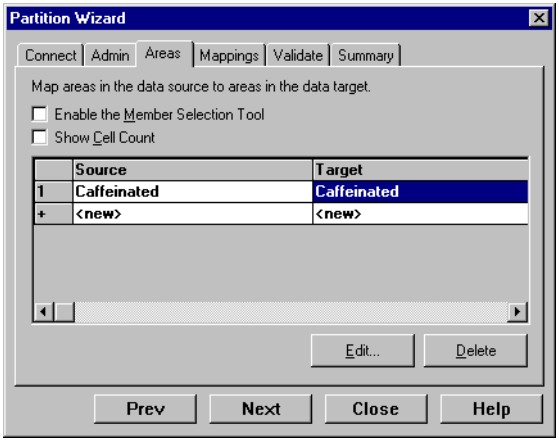


Figure 6-11: Incorrect Partitioning

Based on the previous example, the following example is correct:

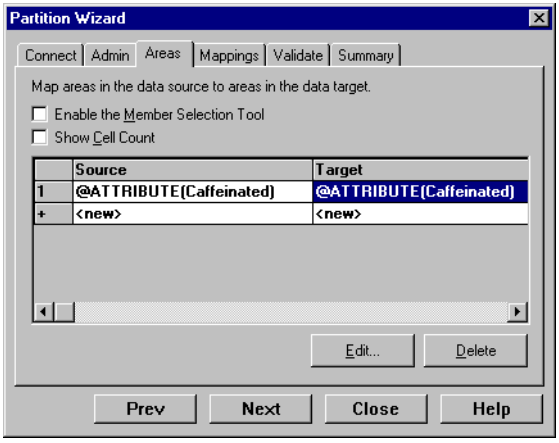


Figure 6-12: Correct Partitioning

For more information about these commands, refer to the section on Calculation Commands in the *Hyperion Essbase Quick Technical Reference*.

For more information on attribute dimensions, see Chapter 10, “Working with Attributes.”

Setting Up Security for Partitioned Databases

Users accessing replicated, transparent or linked partitions may need to view data stored in two or more databases. The following sections describe how to set up security for each partition type so that users do not view or change inappropriate data.

Setting Up Security for Replicated Partitions

To set up security for users in a replicated partition, you should:

- Create filters at the data target to determine what end users can view and update as local data, just as you would for a standard Hyperion Essbase database. See Chapter 17, “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 Hyperion 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 16-5.

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. Hyperion 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 16-6.
- If necessary, set up read filters on the administrative account at the data source to determine which data Hyperion Essbase reads when replicating.
- If necessary, set up write filters on the administrative account at the data target to determine which data Hyperion 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 Hyperion Essbase database. Remember that, unlike replicated partitions, end users can update transparent partitions unless you create filters preventing them from doing so. See Chapter 17, “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. Hyperion 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 16-6.
- Set up read filters on the administrative account at the data source to determine which data Hyperion Essbase retrieves for end users.
- Set up write filters on the administrative account at the data source to determine which data Hyperion 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 17-9.
- Create a default account that users can use when accessing target databases. For example, if users access data through a data source named Company and a data target named East, create a guest account for the East database with the appropriate privileges. Once you have created the account, use the guest account login and password as the default login when creating the linked partition. For more information on using default accounts in partitions, see “Specifying the Partition Type and Connection Information” on page 16-31.

For information on creating user accounts and filters, see Chapter 17, “Managing Security at Global and User Levels.”

When a user drills across on data to a data target, Hyperion 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, Hyperion Essbase logs the user in using that account.
2. Checks to see if you have specified a default account on the data target when you created your partition. If you did, Hyperion 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, Hyperion 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 Hyperion Essbase.

Here are the steps to take:

1. Determine which data to partition.

   TBC's Sample Basic database contains five standard dimensions: Year, Measures, Product, Market, and Scenario.

   – Partition the database along the East member of the Market dimension to give the eastern region more control over the contents of its database.

   – Partition the database along the Actual and Corp_Budget members of the Scenario dimension.

2. Choose the data source and the data target.

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 6-13 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.

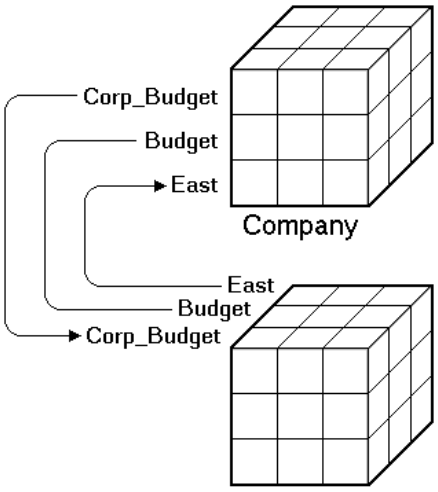


Figure 6-13: Decentralizing a Database

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 us assume that TBC has several databases, such as Inventory, Payroll, Marketing, and Sales. Users viewing the Sample Basic database want to share data with and navigate to those other databases and you, the database administrator, want to synchronize related data. It is impractical to combine all of the databases into one database, because:

- So many users access it that performance is slow.
- You could not find a down time to administer the database.
- No one has control over their own data, because it is centrally managed.
- The database is very sparse, because so much of the data is unrelated.

By connecting the databases instead, you can:

- Leverage work that's already been completed.
- Synchronize the data.

So you decide to connect multiple databases.

Note: This scenario is not shipped with Hyperion Essbase.

1. Determine which data to connect. First, connect the Inventory database.
   - Replicate the Opening_Inventory and Ending_Inventory members from the Measures dimension of the Inventory database into the Measures dimension of the Sample Basic database.
   - Do not replicate the Number_On_Hand, Number_Shipped, and Number_Returned members in the Measures dimension of the Inventory database to the Sample Basic database.
   - Add a link to the Inventory database so that users can view these more detailed measures if they need to.
   - Create a partition containing data from the Payroll, Marketing, and Sales databases in the Sample Basic database.
2. Choose the data source and the data target. In the case of the Opening_Inventory and Ending_Inventory members, the Inventory database is the data source and the Sample Basic database is the data target.
3. Decide which type of partition to use.

   You decide to use a replicated partition for the Opening_Inventory and Ending_Inventory members because the network connection is slow.
4. Connect the Payroll, Marketing, and Sales databases. Perform the tasks in 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 us assume that TBC, the fictional soft drink company upon which the Sample Basic database is based, has two main databases the Sample Basic database and TBC Demo. Both databases have similar outlines, but TBC Demo has two additional dimensions: Channel, which describes where a product is sold, and Package, which describes how the product is packaged.

The database administrator for the Sample Basic database notices that more and more users are requesting that she add channel information to the Sample Basic database. But, since she doesn't own the data for channel information, she is reluctant to do so. She decides instead to allow her users to link to the TBC Demo database which already contains this information.

Note: This scenario is not shipped with Hyperion Essbase.

Here are the steps to take:

1. Determine which data to link.

   The database administrator decides to link the Product dimension of the Sample Basic database to the Product dimension of TBC Demo. Users can then drill across to TBC Demo and view the Channel and Package information.

2. Choose the data source and the data target. Because users start at the Sample Basic database, it is considered the data target. Likewise, because users move to TBC Demo, it is considered the data source.

   Note: This is the opposite of replicated and transparent databases, where users move from the data target to the data source.

3. Decide on the type of partition to use.

Members	Partition type	Reasons
Product	Linked	The database administrator wants to link from the Product dimension.

4. Finally, create the partition 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 16-11.
   - 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 17-4. For more information on assigning those accounts to linked partitions, see "Specifying the Partition Type and Connection Information" on page 16-31.

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 Properties,” illustrates how to set attributes for the dimensions and members in an outline using the Application Manager.
- Chapter 10, “Working with Attributes,” describes attribute dimensions and members and how to define them using the Application Manager.
- Chapter 11, “Creating and Managing Aliases,” describes how to create one or more aliases for dimensions and members in an outline using the Application Manager.
- Chapter 12, “Linking Objects to Hyperion Essbase Data,” describes how to link various kinds of data with any cell in a Hyperion Essbase database.
- Chapter 13, “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 14, “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 15, “Estimating Disk and Memory Requirements for a Database,” describes how to estimate disk and memory requirements and specify settings so that you can allocate the right amount of space to run Hyperion Essbase efficiently.
- Chapter 16, “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

An 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 47, "Working with Hyperion Essbase Files and Cross-Platform Environments."

This chapter includes the following sections:

Hyperion Essbase Applications

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

A Hyperion 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:

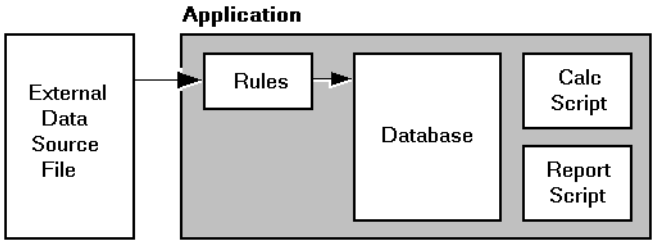


Figure 7-1: Parts of a Hyperion Essbase Application

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 7-5.

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 Hyperion Essbase performs on data from an external data source file as it is loaded, or copied, into the Hyperion Essbase database. You need to load data into a Hyperion 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 20, "Introducing Data Loading" for information on creating data load rules.

See Chapter 13, "Introducing Dynamic Dimension Building" and Chapter 14, "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 31, “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 36, “Developing Report Scripts” for information on creating report scripts.

Security Definitions

With Hyperion Essbase Application Manager, you also maintain user information, security information, application activity logs, and server activity logs. This information is saved on the Hyperion Essbase server. Within your organization, you can have multiple servers, each with its own users, applications, and databases. See Chapter 17, “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 a complete list of these steps, see Chapter 2, “Steps for Implementing Hyperion Essbase”.

Starting and Stopping Applications and Databases

➤ To 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. Select Application > Start/Stop. If you are stopping the application, a confirmation dialog box is displayed:

Figure 7-2: Stop Application Dialog Box

4. Click OK to stop the application. Hyperion 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 the `DOCS` directory for information about these commands. See Chapter 44, “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. Select Database > Start/Stop. If you are stopping the database, a confirmation dialog box is displayed:

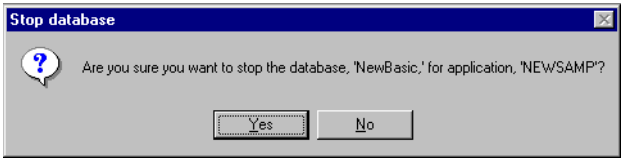


Figure 7-3: Stop Database Dialog Box

3. Click Yes to stop the database. Hyperion 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 the `DOCS` directory for information about these commands. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Using the Application Desktop Window

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

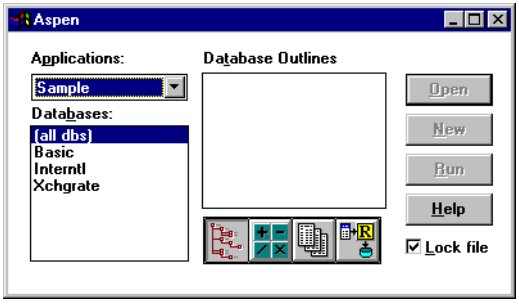


Figure 7-4: Application Desktop Window

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 is displayed as follows:

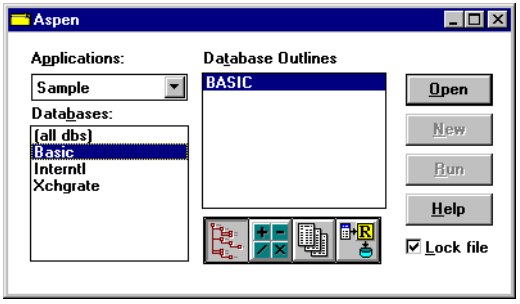


Figure 7-5: Application Desktop Window with Database Selected

When you select an application and database from the list boxes, select 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 select 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 7-11.
- To run a calc script or report script, select the file from the list, and then select Run.
- Click Help for details about this window.

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

Deciding Where to Build an Application

Before creating an application and database, you should decide:

- Whether to build the application and database on the Hyperion 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 a Hyperion 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—Hyperion Essbase Application Manager's graphical user interface or ESSCMD commands.

Creating Applications and Databases

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

Creating a New Application

➤ To create a new application with Hyperion Essbase Application Manager:

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

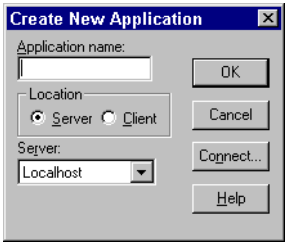


Figure 7-6: Create New Application Dialog Box

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

2. Type the name of the new application in the Application Name text box. This must be a unique name of 8 characters or less.
3. Select the client or server on which to create the application:
   - To create a client-based application, click Client. Hyperion Essbase creates a subdirectory for the application within the *essbase*`\client` directory, where *essbase* is the drive and directory into which you installed your Hyperion Essbase client. The new subdirectory has the same name as the application.
   - To create a server-based application, click Server and select one of the servers listed in the Server list box. If the Server list box is empty, follow the directions for connecting to a server. Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Creating a New Database

➤ To create a database using Hyperion Essbase Application Manager:

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

   The Application Desktop window is displayed:

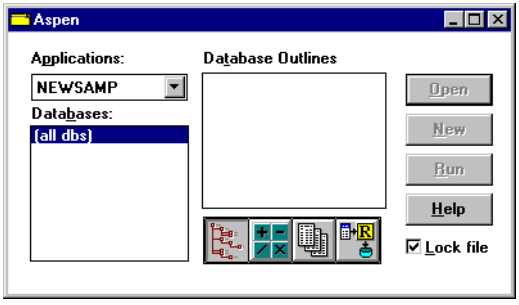


Figure 7-7: Application Desktop Window

2. From the Applications list box, select the name of the application in which you want to create the database.
3. Select File > New > Database. The following dialog box is displayed:

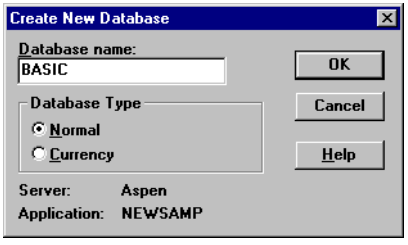


Figure 7-8: Create New Database Dialog Box

4. Type the name of the new database in the Database name text box. The Hyperion Essbase Application Manager limits the database name to 8 characters.
5. In most cases, the database type is Normal. To create a currency database, select a database type of Currency. For more information on currency databases, see Chapter 43, "Designing and Building Currency Conversion Applications."
6. Click OK to create the database. Hyperion 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 7-9. 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 the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Rules for Naming Applications and Databases

When naming applications and databases, follow these rules:

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

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

- Do not use spaces anywhere in the name.

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 Hyperion Essbase 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. Select Database > Set Note. The following dialog box is displayed:

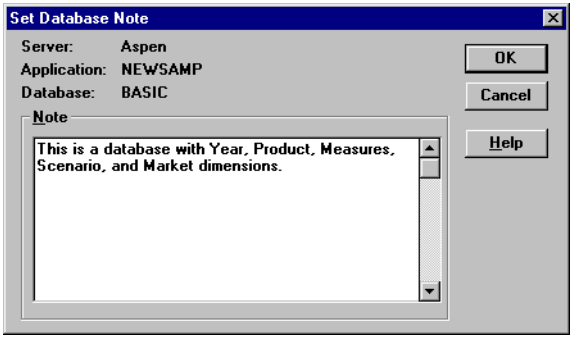


Figure 7-9: Set Database Note Dialog Box

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

Using Dynamic Calculations

When designing and creating your Hyperion 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 29, “Dynamically Calculating Data Values” and Chapter 30, “Calculating Time Series Data.”

Using Substitution Variables

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

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

You can use substitution variables in calc scripts, report scripts, or in Hyperion Essbase Spreadsheet Add-in. You cannot use substitution variables in formulas that you apply to the database outline. For information about using substitution variables, refer to the following:

- For calc scripts, see Chapter 31, “Developing Calc Scripts.”
- For Report Writer, see Chapter 36, “Developing Report Scripts.”
- For Hyperion Essbase Spreadsheet Add-in, see the *Hyperion Essbase Spreadsheet Add-in User’s Guide*.

You can set substitution variables on the server using either Hyperion Essbase 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

Guidelines for Setting Substitution Variables

Keep in mind the following guidelines when setting substitution variables:

- The substitution variable name must be composed of alphanumeric characters or underscores (_) and cannot exceed 80 characters. This name cannot include non-alphanumeric characters, such as hyphens (-), asterisks (*), and slashes(/).
- The value of the substitution variable cannot exceed 256 characters. You can use any combination of characters in the value name. The value may contain any character except the leading ampersand (&).
- If the value of the substitution variable is numeric, you must enclose it in quotes. For example, if the variable name is Month and its corresponding value is 01 (corresponding to January), place quotes around 01 (“01”).

Setting a Substitution Variable

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

➤ To set a substitution variable using Hyperion Essbase Application Manager:

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

   Hyperion Essbase displays the Substitution Variables dialog box:

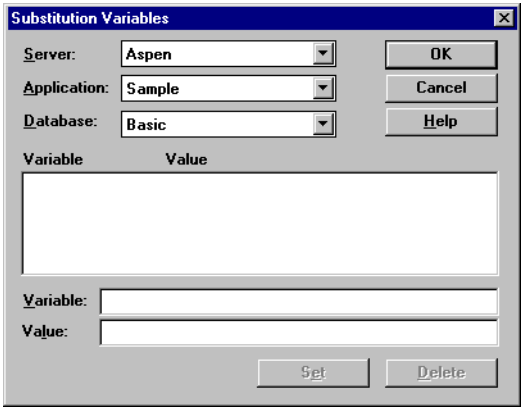


Figure 7-10: Creating a Substitution Variable

2. In the Server list box, select the server to apply the variable to.
3. In the Application list box, select the application to apply the variable to. Select "(all apps)" to apply the variable to all applications on the server.
4. In the Database list box, select the database to apply the variable to. Select "(all dbs)" to apply the variable to all databases in the selected application.
5. In the Variable box, type the new variable name. See "Guidelines for Setting Substitution Variables" on page 7-15.
6. In the Value box, type the value name. See "Guidelines for Setting Substitution Variables" on page 7-15.
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 the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Deleting a Substitution Variable

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

➤ To delete a substitution variable using Hyperion Essbase Application Manager:

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

   Hyperion Essbase displays the Substitution Variables dialog box:

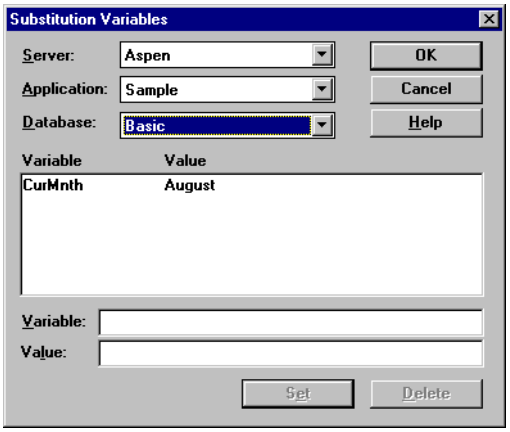


Figure 7-11: Deleting a Substitution Variable

2. In the Server list box, select the server to delete the variable from.
3. In the Application list box, select the application to delete the variable from. If the variable applies to all applications on the server, select “(all apps)”.
4. In the Database list box, select the database to delete the variable from. If the variable applies to all databases on the server, select “(all dbs)”.

5. In the Variable list, select the variable you want to delete.
6. Click Delete to delete the variable and value.

You can use the DELETEVARIABLE command in ESSCMD to perform this task. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Updating a Substitution Variable

You can modify or update existing substitution variables.

➤ To update a substitution variable using Hyperion Essbase Application Manager:

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

   The name of the existing variable and its current value are displayed in the list.
5. In the Variable box, type the name of the existing variable exactly as it is displayed in the Variable list.
6. In the Value box, type the value name. See "Guidelines for Setting Substitution Variables" on page 7-15.

7. Click Set to apply the new value to the existing variable.

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

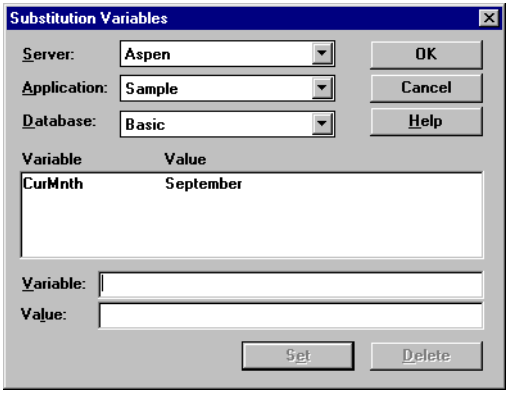


Figure 7-12: Updating a Substitution Variable

You can use the UPDATEVARIABLE command in ESSCMD to perform this task. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, “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 the `DOCS` directory for information about this command. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Using Location Aliases

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

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

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

Creating Location Aliases

You can create a location alias for a particular database.

➤ To create a location alias using Hyperion Essbase Application Manager:

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

   Hyperion Essbase displays the Location Aliases dialog box.

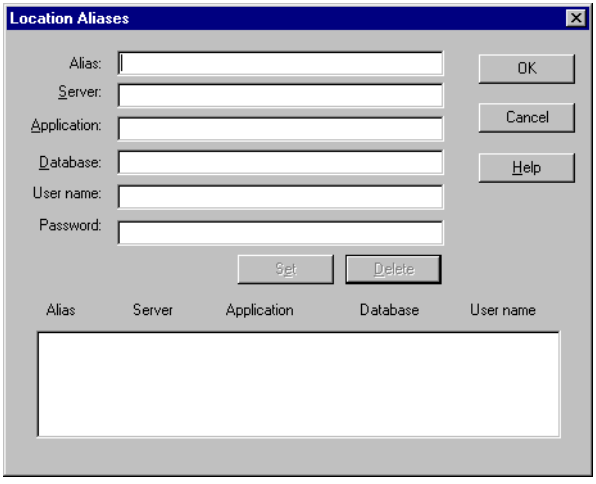


Figure 7-13: Location Aliases Dialog Box

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

   Hyperion Essbase displays the settings in the lower text box.

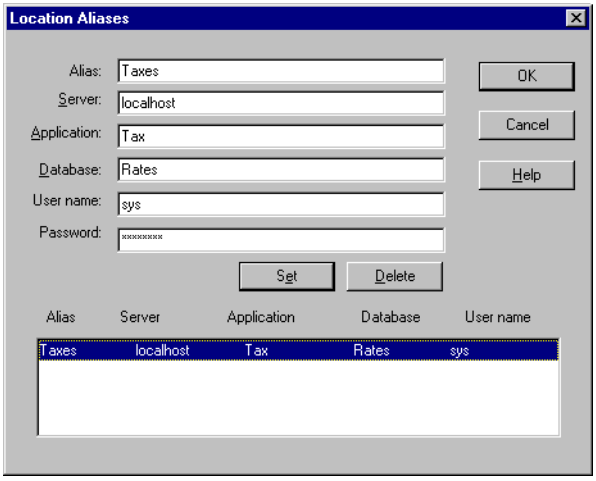


Figure 7-14: Creating a Location Alias

5. Click OK.

You can use the CREATELOCATION command in ESSCMD to perform this task. See the online *Technical Reference* in the `DOCS` directory for information about these commands. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Editing or Deleting Location Aliases

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

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

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

   Hyperion Essbase displays the Location Aliases dialog box.

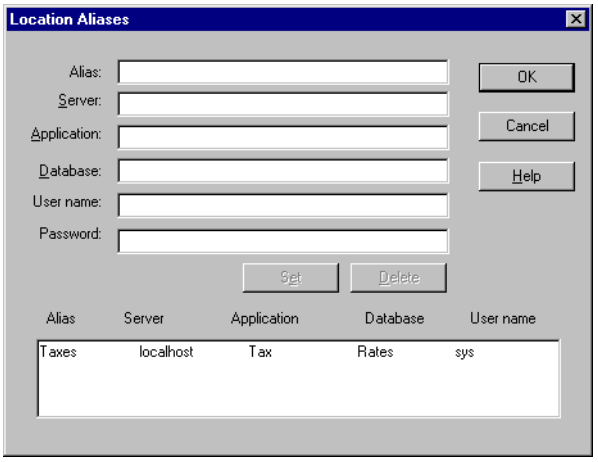


Figure 7-15: Editing or Deleting a Location Alias

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

You can use the LISTLOCATIONS command to list existing location aliases. You can use the DELETELOCATION command in ESSCMD to delete location aliases. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, “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 an 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 13, "Introducing Dynamic Dimension Building." All examples in this chapter are based on the Sample Basic database shipped with Hyperion Essbase.

This chapter contains the following sections:

For information on setting properties, see Chapter 9, "Setting Dimension and Member Properties."

Creating Outlines

The database outline defines the structure of the database. For more information about outlines, see “Arranging Dimensions into Hierarchies” on page 3-4.

➤ To create an outline:

1. Create a new database. See “Creating a New Database” on page 7-11.
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 8-15.
   - 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, Hyperion Essbase loads the outline into memory on your client machine and you can view and manipulate its dimensions and members graphically.

➤ To open an outline:

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

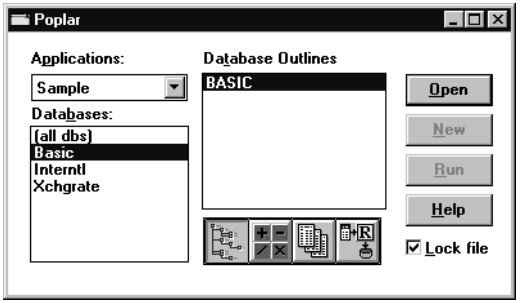


Figure 8-1: Application Desktop Window

3. If Lock file is selected, 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.

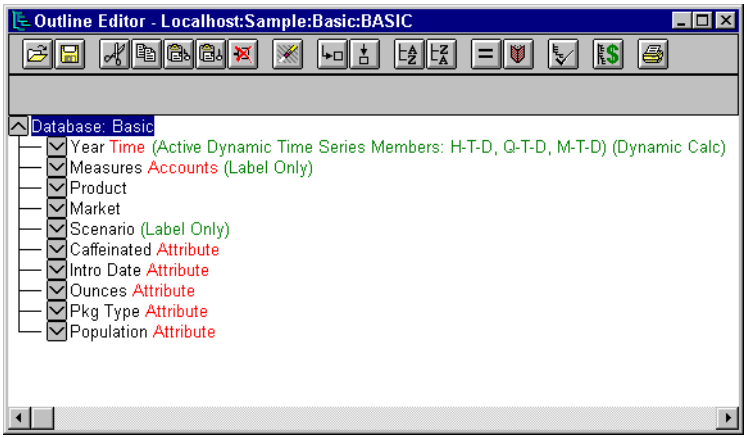


Figure 8-2: Outline Editor

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. Select Database > Copy. Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, "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. Select File > Save As to open the Save Server Object dialog box.

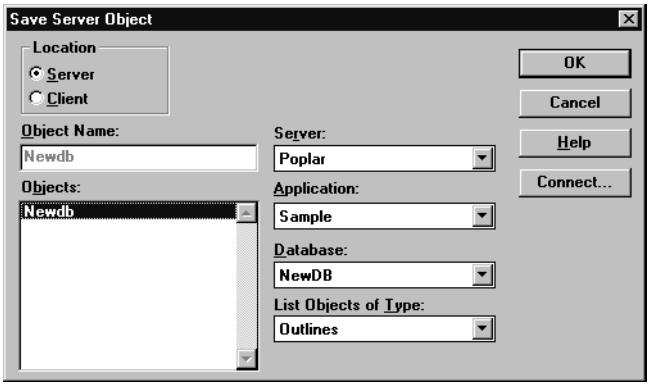


Figure 8-3: Save Server Object Dialog Box

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 is displayed, asking if you should overwrite the existing database. Click Yes.

Verifying and Saving Outlines

When you save an outline to the server, Hyperion Essbase Application Manager verifies it. Hyperion Essbase Application Manager 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. The section "Rules for Naming Dimensions and Members" on page 8-15 presents conventions for member names.
- Only one dimension is tagged as accounts, time, currency type, or country.
- Shared members are valid. Shared members with formulas or children cannot occur before the actual members on which they are based. Shared members do not have any attributes or UDAs. 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 of standard dimensions have a formula.
- Dynamic Calc members do not have more than 100 children.
- Boolean attribute dimensions have only two members. Boolean attribute names are the same as the Boolean attribute member names defined for the database.
- The level 0 member name of a date attribute dimension must match the date format name setting (mm-dd-yyyy or dd-mm-yyyy). If the dimension has no members, because the dimension name is the level 0 member, the dimension name must match the setting.
- The level 0 member name of a numeric attribute dimension is a numeric value. If the dimension has no members, because the dimension name is the level 0 member, the dimension name must be a numeric value.
- Attribute dimensions are located at the end of the outline, following all standard dimensions.

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

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

For more information about numeric attribute dimensions, see “Attribute Types” on page 10-6.

Verifying Outlines

➤ To verify an outline:

1. Select Outline > Verify or click the Verify,  , button. If the outline has no errors, the Verify dialog box is displayed:

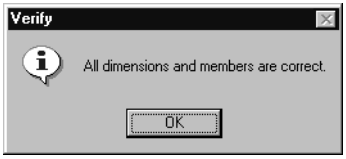


Figure 8-4: Verify Dialog Box

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

Saving Outlines

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

➤ To save an outline:

1. Select File > Save. Hyperion Essbase saves the outline.

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

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

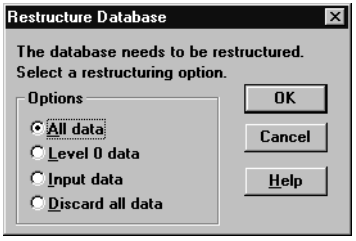


   Figure 8-5: Restructure Database Dialog Box

3. Select the data that Hyperion Essbase should restructure.

   - Select All data when you want to keep all changes that pertain to the modified outline.
   - Select Level 0 data when you expect to recalculate the database. If all data in the outline is at level 0, choosing Level 0 data can save space and improve calculation performance.
   - Select Input data when you expect to calculate and you have loaded data into non-level 0 members.
   - Select Discard all data when you expect to reload the data or when your outlines is so radically changed that no existing data applies.

   For more information, see "Database Restructuring" on page 40-14.

4. Click OK.

Note: Be careful when using older versions of Hyperion Essbase Application Manager to open and save outlines that are created with newer versions of Hyperion 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 Hyperion Essbase Application Manager. For more information, see the *Hyperion Essbase Start Here* booklet.

Setting Dense and Sparse Data Storage

When you create and save an outline, Hyperion 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.

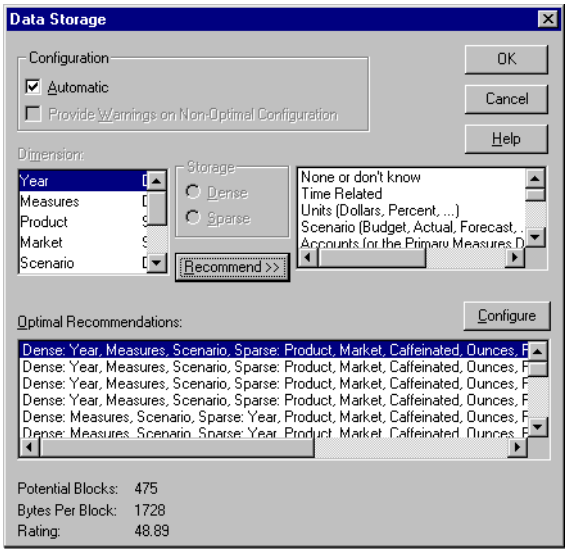


Figure 8-6: Data Storage Dialog Box

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

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

For information on choosing dense or sparse storage, see “Selection of Sparse and Dense Dimensions” on page 4-8.

Renaming Dimensions and Members

You can change the names of existing dimensions or members.

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

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

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

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

Importing and Exporting 2.x Outlines

You can import Version 2.x outlines that have been exported to text files.

CAUTION: When you export an outline to a text file, Hyperion Essbase does *not* export any functionality added to the outline since Release 2.x of the product.

Exporting Outlines

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

CAUTION: Only use this feature to export outlines with Release 2.x functionality.

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

1. Open the outline to export. If you do not know how to do this, see "Opening Outlines" on page 8-2.
2. Select File > Export > Outline to open the Export Server Outline Object dialog box.

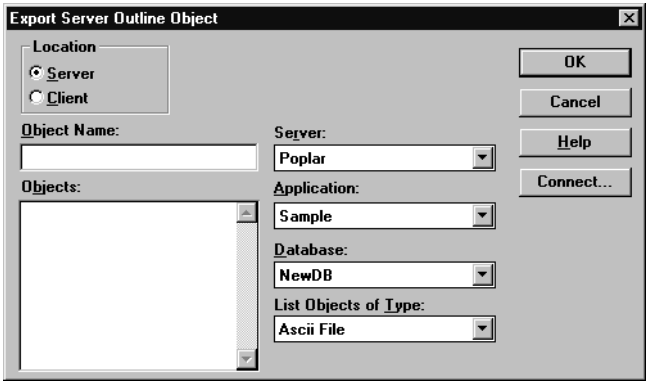


Figure 8-7: Export Server Outline Object Dialog Box

3. Make sure the appropriate Hyperion 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.

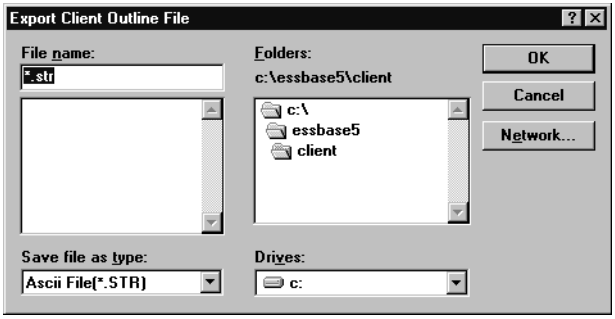


Figure 8-8: Export Client Outline File Dialog Box

5. Enter the name of the file to which to export the outline. Hyperion Essbase appends a `.STR` to the file name if you accept the default client or server directory. If you clicked the File System button, you must type in the `.STR` extension.
6. Click OK. Hyperion 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.

CAUTION: Only use this feature to import outlines with Release 2.x functionality.

➤ To import an outline:

1. Open the outline into which you want to import the outline. If you do not know how to do this, see "Opening Outlines" on page 8-2.
2. Select File > Import > Outline to open the Import Server Outline Object dialog box.

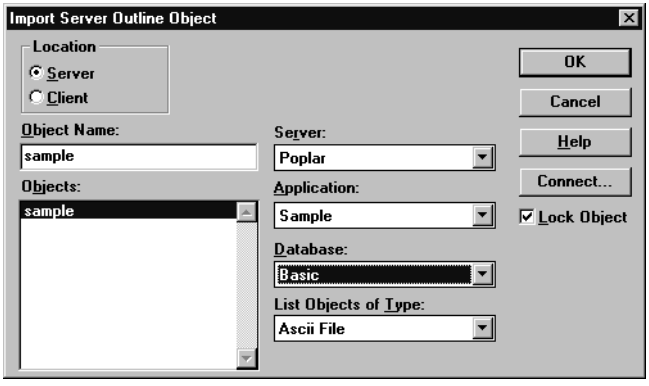


Figure 8-9: Import Server Outline Object Dialog Box

3. Make sure the appropriate Hyperion 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 select "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. Select 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.

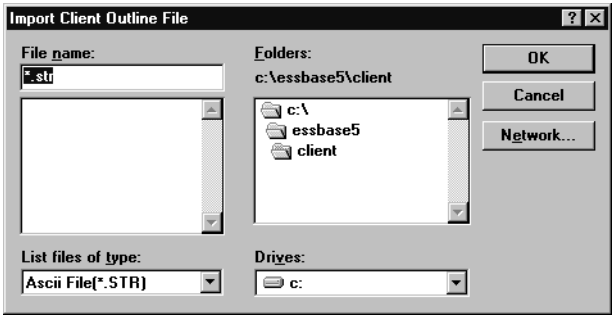


Figure 8-10: Open Client Outline File Dialog Box

5. Click OK. Hyperion Essbase imports the outline.

Use the LISTMEMBERS command in ESSCMD to import outlines. See the online *Technical Reference* in the `DOCS` directory for information about this commands. See Chapter 44, "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:

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

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

Rules for Naming Dimensions and Members

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

- The maximum length is 80 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)
  {} (braces)
  , (comma)
  - (dash, hyphen, or minus sign)
  = (equal sign)
  < (less than sign)
  ( ) (parentheses)
  . (period)
  + (plus sign)
  ' (single quotation mark)
  _ (underscore)
  | (vertical bar)

- Do not place spaces at the beginning or end of a name. Hyperion Essbase ignores 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, operators, and keywords. For a list of commands, see the online *Technical Reference* in the `DOCS` directory.
  - Report script commands. For a list of commands, see the online *Technical Reference* in the `DOCS` directory.
  - Function names and function arguments. For a list of functions, see the online *Technical Reference* in the `DOCS` directory.
  - Names of other dimensions, members (unless the member is shared), generation names, level names, aliases, and combinational aliases in the database

  Note: If you enable Dynamic Time Series members, do not use the associated generation names: History, Year, Season, Period, Quarter, Month, Week, or Day. See "Applying Predefined Generation Names to Dynamic Time Series Members" on page 30-14.

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

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

> (greater than sign)
< (less than sign)
( ) (parentheses)
% (percent sign)
. (period)
+ (plus sign)
; (semicolon)
/ (slash)

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

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

Tips to make names unique:

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

Adding Dimensions to Outlines

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

➤ To add dimensions as children of the outline:

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

➤ To add dimensions as siblings of other dimensions:

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

Figure 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 “Rules for Naming Dimensions and Members” on page 8-15 for restrictions.
2. Press Enter. A dialog box opens asking if you are sure that you want to add the dimension.

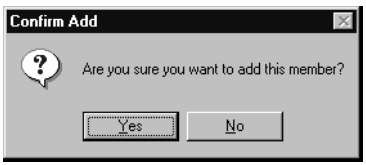


Figure 8-12: Confirm Add Dialog Box for Dimensions and Members

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 8-20.

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

➤ To add members as siblings of other members:

1. Select the member after which you want to add the new member; for example, Qtr1.
2. Select Edit > Add Sibling or click the Add Sibling button,  , to open a member edit text box in the Outline Editor.

Figure 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 "Rules for Naming Dimensions and Members" on page 8-15 for restrictions.

2. Press Enter. A dialog box opens asking if you are sure that you want to add the member.

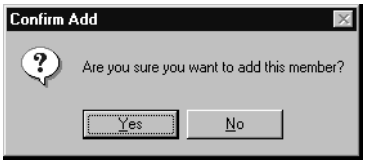


Figure 8-14: Confirm Add Dialog Box for Dimensions and Members

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 8-20.

Positioning Dimensions and Members

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

- "Sorting Dimensions and Members" on page 8-21
- "Moving Members" on page 8-22

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

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

The positions of dimensions in an outline can affect performance. See "Outline Performance Considerations" on page 10-14.

Sorting Dimensions and Members

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

When you sort level 0 members of numeric attribute dimensions in outlines, the members are sorted by their values. For example, Figure 8-15 shows text and numeric versions of the Sizes attribute dimension after sorting the members in ascending order. The members of the numeric attribute dimension are sequenced by the numeric values of the members; the member 8 is before the other members. In the text attribute dimension, because the characters are sorted left to right, the member 8 is after the member 24.

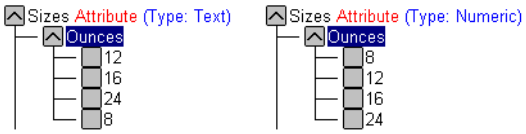


Figure 8-15: Sorting Numeric Versus Text Attribute Dimension in Ascending Order

You cannot sort Boolean attribute dimensions. For more information about attribute dimension types, see “Attribute Types” on page 10-6.

CAUTION: Moving dimensions and members can affect the results of your batch calculations. See “Designing an Outline to Optimize Performance” on page 5-23. Also, sorting members could move a shared member before the actual member in the outline, something that we do not recommend.

➤ To sort all dimensions in an outline:

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

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

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

Moving Members

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

CAUTION: Moving dimensions and members can affect the results of your calculations and retrievals. See "Designing an Outline to Optimize Performance" on page 5-23. 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  is displayed.
3. Place the icon over a member at the new location. A border is displayed around the member name.
   - To insert the member as a sibling of the member, place the icon on the  , or to the left of the border. The icon is displayed with a straight line  .
   - To insert the member as a child of the member, place the icon in or to the right of the border. The icon is displayed with a bent line  .
4. Release the mouse button.

Naming Generations and Levels

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

Use generation and level names in 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 31, “Developing Calc Scripts” for information about developing calc scripts.

See “Member Relationships, Generations, and Levels” on page 3-5 for information about generations and levels.

➤ To create a generation name:

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

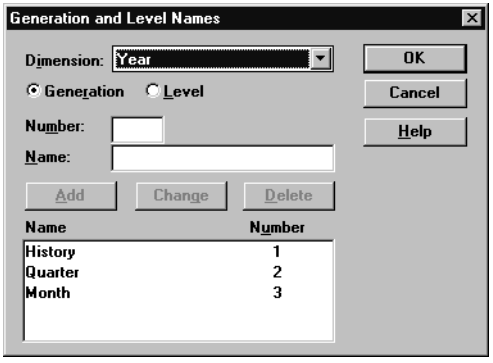


Figure 8-16: Generation and Level Names Dialog Box

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

5. Enter the generation or level name in the Name text box. For example, to name the months in the Sample Basic database, enter `Months`.

   Note: You can define only one name for each generation or level. When you name the generations and levels, follow the same naming rules as for members. See “Rules for Naming Dimensions and Members” on page 8-15.

6. Click Add. The new name is displayed in the list box.
7. Click OK.

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

Customizing the Outline Editor

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

Making Members Case-Sensitive

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

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

Customizing the View

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

- Consolidation Objects—displays consolidation operators (such as +, -, and %) in parentheses to the right of the member. For information on setting consolidation operators, see “Setting Member Consolidation Properties” on page 9-18.
- Formula Objects—displays formulas to the right of the member. For information on creating formulas, see “Naming Generations and Levels” on page 8-23.
- Dimension Tags—displays dimension tags to the right of the member, if a dimension tag other than None is selected. For information on adding dimension tags, see “Setting the Dimension Type” on page 9-2.
- Aliases—displays aliases in parentheses to the right of the member. For more information on creating aliases, see Chapter 11, “Creating and Managing Aliases.”
- Member Properties—displays member properties to the right of the member. For more information on assigning properties, see Chapter 9, “Setting Dimension and Member Properties.”
- Comments—displays comments to the right of the member. For more information on adding comments to members, see “Setting Comments on Dimensions and Members” on page 9-27.
- Attribute Associations—displays associated attribute dimension names to the right of the base dimension and associated attribute dimension members to the right of base dimension members. For information on associating attribute dimensions and members to base dimensions, see Chapter 10, “Working with Attributes.”
- Attributes Type—displays the attribute dimension type to the right of attribute dimension names. For information on attribute dimension types, see “Attribute Types” on page 10-6.
- Toolbar—displays the Outline Editor toolbar, which contains shortcuts for working with outlines, dimensions, and members.
- Properties Bar—displays the Outline Editor properties bar, which contains shortcuts for changing or assigning properties to dimensions and members.

Setting the Outline Font

You can specify the font that Hyperion Essbase uses to display text in the Outline Editor. To set a font:

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

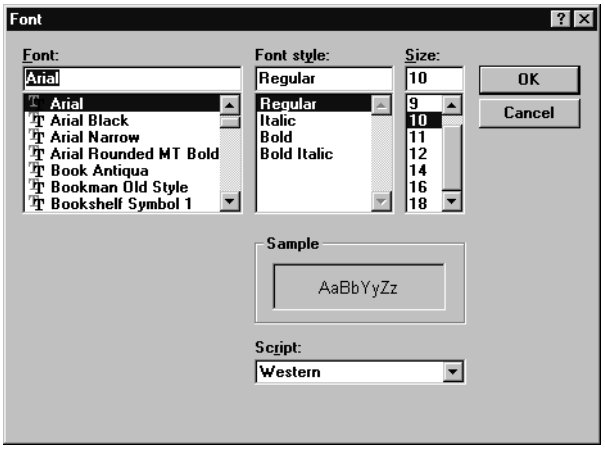


Figure 8-17: Font Dialog Box

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

Confirming Changes to Dimensions and Members

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

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

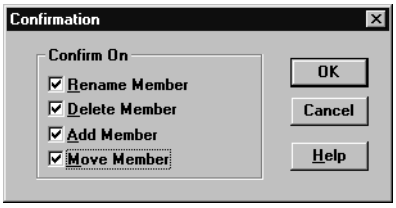


Figure 8-18: Confirmation Dialog Box

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

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

3. Click OK.

Chapter 9

Setting Dimension and Member Properties

After you create and organize your Hyperion Essbase OLAP Server outline, as described in Chapter 8, “Creating and Changing Database Outlines,” you are ready to specify how the dimensions and members in the outline behave. This chapter describes each dimension and member property and how to set them in the Outline Editor in Hyperion Essbase Application Manager.

Note: For information on setting Dynamic Time Series members, see Chapter 30, “Calculating Time Series Data.”

Setting the Dimension Type

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

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

The following sections describe how to tag dimensions:

- "Tagging a Time Dimension" on page 9-2
- "Tagging an Accounts Dimension" on page 9-3
- "Tagging a Country Dimension" on page 9-11
- "Tagging a Currency Partition" on page 9-12
- "Tagging an Attribute Dimension" on page 9-13

Tagging a Time Dimension

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

Follow these rules when tagging a dimension as time:

- You can only tag one dimension in an outline as time.
- When you tag a dimension as time, all members in that dimension inherit the time property.

  Note: 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 Property

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

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

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

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

Tagging an Accounts Dimension

Tag a dimension as accounts if it contains items that you want to measure, such as profit or inventory.

Follow these rules when tagging an accounts dimension:

- You can only tag one dimension in an outline as accounts.
- When you tag a dimension as accounts, all members in that dimension inherit the accounts property.
- To calculate members of the accounts dimension on the second pass through the outline, see "Setting Two-Pass Calculation Properties" on page 9-15.

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

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

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 Property

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

1. Select the dimension that you want to tag; for example, Measures.
2. Click the Accounts button, 

. “Accounts” displays next to the dimension name.

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

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

Introducing Time Balance Properties

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

Example of Setting the Time Balance as None

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

Example of Setting the Time Balance as First

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

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

To do this, tag OpeningInventory as first. Now Hyperion 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 Hyperion 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 Hyperion 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 Properties

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

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

If you select...	Hyperion Essbase...
None	Does not skip data when calculating the parent value. However, if Hyperion 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 property 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 property 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 Property

Setting Time Balance Properties

➤ To use buttons to set a time balance property:

1. Select the member that you want to set the property for; for example, OpeningInventory.
2. Click the button that corresponds to the property that you want to set.

   - To set the time balance property as first, click the First button,  . "TB First" displays next to the member's name.
   - To set the time balance property as last, click the Last button,  . "TB Last" displays next to the member's name.
   - To set the time balance property as average, click the Average button,  . "TB Average" displays next to the member's name.
   - To set the time balance property as none, click the None button,  .
3. Click the button that corresponds to the skip property that you want to set.

   - To not skip any values, click the None button,  .
   - To skip missing values, click the Missing button,  .
   - To skip zero values, click the Zero button,  .
   - To skip both zero and missing values, click the Zero and Missing button,  .

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

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

Setting Variance Reporting Properties

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

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

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

By default, members are non-expense.

➤ To use a button to set an expense property:

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

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

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

Setting Hyperion Essbase Currency Conversion Properties

Currency conversion properties define categories of currency exchange rates. These properties are used only in currency databases. For more information on currency properties, see Chapter 43, “Designing and Building Currency Conversion Applications.”

If you select...	Hyperion 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 properties in the Outline Editor:

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

Tagging a Country Dimension

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

In a country dimension, you can specify the type of currency used in each member. For example, in the Interntl application and database shipped with Hyperion 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 43, "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" displays next to the dimension name.

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

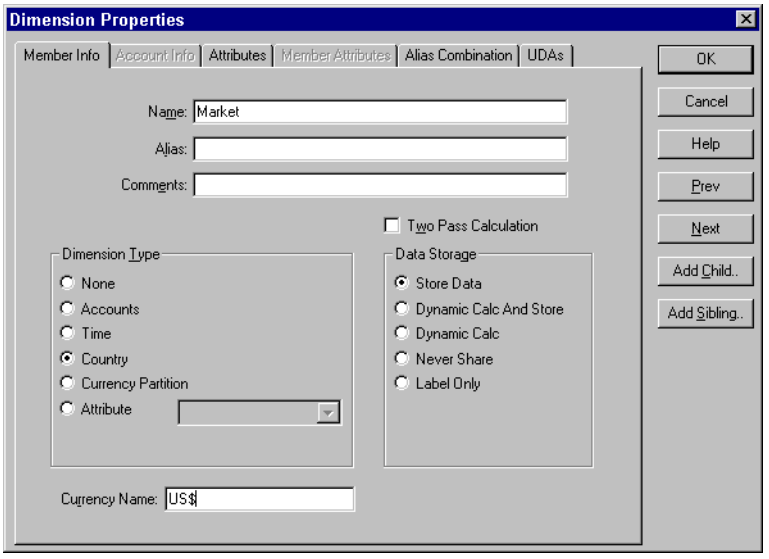


Figure 9-5: Country Dimension Tagged in Dimension Properties Dialog Box

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 “Modify the Scenario Dimension” on page 43-14.

Tagging an Attribute Dimension

Use attribute dimensions to report and aggregate data based on characteristics of standard dimensions. In the Sample Basic database, for example, the Product dimension is associated with the Ounces attribute dimension. Members of the Ounces attribute dimension categorize products based on their size in ounces. For information about attribute dimensions, see Chapter 10, “Working with Attributes.”

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

- You can tag only sparse dimensions as attribute.
- Before you can save an outline to the server, each attribute dimension must be associated with a standard, sparse dimension as its base dimension.
- Attribute dimensions must be the last dimensions in the outline.
- Attribute dimensions have a type setting: text, numeric, Boolean, or date. Text is the default setting. For more information, see “Attribute Types” on page 10-6.
- If you remove the attribute tag from a dimension, Hyperion Essbase Application Manager removes prefixes or suffixes from its member names. Prefixes and suffixes are not visible in the outline. For more information, see “Defining a Prefix or Suffix Format for the Names of Members of Attribute Dimensions” on page 10-16.

Setting the Attribute Property

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

1. Select the dimension that you want to tag; for example, Ounces.
2. Click the Attribute button,  . Outline Editor displays “Attribute” next to the dimension name.

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

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

Setting the Attribute Type

Attributes have a text, Boolean, date or numeric type property. Although assigned at the dimension level, the type applies only to the level 0 members of the dimension. Hyperion Essbase sets text as the default attribute dimension type. See “Attribute Types” on page 10-6 for more information.

➤ To use buttons to set the attribute type:

1. Select the attribute dimension; for example, Ounces.
2. Click the button corresponding to the type that you specify:
   - For text, click the  button, for example, for member names Bottle and Can.
   - For Boolean, click the  button, for example, for attributes describing binary situations such as TRUE and FALSE.
   - For date, click the  button, for example, for 09-15-1999. Date attributes cannot support member names like September 15, 1999.
   - For numeric, click the  button, for example, for attribute names you use in calculations such as 12 or 16.

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

1. Select the dimension that you want to tag; for example, Ounces.

2. Click the Data Dictionary button,  , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.
3. Select the dimension type in the drop-down list box next to Attribute in the Dimension Type box.

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

4. Click OK.

Setting Two-Pass Calculation Properties

By default, Hyperion 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, Hyperion Essbase must first calculate the outline and then re-calculate the members that are dependent on the calculated values of other members. The members that are calculated on the second pass through the outline are called *two-pass calculations.*

For more information on bottom-up calculations, see “Understanding Bottom-Up Versus Top-Down Calculation” on page 33-12.

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

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

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

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

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

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

Introducing Member Consolidation Properties

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

Note: Hyperion Essbase does not use consolidation properties with members of attribute dimensions. The Attribute Calculations dimension provides consolidation totals for attribute dimensions. See "Calculating Attribute Data" on page 10-30.

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, Hyperion Essbase calculates the data in top-down order. The following section describes how Hyperion 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

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

Setting Member Consolidation Properties

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

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

1. Select the member.
2. Click the button corresponding to the consolidation you specify:
   - For addition, click the  button.
   - For subtraction, click the  button.
   - For multiplication, click the  button.
   - For division, click the  button.
   - For percents, click the  button.
   - To exclude the member from the consolidation, click the  button.

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

1. Select the member.

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

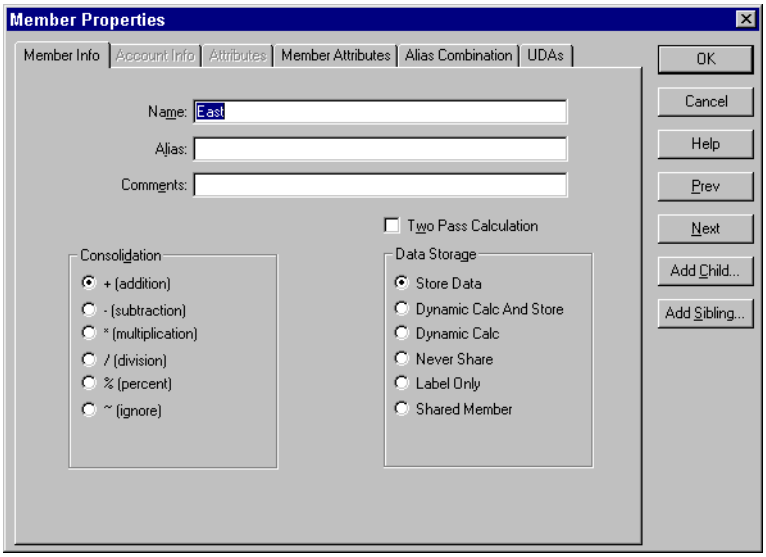


Figure 9-10: Setting Consolidation Properties

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

Setting Storage Properties

You can determine how and when Hyperion Essbase stores the data values for a member. For example, you can tell Hyperion 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 property and tells you when to set it and where to go to learn how to set it.

Table 9-2: Choosing Storage Properties

Use this property...	When you want to...	For more information, see...
Store	Store the data value with the member.	"Storing Data" on page 9-20
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 9-21
Dynamic Calc	Not calculate the data value until a user requests it, and then discard the data value.	"Dynamically Calculating Data" on page 9-21
Never share	Not allow members to be shared implicitly.	"Implied Sharing" on page 9-23
Label only	Create members for navigation only, that is, members that contain no data values.	"Creating Label Only Members" on page 9-22
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 9-23

Storing Data

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

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

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

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

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

Dynamically Calculating Data

When a member is Dynamic Calc, Hyperion Essbase does not calculate the value for that member until a user requests it. After the user views it, Hyperion Essbase does not store the value for that member. If you tag a member as Dynamic Calc And Store, Hyperion Essbase performs the same operation as for a Dynamic Calc member, except that Hyperion 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 29, “Dynamically Calculating Data Values.”

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

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

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

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

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

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

Creating Label Only Members

Label only members have no data associated with them. Use them to group members or to ease navigation and reporting from the Hyperion Essbase Spreadsheet Add-in. Typically, you should give label only members the no consolidation property. For more information on the no consolidation property, see "Setting Member Consolidation Properties" on page 9-18.

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

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

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

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

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

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. Hyperion Essbase stores the data value only once, but it displays 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 UDAs, formulas, consolidation properties, or account properties to shared members.
- You cannot associate attributes with shared members.
- You can assign aliases to shared members.
- You should not create an outline where shared members are located before actual members in a dimension.

Implied Sharing

The shared member property defines a shared data relationship explicitly. Some members are shared even if you don't explicitly set them as shared. These members are said to be *implied shared members*.

Hyperion 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. Hyperion Essbase ignores the consolidation property on the child and stores the data only once—thus the parent has an implied shared relationship with the child. In Figure 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. Hyperion Essbase ignores the consolidation property on the child and stores the data only once—thus the parent has an implied shared relationship with the child. In Figure 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.

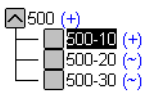


Figure 9-12: Implied Sharing of a Parent with Multiple Children

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 Property” on page 9-25.

Setting the Shared Member Property

➤ To tag a member as shared in the Outline Editor:

1. Select the shared member.
2. Click the Shared Member button,  . "Shared Member" displays next to the member name.
3. If the shared member and the actual member roll up to multiple parents, their values are counted twice in a consolidation of the database. If you want to prevent this, select the shared member and click the No Consolidate button,  . The tilde character (~) displays next to the member's name.

Setting UDAs

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

- Calc scripts. After you define a UDA, you can query a member for its UDA in a calc script. For example, you could multiply all members with the UDA Debit by -1 so that they display as either positive or negative (depending on how the data is currently stored). See Chapter 31, "Developing Calc Scripts."
- Data loading. You can change the sign of the data as it is loaded into the database based on its UDA. See "Flipping Field Signs" on page 22-27.

If you want to perform a calculation, selectively retrieve data based on attribute values, or provide full crosstab, pivot, and drill-down support in the spreadsheet, create attribute dimensions instead of UDAs. See "Differences Between Attributes and UDAs" on page 10-9.

Rules for UDAs

Follow these rules when creating UDAs:

- You can define multiple UDAs per member.
- You cannot set the same UDA twice for one member.

- You can set the same UDA for different members.
- A UDA name can be the same as a member, alias, level, or generation name. When you name UDAs, follow the same naming rules as for members. See “Rules for Naming Dimensions and Members” on page 8-15.
- You cannot create a UDA on shared members.
- You cannot create a UDA on members of attribute dimensions.
- A UDA applies to the specified member only. Descendants and ancestors of the member don’t automatically receive the same UDA.

Creating UDAs

➤ To create a UDA in the Outline Editor:

1. Select the member to create the UDA for; for example, East.
2. Click the Data Dictionary button,  , press the Enter key, or select Edit > Properties to open the Member Properties dialog box and click the UDAs tab.

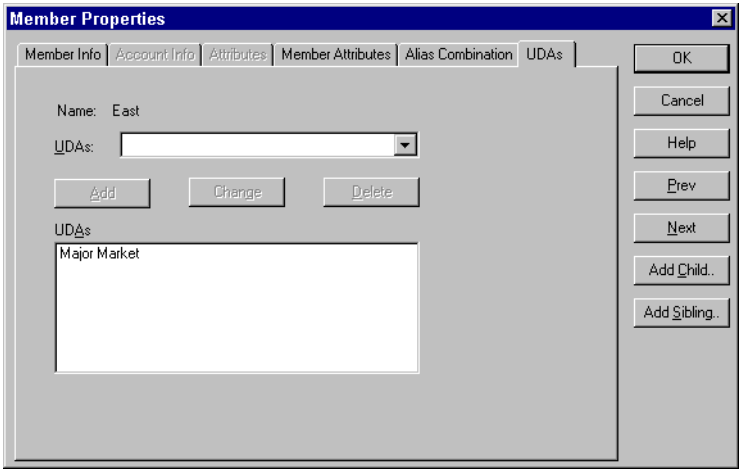


Figure 9-13: UDAs Page

3. Enter the UDA in the UDAs text box or select it from the list box; for example, Major Market.

4. Click the Add button to add it to the UDAs list. The UDAs box lists all UDAs for the selected member.
5. Click OK.

Setting Comments on Dimensions and Members

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

```
/* comment */
```

➤ To add comments to dimensions or members:

1. Select the dimension or member that you'd like to add the comment to; for example, Market.
2. Click the Data Dictionary button,  , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.

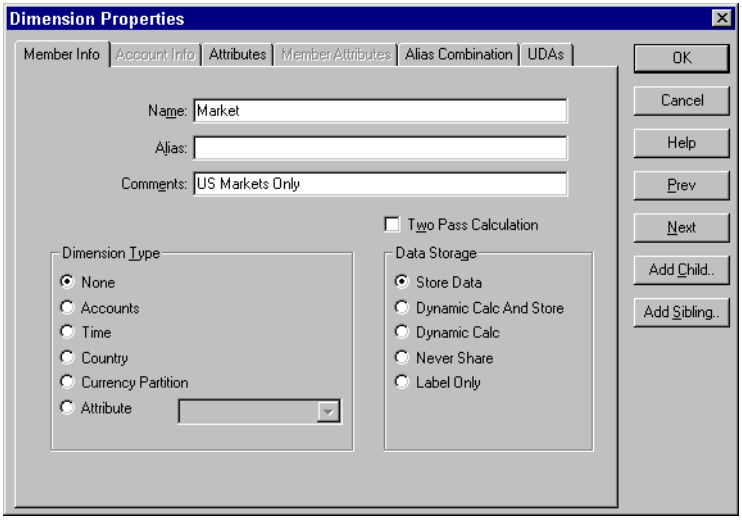


Figure 9-14: Comments in the Dimension Specification Dialog Box

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

Figure 9-15: Sample Dimension Comment

Setting Formulas for Dimensions and Members

You can apply formulas to standard dimensions and members. You cannot set formulas for attribute dimensions and their members. The formula determines how Hyperion Essbase calculates the outline data. For more information about formulas, see Chapter 26, “Developing Formulas.”

➤ To add a formula to a dimension or member:

1. Select the dimension or member to which to add the formula; for example, Variance %.

2. Click the Formula button, =, or press the = sign on the keyboard to open the Formula Editor:

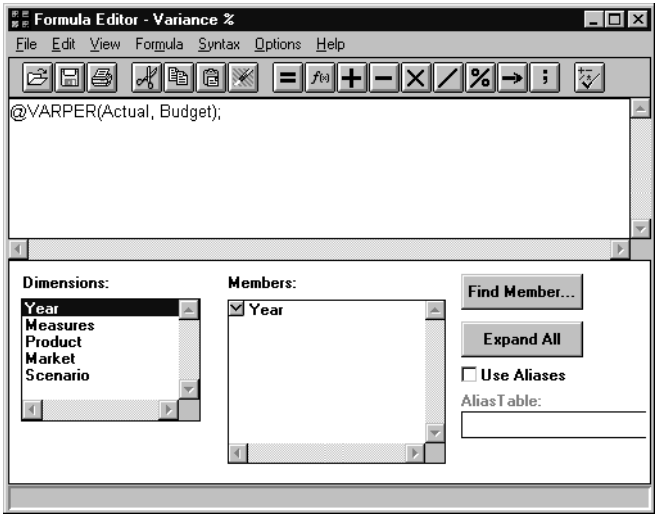


Figure 9-16: Formula Editor

3. Type the formula into the edit field; for example, @VARPER(Actual, Budget).
4. Select File > Close to close the Formula Editor. Hyperion Essbase checks the formula's syntax. If the syntax is correct, the formula displays next to the member's name. If not, open the Formula Editor and check the syntax.

   For more information about using the Formula Editor, see "Creating Formulas" on page 26-8.

Chapter 10

Working with Attributes

Attributes describe characteristics of data such as the size and color of products. Through attributes you can group and analyze members of dimensions based on their characteristics.

This chapter describes how to create and manage attributes in a Hyperion Essbase OLAP Server outline. It contains the following sections:

You can find other information about attributes in relevant sections of this book.

For information about...	See...
Defining attributes through dimension build	• Chapter 13, "Introducing Dynamic Dimension Building" • Chapter 14, "Building Dimensions Using a Rules File"
Using attributes in partitions	• Chapter 6, "Designing Partitioned Applications" • Chapter 16, "Building and Maintaining Partitions"
Using attributes in report writer	• Chapter 36, "Developing Report Scripts" • Chapter 37, "Examples of Report Scripts"

About Attributes

You can use the Hyperion Essbase attribute feature to retrieve and analyze data not only from the perspective of dimensions, but also in terms of characteristics, or attributes, of those dimensions. For example, you can analyze product profitability based on size or packaging, and you can make more effective conclusions by incorporating into your analysis market attributes such as the population size of each market region.

Such an analysis could tell you that decaffeinated drinks sold in cans in small (less than 6,000,000-population) markets are less profitable than you anticipated. For more details, you can filter your analysis by specific attribute criteria, including minimum or maximum sales and profits of different products in similar market segments.

Here are a few ways analysis by attribute provides depth and perspective, supporting better-informed decisions.

- You can select, aggregate, and report on data based on common features (attributes).
- By defining attributes as having a text, numeric, Boolean, or date type, you can filter (select) data using type-related functions such as AND, OR, and NOT operators and <, >, and = comparisons.
- You can use the numeric attribute type to group statistical values by attribute ranges; for example, population groupings such as <500,000, 500,000–1,000,000, and >1,000,000.
- Through the Attribute Calculations dimension automatically created by Hyperion Essbase, you can view sums, counts, minimum or maximum values, and average values of attribute data. For example, when you enter Avg and Bottle into a spreadsheet, Hyperion Essbase retrieves calculated values for average sales in bottles for all the column and row intersections on the sheet.
- You can perform calculations using numeric attribute values in calc scripts and member formulas; for example, to determine profitability by ounce for products sized by the ounce.

- You can create crosstabs of attribute data for the same dimension, and you can pivot and drill down for detail data in spreadsheets.

  An attribute *crosstab* is a report or spreadsheet showing data consolidations across attributes of the same dimension. For example, the crosstab in Figure 10-1 displays product packaging as columns and the product size in ounces as rows. At their intersections, you see the profit for each combination of package type and size. From this information, you can see which size-packaging combinations were most profitable in the Florida market.

```
                Product Year Florida Profit Actual

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

Figure 10-1: Crosstab Example

About Attribute Dimensions

In the Sample Basic database, products have attributes that are characteristics of the products. For example, products have an attribute that describes their packaging. In the outline, you see this as two dimensions, the Products dimension, and the Pkg Type attribute dimension that is associated with it. An *attribute* dimension has the word Attribute next to its name in the outline. Figure 10-2 shows part of the Sample Basic outline featuring the Product dimension and three attribute dimensions, Caffeinated, Ounces, and Pkg Type.

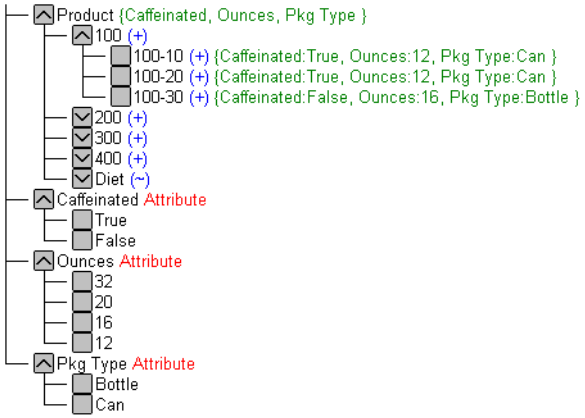


Figure 10-2: Outline Showing Base and Attribute Dimensions

In the outline, to the right of the Product dimension, the terms Caffeinated, Ounces, and Pkg Type, enclosed in braces, show that these attribute dimensions are associated with the Product dimension.

A *standard* dimension is any dimension that is not an attribute dimension. When an attribute dimension is associated with a standard dimension, the standard dimension is the *base* dimension for that attribute dimension. In the outline in Figure 10-2, the Product dimension is the base dimension for the Caffeinated, Ounces, and Pkg Type attribute dimensions.

Note: Attribute dimensions and members are Dynamic Calc. This means Hyperion Essbase calculates attribute information at retrieval time. Attribute data is not stored in the database.

Members of Attribute Dimensions

Members of an attribute dimension are potential attributes of the members of the associated base dimension. After you associate a base dimension with an attribute dimension, you associate members of the base dimension with members of the associated attribute dimension. The Market dimension member Connecticut is associated with the 6000000 member of the Population attribute dimension. That makes 6000000 an attribute of Connecticut.

In the outline, the information enclosed in braces next to a base dimension member shows the attributes of that member. In Figure 10-2, next to product "100-10, {Caffeinated:True, Ounces:12, Pkg Type:Can}" shows that product 100-10 has three attributes: product 100-10 has caffeine, it is sold in 12-ounce containers, and the containers are cans.

There are several important rules regarding members of attribute dimensions and their base dimensions.

- All base dimension members associated with members of a particular attribute dimension must be at the same level.

  For example, in Figure 10-3, all Market dimension members that have Population attributes are at level 0. You cannot associate East, which is a level 1 member, with a Population attribute since the other members of the Market dimension that have Population attributes are level 0 members.

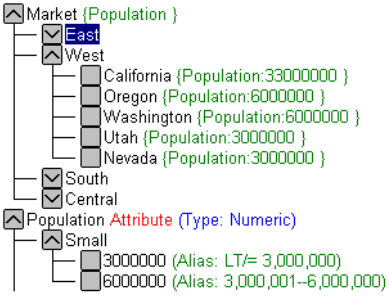


Figure 10-3: Association of Attributes with the Same Level Members of the Market Dimension

- The level 0 members of attribute dimensions are the only members that you can associate with base dimension members.

  For example, in the Population attribute dimension, you can associate only level 0 members such as 3000000, 6000000, and 9000000, with members of the Market dimension. You cannot associate a level 1 member such as Small. The only members of attribute dimensions that have attribute values are level 0 members.

  You can use the higher-level members of attribute dimensions to select and group data. For example, you can use Small, the level 1 member of the Population attribute dimension, to retrieve sales in both the 3000000 and 6000000 population categories.

- A base dimension member can have many attributes, but only one attribute from each particular attribute dimension.

  For example, product 100-10 can have size and packaging attributes, but only one size and only one type of packaging.

Attribute Types

Attribute dimensions have a text, numeric, Boolean, or date type that enables different functions for grouping, selecting, or calculating data. Although assigned at the dimension level, the attribute type applies only to level 0 members of the attribute dimension. The name of the level 0 member of an attribute dimension is the attribute value.

The default attribute type is *text*. Text attributes enable the basic attribute member selection and attribute calculations functionality. This includes AND, OR, NOT, >, <, and = comparisons. When you perform such comparisons, Hyperion Essbase compares characters. For example, the package type Bottle is greater than the package type Can because B precedes C in the alphabet. In Sample Basic, Pkg Type is a text attribute dimension.

The names of level 0 members of *numeric* attribute dimensions are numeric values. You can perform AND, OR, NOT, >, <, and = operations on numeric attributes and you can include the names (values) of numeric attribute dimension members in calculations. For example, you can use the number of ounces specified in the Ounces attribute to calculate profit per ounce for each product.

You can also associate numeric attributes with ranges of base dimension values; for example, to analyze product sales by market population groupings—states with 3,000,000 population or less in one group, states with a population between 3,000,001 and 6 million in another group, and so on. See “Assigning the Names of Members of Numeric Attribute Dimensions to Ranges of Values” on page 10-23.

All *Boolean* attribute dimensions in a database contain only two members. The member names must match the settings for the database; for example, True and False. See “Changing the Member Names for Boolean Attribute Dimensions” on page 10-21.

You can use *date* attributes to specify the date format—month-day-year or day-month-year—and to sequence information accordingly. See “Setting the Member Name Format of Date Attribute Dimensions” on page 10-22. You can perform AND, OR, NOT, >, <, and = operations on date attributes and you can compare dates in a calculation; for example, to select product sales from markets established since 10-12-1999.

Hyperion Essbase supports date attributes from January 1, 1970 through January 1, 2038.

Differences Between Attribute and Standard Dimensions

In general, attribute dimensions and their members are similar to standard dimensions and members. You can provide aliases and member comments for attributes. Attribute dimensions can include hierarchies and you can name generations and levels. You can perform the same spreadsheet operations on attribute dimensions and members as you can on standard dimensions and members; for example, to analyze data from different perspectives, you can retrieve, pivot, and drill down in the spreadsheet.

Table 10-1 describes major differences between attribute and standard dimensions and their members.

Table 10-1: Differences Between Attribute and Standard Dimensions

	Attribute Dimensions	**Standard Dimensions**
Storage	Must be sparse. Their base dimensions must also be sparse.	Can be dense or sparse
Storage property	Dynamic Calc only, therefore not stored in the database. The outline does not display this property.	Can be Store Data, Dynamic Calc And Store, Dynamic Calc, Never Share, or Label Only
Position in outline	Must be the last dimensions in the outline	Must be ahead of all attribute dimensions in the outline
Partitions	Cannot be defined along attribute dimensions, but you can use attributes to define a partition on a base dimension.	Can be defined along standard dimensions.
Formulas (on members)	Cannot be associated	Can be associated
Shared members	Not allowed	Allowed
Alias combinations	Cannot include attribute dimensions or members	Can include standard dimensions and members
Two-pass calculation member property	Not available	Available
UDAs on members	Not allowed	Allowed
Consolidations	For all members, calculated through the Attribute Calculations dimension members: Sum, Count, Min, Max, and Avg.	Consolidation operation indicated by assigning the desired consolidation symbol to each member
Member selection facilitated by Level 0 member typing	Available types include: text, numeric, Boolean, and date.	All members treated as text.

Table 10-1: Differences Between Attribute and Standard Dimensions (Continued)

	Attribute Dimensions	Standard Dimensions
Associations	Must be associated with a base dimension	N/A
Spreadsheet drill-downs	List the base dimension data associated with the selected attribute. For example, drilling down on the attribute Glass displays sales for each product packaged in glass, where Product is the base dimension for the Pkg Type attribute dimension.	List lower or sibling levels of detail in the standard dimensions. For example, drilling down on QTR1 displays a list of products and their sales for that quarter.

Differences Between Attributes and UDAs

Attributes and UDAs both enable analysis based on characteristics of the data. Attributes provide much more capability than UDAs. Table 10-2 compares them. Checkmarks indicate the feature supports the corresponding capability.

Table 10-2: Comparing Attributes and UDAs

Capability	Attributes Feature	UDAs Feature
Data Storage		
You can associate with sparse dimensions.	✓	✓
You can associate with dense dimensions.		✓
Data Retrieval		
You can group and retrieve consolidated totals by attribute or UDA value. For example, associate the value High Focus Item to various members of the Product dimension and use that term to retrieve totals and details for just those members.	✓ Simple	✓ More difficult to implement, requiring additional calc scripts or commands

Table 10-2: Comparing Attributes and UDAs (Continued)

Capability	Attributes Feature	UDAs Feature
Data Retrieval (continued)		
You can categorize attributes in a hierarchy and retrieve consolidated totals by higher levels in the attribute hierarchy; for example, if each product has a specific size attribute such as 8, 12, 16, or 32, and the sizes are categorized as small, medium, and large. You can view the total sales of small products.	✓	✓ More difficult to implement
You can create crosstab views displaying aggregate totals of attributes associated with the same base dimension.	✓ You can show a crosstab of all values of each attribute dimension.	✓ You can only retrieve totals based on specific UDA values.
You can use Boolean operators AND, OR, and NOT with attribute and UDA values to further refine a query. For example, you can select decaffeinated drinks from the 100 product group.	✓	✓
Because attributes have a text, Boolean, date, or numeric type, you can use appropriate operators and functions to work with and display attribute data. For example, you can view sales totals of all products introduced after a specific date.	✓	
You can group numeric attributes into ranges of values and let the dimension building process automatically associate the base member with the appropriate range. For example, you can group sales in various regions based on ranges of their populations: less than 3 million, between 3 and 6 million, and so on.	✓	

Table 10-2: Comparing Attributes and UDAs (Continued)

Capability	Attributes Feature	UDAs Feature
Data Retrieval (continued)		
Through the Attribute Calculations dimension, you can view aggregations of attribute values as sums, counts, minimums, maximums, and averages.	✓	
You can use an attribute in a calculation that defines a member. For example, you can use the weight of a product in ounces to define the profit per ounce member of the Measures dimension.	✓	
You can retrieve specific base members using attribute-related information	✓ Powerful conditional and value-based selections	✓ Limited to text string matches only
Data Conversion		
Based on the value of a UDA, you can change the sign of the data as it is loaded into the database. For example, you can reverse the sign of all members with the UDA Debit.		✓
Calc Scripts		
You can perform calculations on a member if its attribute or UDA value matches a specific value. For example, you can increase the price by 10% of all products with the attribute or UDA of Bottle.	✓	✓
You can perform calculations on base members whose attribute value satisfies conditions that you specify. For example, you can calculate the Profit per Ounce of each base member.	✓	

Introducing the Attribute Calculations Dimension

When you create the first attribute dimension in the outline, Hyperion Essbase also creates the Attribute Calculations dimension comprising five members with the default names Sum, Count, Min (minimum), Max (maximum), and Avg (average). You can use the members of the Attribute Calculations dimension to retrieve calculated information; for example, the average sales of cola within different market population ranges.

The Attribute Calculations dimension is not visible in the outline. You can see it wherever you select dimension members, such as in the Member Selection dialog box and in the Hyperion Essbase Query Designer in the Hyperion Essbase Spreadsheet Add-in.

For more information about the Attribute Calculations dimension, see “Attribute Calculations Dimension” on page 10-31 and “Changing Member Names of the Attribute Calculations Dimension” on page 10-19.

Attribute Design Considerations

Hyperion Essbase provides more than one way to design attribute information into a database. Most often, defining characteristics of the data through attribute dimensions and their members is the best approach.

Using Attribute Dimensions

For the most flexibility and functionality, use attribute dimensions to define attribute data. Using attribute dimensions provides the following features:

- Sophisticated, flexible data retrieval

  You can view attribute data only when you want to, you can create meaningful summaries through crosstabs, and using type-based comparisons, you can selectively view just the data you want to see.

- Additional calculation functionality

  Not only can you perform calculations on the names of members of attribute dimensions to define members of standard dimensions, you can also access five different types of consolidations of attribute data: sums, counts, averages, minimums, and maximums.

- Economy and simplicity

  Because attribute dimensions are sparse, Dynamic Calc, they are not stored as data. Compared to using shared members, outlines using attribute dimensions contain fewer members and are easier to read.

For more about attribute features, see “About Attributes” on page 10-2.

Using Alternative Design Approaches

In some situations, you should consider one of the following approaches:

- UDAs—Although UDAs provide less flexibility than attributes, you can use them to group and retrieve data based on its characteristics. See “Differences Between Attributes and UDAs” on page 10-9.
- Shared members—For example, to include a seasonal analysis in the Year dimension, repeat the months as shared members under the appropriate season; Winter: Jan (shared member), Feb (shared member), and so on. A major disadvantage of using shared members is that the outline becomes very large if the categories repeat a lot of members.
- Standard dimensions and members—Additional standard dimensions provide flexibility, but they add storage requirements and complexity to a database. See “Analyzing Database Design” on page 5-10.

Table 10-3 describes situations where you might consider one of these alternative approaches for managing attribute data in a database.

Table 10-3: Considering Alternatives to Attribute Dimensions

If you want to...	Consider using...
Analyze attributes of dense dimensions	UDAs or shared members.
Perform batch calculation of data	Shared members or members of separate, standard dimensions.
Define the name of a member of an attribute dimension as a value as that results from a formula	Shared members or members of separate, standard dimensions

Table 10-3: Considering Alternatives to Attribute Dimensions (Continued)

If you want to...	Consider using...
Define attributes that vary over time	Members of separate, standard dimensions. For example, to track product maintenance costs over a period of time, the age of the product at the time of maintenance is important. However, using the attribute feature you could associate only one age with the product. You need multiple members in a separate dimension for each time period that you want to track.
Minimize retrieval time with large numbers of base members	Batch calculation with shared members or members of separate, standard dimensions.

Outline Performance Considerations

Outline layout and content can affect attribute calculation and query performance. For general outline design guidelines, see “Designing an Outline to Optimize Performance” on page 5-23.

To optimize attribute query performance, consider the following design tips:

- Ensure that attribute dimensions are the only sparse Dynamic Calc dimensions in the outline.
- Locate sparse dimensions after dense dimensions in the outline. Place the most-queried dimensions at the beginning of the sparse dimensions and attribute dimensions at the end of the outline. In most situations, the base dimensions are the most queried dimensions.

For information on optimizing calculation of outlines containing attributes, see “Calculation and Retrieval Performance Considerations” on page 10-35.

Working with the Names of Members of Attribute Dimensions

All member names in an outline must be unique. When you use the attribute feature, Hyperion Essbase establishes some default member names in some places. These default names might duplicate names that already exist in the outline. Use the Settings > Attribute Member Names command to change these system-defined names for the database. You can also use this command to establish other settings for members of attribute dimensions in the database.

Define the member name settings before you define or build the attribute dimensions. Changing the settings after the attribute dimensions and members are defined could result in invalid member names.

The following sections describe how to work with the names of members of attribute dimensions:

- "Defining a Prefix or Suffix Format for the Names of Members of Attribute Dimensions" on page 10-16
- "Changing Member Names of the Attribute Calculations Dimension" on page 10-19
- "Changing the Member Names for Boolean Attribute Dimensions" on page 10-21
- "Setting the Member Name Format of Date Attribute Dimensions" on page 10-22
- "Assigning the Names of Members of Numeric Attribute Dimensions to Ranges of Values" on page 10-23

Note: If you partition on outlines containing attribute dimensions, the name format settings of members described in this section must be identical in source and target outlines.

You can use the GETATTRIBUTESPECS command in ESSCMD to view the attribute member name format specifications for the database. For example, :

```
GETATTRIBUTESPECS Sample Basic
```

See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Defining a Prefix or Suffix Format for the Names of Members of Attribute Dimensions

The names of members of Boolean, date, and numeric attribute dimensions are values. It is possible to encounter duplicate attribute values in different attribute dimensions.

- Boolean example—If you have more than one Boolean attribute dimension in an outline, the two members of each of those dimensions have the same names, by default, True and False.
- Date example—If you have more than one date attribute dimension, some member names in both dimensions could be the same. For example, the date that a store opens in a certain market could be the same as the date a product was introduced.
- Numeric example—12 can be the attribute value for the size of a product and 12 could also be the value for the number of packing units for a product. This would result in two members with the same name: 12.

Because Hyperion Essbase does not allow duplicate member names, Application Manager helps you to define unique names by attaching a prefix or suffix to member names in Boolean, date, and numeric attribute dimensions in the outline. For example, by setting member names of attribute dimensions to include the

dimension name as the suffix, attached by an underscore, the member value 12 in the Ounces attribute dimension assumes the unique, full attribute member name, 12_Ounces.

By default, Hyperion Essbase assumes that no prefix or suffix is attached to the names of members of attribute dimensions.

Note: The convention that you select applies to the level 0 member names of all numeric, Boolean, and date attribute dimensions in the outline.

For the examples in the following instructions, consider the attribute dimension shown in Figure 10-4.

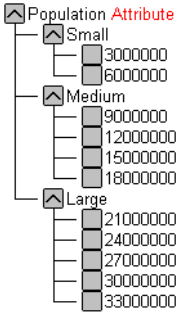


Figure 10-4: The Population Attribute Dimension and Members

➤ To define prefixes or suffixes for the names of members of Boolean, date, and numeric attribute dimensions:

1. Select Settings > Attribute Member Names.

   The Sample box shows the format of member names in numeric, Boolean, and date attribute dimensions.

2. In the Value option group, select the source of the value that Hyperion Essbase attaches to the attribute member name in the outline.

   - **None** uses the member name in the outline as the full name. Nothing is attached; for example, 3000000.
   - **Parent** attaches the immediate parent; for example, Small_3000000.
   - **Grandparent and Parent** attaches the parent, preceded by the parent's parent; for example, Population_Small_3000000.

- **Ancestor** attaches all generations; for example, Population_Small_3000000.
- **Dimension** attaches the attribute dimension name; for example, Population_3000000.

Note: If you select a value other than None, Hyperion Essbase assumes the other format options are Prefix and "_(underscore)."

3. In the Separator option group, select the separator character to insert between the attribute member name in the outline and the attached value.
   - **_(underscore)** inserts an underscore between each value in the prefix or suffix; for example, Population_Small_3000000.
   - **| (pipe)** inserts a pipe between each value in the prefix or suffix; for example, Population|Small|3000000.
   - **^ (caret)** inserts a caret between each value in the prefix or suffix; for example, Population^Small^3000000.
4. In the Prefix/Suffix option group, select whether to attach a prefix or a suffix to the attribute member name in the outline.
   - **Prefix** attaches the value selected in the Value option group before the attribute member name; for example, Population_3000000.
   - **Suffix** attaches the value selected in the Value option group after the attribute member name; for example, 3000000_Population.
5. Click OK.

Note: The outline does not show the full attribute names. You can see and use the full attribute names anywhere you select members, such as when you define partitions or select information to be retrieved.

You can use the GETATTRINFO command in ESSCMD to view the dimension, attribute value and attribute type of a specific attribute member. For example, to view attribute name information for the Caffeinated_True attribute:

```
GETATTRINFO "Caffeinated_True"
```

See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Changing Member Names of the Attribute Calculations Dimension

To avoid duplicating names in an outline, you may need to change the name of the Attribute Calculations dimension or its members. For more information about the Attribute Calculations dimension, see “Attribute Calculations Dimension” on page 10-31.

Regardless of the name that you use for a member, its function remains the same. For example, the second (Count) member always counts, no matter what you name it.

➤ To change the names of the Attribute Calculations dimension and its members:

1. Select Settings > Attribute Member Names and select the Attribute Calculations tab.

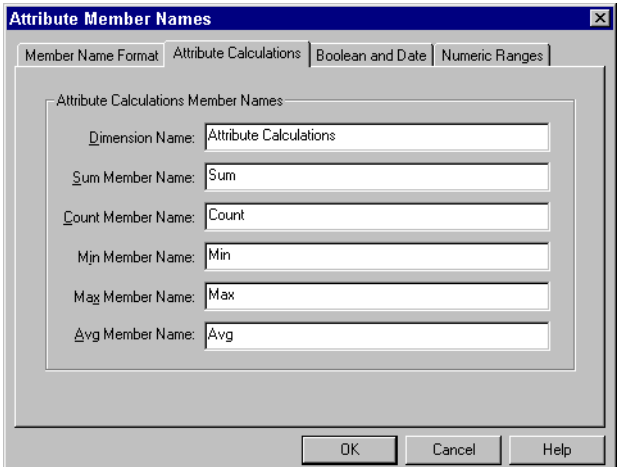


Figure 10-5: Attribute Calculations Page in the Attribute Member Names Dialog Box

2. In the Attribute Calculations Member Names option group, type the new member name for each member name that you want to change. Follow the member naming rules described in “Rules for Naming Dimensions and Members” on page 8-15.

   The default names are:

Text Box Label	Default Name
Dimension Name	Attribute Calculations
Sum Member Name	Sum
Count Member Name	Count
Min Member Name	Min
Max Member Name	Max
Avg Member Name	Avg

   Note: The Sum member totals members based on their consolidation property or formula. For example, the Sum member uses the following formula to consolidate the profit percentages of 12-ounce products:

$$\text{Sum of Profit\% of 12-ounce products} = \frac{\text{Sum of Profit of base members with the Ounces attribute 12}}{\text{Sum of Sales of base members with the Ounces attribute 12}} * 100$$

   This calculation is not the sum of all percentages for all base members with the Ounces attribute 12. For more information about each member, see “Default Attribute Calculations Members” on page 10-32.

3. Click OK.

Changing the Member Names for Boolean Attribute Dimensions

All Boolean attribute dimensions in a single database have two, level 0 members with the same names, by default, True and False.

Hyperion Essbase enables you to change the names of members of Boolean attribute dimensions; for example, to Yes and No.

Note: Changing the Boolean member-name setting does not retroactively change the names of members of existing Boolean attribute dimensions. You must manually change the names of existing Boolean members to the names specified in the Boolean member names setting.

When you set an attribute dimension type as Boolean, Hyperion Essbase automatically creates two members with the names specified in the setting. If other members exist in the Boolean attribute dimension, you must remove them.

➤ To change the member names setting of Boolean attribute dimensions in a database:

1. Select Settings > Attribute Member Names and select the Boolean and Date tab.

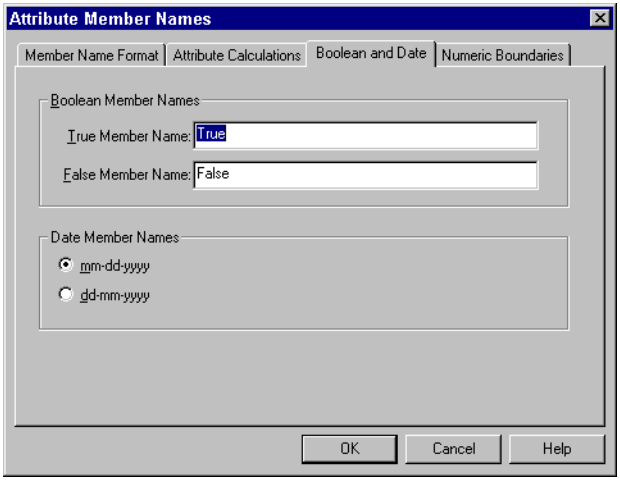


Figure 10-6: Boolean and Date Page in the Attribute Member Names Dialog Box

2. In the Boolean Member Names option group, type the new member name for each member name that you want to change. Follow the member naming rules described in “Rules for Naming Dimensions and Members” on page 8-15.

   The default names are:

Text Box Label	Default Name
True Member Name	True
False Member Name	False

3. Click OK.

   Note: If you have more than one Boolean attribute dimension, you must specify a prefix or suffix member name format to ensure unique member names; for example, Caffeinated_True and Caffeinated_False. See “Defining a Prefix or Suffix Format for the Names of Members of Attribute Dimensions” on page 10-16.

Setting the Member Name Format of Date Attribute Dimensions

➤ To change the format of members of date attribute dimensions at the database level:

1. Select Settings > Attribute Member Names and click the Boolean and Date tab.
2. In the Date Member Names group, select a format:
   - mm-dd-yyyy displays the month before the day; for example, October 18, 1999 is displayed as 10-18-1999.
   - dd-mm-yyyy displays the day before the month; for example, October 18, 1999 is displayed as 18-10-1999.

3. Click OK.

Note: If you change the date member name format, the names of existing members of date attribute dimensions may be invalid. For example, if the 10-18-1999 member exists and you change the format to dd-mm-yyyy, outline verification will find this member invalid. If you change the date format, you must rebuild the date attribute dimensions.

Assigning the Names of Members of Numeric Attribute Dimensions to Ranges of Values

Members of numeric attribute dimensions can represent single numeric values. For example, the member 12 in the Ounces attribute dimension represents the single numeric value 12; you associate this attribute with all 12-ounce products. The outline includes a separate member for each size; for example, 16, 20, and 32.

Members of numeric attribute dimensions can represent ranges of values. Consider, for example, the Population attribute dimension shown in Figure 10-7.

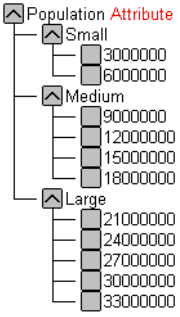


Figure 10-7: Population Attribute Dimension and Members

In this outline, the members of the Population attribute dimension represent ranges of population values in the associated Market dimension. The 3000000 member represents populations from zero through 3,000,000; the 6000000 member represents populations from 3,000,001 through 6,000,000; and so on. Each range includes values greater than the name of the preceding member up to and including the member value itself. A setting for the outline establishes that each numeric member represents the top of its range.

You can also define this outline setting so that members of numeric attribute dimensions are the bottoms of the ranges that they represent. For example, if numeric members are set to define the bottoms of the ranges, the 3000000 member represents populations from 3,000,000 through 5,999,999 and the 6000000 member represents populations from 6,000,000 through 8,999,999.

When you build the base dimension, Hyperion Essbase automatically associates members of the base dimension with the appropriate attribute range. For example, if numeric members represent the tops of ranges, Hyperion Essbase automatically associates the Connecticut market, with a population of 3,269,858, with the 6000000 member of the Population attribute dimension.

In the dimension build rules file, you specify the size of the range for each member of the numeric attribute dimension. In the above example, each attribute represents a range of 3,000,000. See “Working With Numeric Ranges” on page 13-25.

➤ To define the rule for assigning numeric attribute member names to ranges of values:

1. Select Settings > Attribute Member Names and select the Numeric Ranges tab.

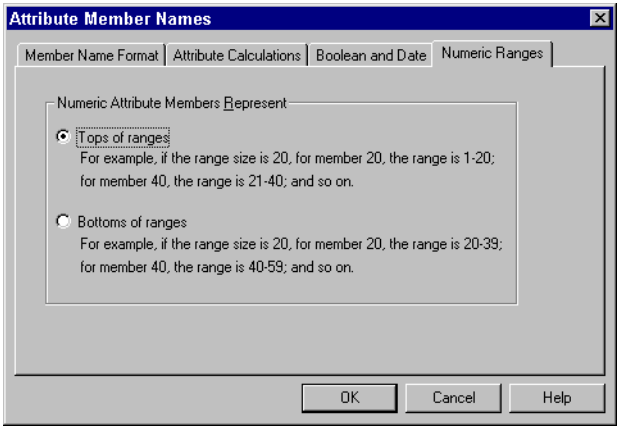


Figure 10-8: Numeric Ranges Page in the Attribute Member Names Dialog Box

2. Select the option that sets whether numeric attribute values define the tops or bottoms of the ranges that they represent.

   - Tops of ranges. For example, Hyperion Essbase associates the 6000000 attribute with all markets in a population area greater than the value of the next lower member of the attribute dimension and less than or equal to 6,000,000. This is the default setting.
   - Bottoms of ranges. For example, Hyperion Essbase associates the 6000000 attribute with all markets in a population area equal to or greater than 6000000 and less than the value of the next higher member of the attribute dimension.

3. Click OK.

Defining Attributes Manually

When manually working with attributes, use the Outline Editor in Hyperion Essbase Application Manager to perform the following dimension and member-related tasks:

- Create the attribute dimensions. See “Adding Dimensions to Outlines” on page 8-18. In the outline, position the attribute dimensions after all standard dimensions.
- Tag the dimensions as attribute dimensions. See “Tagging an Attribute Dimension” on page 9-13.
- Set each attribute dimension type as text, numeric, Boolean, or date. See “Setting the Attribute Type” on page 9-14.
- Add members to the attribute dimensions. See “Adding Members to Dimensions” on page 8-19.
- Associate base dimensions with the attribute dimensions. See “Associating Base Dimensions with Attribute Dimensions” on page 10-26.
- Associate base dimension members with members of the attribute dimensions. See “Associating Base Dimension Members with Attributes” on page 10-28.

Associating Base Dimensions with Attribute Dimensions

When you associate an attribute dimension with a standard dimension, the standard dimension is known as the *base* dimension for that attribute dimension.

- You can only associate attribute dimensions with sparse standard dimensions.
- A standard dimension can be a base dimension for more than one attribute dimension.
- An attribute dimension can be associated with only one base dimension.

  For example, you might have a Size attribute dimension with members Small, Medium, and Large. If you associate the Size attribute dimension with the Product dimension, you cannot also associate the Size attribute dimension with the Market dimension. If you also want to track size-related information for the Market dimension, you must create another attribute dimension with a different name, for example, MarketSize, and associate the MarketSize attribute dimension with the Market dimension.

➤ To associate attribute dimensions with a standard dimension:

1. Select the base dimension, for example Product.
2. Click the Data Dictionary button,  , press the Enter key, or select Edit > Properties to open the Dimension Properties dialog box.

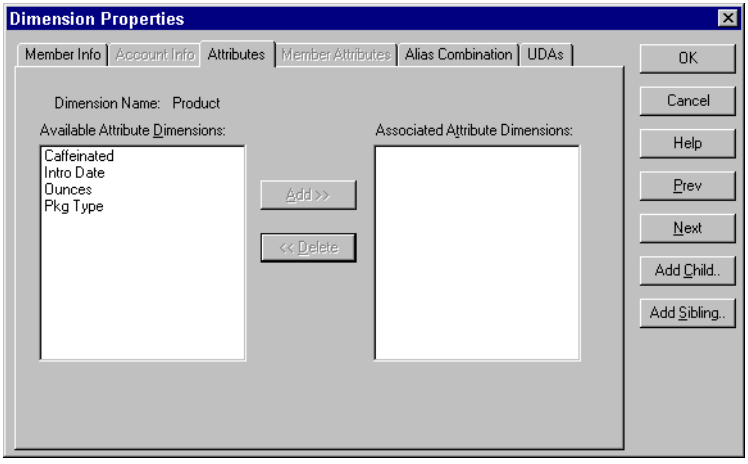


Figure 10-9: Attributes Page in the Dimension Properties Dialog Box

The Attributes page displays the name of the selected base dimension and contains the following items:

- Available Attribute Dimensions list box, which displays the names of attribute dimensions that are not associated with a base dimension
- Associated Attribute Dimensions list box, which displays the names of the attribute dimensions associated with the selected base dimension
- Add button, which moves attribute dimensions from the Available Attribute Dimensions list box to the Associated Attribute Dimensions list box
- Delete button, which removes attribute dimensions from the Associated Attribute Dimensions list box, putting them back in the Available Attribute Dimensions list box

3. Use the Add and Delete buttons to move selected attribute dimension names in and out of the Associated Attribute Dimensions list box.
4. When the Associated Attribute Dimensions list box contains the names of all attribute dimensions to be associated with the base dimension, click OK.

Associating Base Dimension Members with Attributes

Attribute associations must follow these rules:

- You cannot associate multiple members from the same attribute dimension with the same base dimension member. For example, the same product cannot be associated with both bottle and can package types.
- You can associate only level-0 members of attribute dimensions.
- The base dimension members to which you associate members from a specific attribute dimension must all be at the same level.

You can associate members from different attribute dimensions with the same member of a base dimension. For example, a decaffeinated cola product (100-30) sold in 16 ounce bottles has three attributes: Caffeinated:False; Ounces:16; and Pkg Type:Bottle.

After attributes are associated with base dimension members, if you cut or copy and paste base dimension members to another location in the outline, the attribute associations are lost.

➤ To associate a member of a base dimension with members of attribute dimensions:

1. Select the base dimension member with which you want to associate the attributes, for example 100-10.
2. Click the Data Dictionary button,  , press the Enter key, or select Edit > Properties to open the Member Properties dialog box.

3. Select the Member Attributes tab.

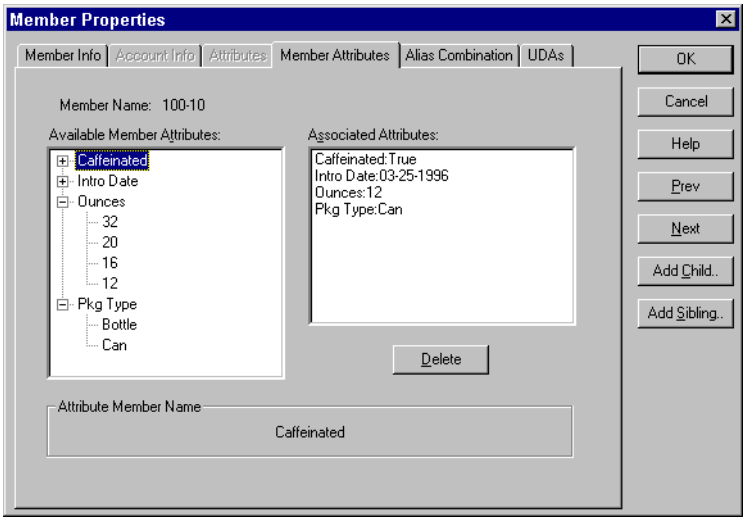


Figure 10-10: Member Attributes Page in the Member Properties Dialog Box

The Member Attributes page displays the name of the selected member of the base dimension and contains the following items:

- Available Member Attributes list box, which lists attribute dimensions associated with the base dimension to which the selected member belongs
- Associated Attributes list box, which lists attributes associated with the selected member of the base dimension
- Delete button, which deletes attribute associations from the Associated Attributes list box
- Attribute Member Name box, which displays the full member name of the selected attribute dimension member

4. Click the plus symbol ⊞ by the name of an attribute dimension or member to display its children.

   Repeat until member names display without plus or minus ⊟ symbols next to them. Attribute values are the level 0 members in an attribute dimension.

5. Double-click the attribute value that you want to associate with the selected base member.

   The Associated Attributes list box displays the name of the selected attribute dimension and the associated attribute value.

6. Repeat for other attribute dimensions, until the Associated Attributes list box lists the correct attribute values to be associated with the base dimension member.

   Note: Hyperion Essbase does not require that each member of a base dimension be associated with a member of an attribute dimension.

7. Click OK.

Calculating Attribute Data

Hyperion Essbase calculates attribute data dynamically at retrieval time, using members from a system-defined dimension created specifically by Hyperion Essbase. Using this dimension, you can apply different calculation functions, such as a sum or an average, to the same attribute. You can also perform specific calculations on members of attribute dimensions; for example, to determine profitability by ounce for products sized by the ounce.

The following information assumes that you understand the concepts of attribute dimensions and Hyperion Essbase calculations, including dynamic calculations. For more information, see Part V, “Calculating Data.”

This section includes the following:

Attribute Calculations Dimension

The Attribute Calculations dimension contains five members. You can use these members in spreadsheets or in reports to dynamically calculate and report on attribute data, such as the average yearly sales of 12-ounce bottles of cola in the West. This dimension has the following properties:

- System-defined. When you create the first attribute dimension in an application, Hyperion Essbase creates the Attribute Calculations dimension and its members (Sum, Count, Min, Max, and Avg). Each member represents a type of calculation to be performed for attributes. For more information, see "Default Attribute Calculations Members" on page 10-32.
- Label only. Like all label only dimensions, the Attribute Calculations dimension shares the value of its first child, Sum. For more information on the label only dimension property, see "Defining Dimension and Member Properties" on page 5-19.
- Dynamic Calc. The data in the Attribute Calculations dimension is calculated when a user requests it and is then discarded. You cannot store calculated attribute data in a database. For more information on dynamic calculations, see Chapter 29, "Dynamically Calculating Data Values."
- Not displayed in Outline Editor. The Attribute Calculations dimension is not displayed in the Outline Editor. Members from this dimension can be viewed in spreadsheets and in reports.

There is no consolidation along attribute dimensions. You cannot tag members from attribute dimensions with consolidation symbols (for example, + or -) or with member formulas in order to calculate attribute data. As Dynamic Calc members, attribute calculations do not affect the batch calculation in terms of time or calculation order. To calculate attribute data at retrieval time, Hyperion Essbase:

1. Finds the base members that are associated with the specified attribute members present in the current query
2. Dynamically calculates the sum, count, minimum, maximum, or average for the attribute-member combination for the current query
3. Displays the results in the spreadsheet or report

4. Discards the calculated values—that is, the values are not stored in the database

   Note: Hyperion Essbase excludes `#MISSING` values when calculating attribute data.

For example, as shown in Figure 10-11, a spreadsheet user specifies two attribute members (Ounces_16 and Bottle) and an Attribute Calculations member (Avg) in a spreadsheet report. Upon retrieval, Hyperion Essbase dynamically calculates the average sales values of all products associated with these attributes for the current member combination (Actual->Sales->East->Qtr1):

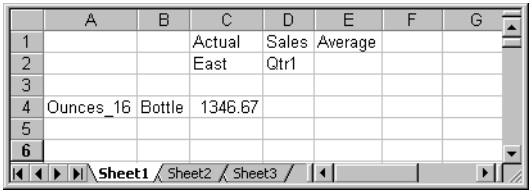


Figure 10-11: Retrieving an Attribute Calculations Member

For more information on accessing calculated attribute data, see “Accessing Attribute Calculations Members Using the Spreadsheet” on page 10-35.

Default Attribute Calculations Members

The Attribute Calculations dimension contains five members used to calculate and report attribute data. These members are:

- Sum: calculates a sum, or total, of the values for a member with an attribute or combination of attributes. Supports non-additive consolidations such as multiplication, and two-pass measures.
- Count: calculates the number of members with the specified attribute or combination of attributes, for which a data value exists. Count includes only those members that have data blocks in existence. To calculate a count of all members with certain attributes, regardless of whether or not they have data values, use the @COUNT function in combination with the @ATTRIBUTE function. For more information, see the online *Technical Reference* in the `DOCS` directory.

- Avg: calculates a mathematical mean, or average, of the non-missing values for an specified attribute or combination of attributes (Sum divided by Count).
- Min: calculates the minimum data value for a specified attribute or combination of attributes.
- Max: calculates the maximum data value for a specified attribute or combination of attributes.

Note: Each of these calculations excludes `#MISSING` values.

You can change these default member names using Application Manager, subject to the same naming conventions as standard members. For more information on changing Attribute Calculations member names, see "Changing Member Names of the Attribute Calculations Dimension" on page 10-19.

The default calculation for attributes is Sum. If a spreadsheet user specifies a member of an attribute dimension in a spreadsheet but does not specify a member of the Attribute Calculations dimension, Hyperion Essbase retrieves the sum for the specified attribute or combination of attributes. For example, in the following spreadsheet view, the value in cell C4 represents the sum of sales values for the attributes Ounces_16 and Bottle for the current member combination (Actual->Sales->East->Qtr1), even though the Sum member is not displayed in the sheet.

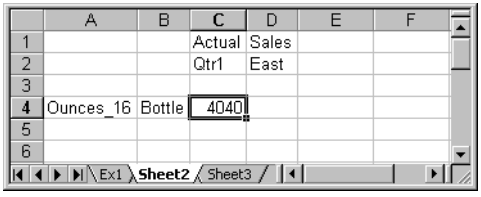


Figure 10-12: Retrieving the Default Attribute Calculations Member

An Attribute Calculation Example

As an example of how Hyperion Essbase calculates attribute data, consider the following yearly sales data for the East:

Table 10-4: Sample Attribute Data

Base Member	Associated Attributes	Sales Value for Attribute-Member Combination
Cola	Ounces_12, Can	23205
Diet Cola	Ounces_12, Can	3068
Diet Cream	Ounces_12, Can	1074
Grape	Ounces_32, Bottle	6398
Orange	Ounces_32, Bottle	3183
Strawberry	Ounces_32, Bottle	5664

A spreadsheet report showing calculated attribute data might look like this:

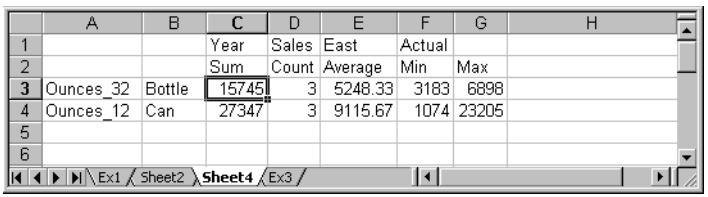


Figure 10-13: Sample Spreadsheet with Attribute Data

As shown in the figure above, you can retrieve multiple Attribute Calculations members for attributes. For example, you can calculate Sum, Count, Avg, Min, and Max for 32-ounce bottles and cans.

Accessing Attribute Calculations Members Using the Spreadsheet

You can access members from the Attribute Calculations dimension in Hyperion Essbase Spreadsheet Add-in. From the spreadsheet, users can view Attribute Calculations dimension members by:

- Entering members directly into a sheet
- Selecting members from the Hyperion Essbase Query Designer
- Selecting members from the Hyperion Essbase Member Selection dialog box
- Entering members as an EssCell parameter

For more information on accessing calculated attribute data from the spreadsheet, see the *Hyperion Essbase Spreadsheet Add-in User's Guide*.

Calculation and Retrieval Performance Considerations

Keep in mind the following calculation and retrieval performance considerations:

- The calculation order for attribute calculations is the same as the order for dynamic calculations. For more information, see "Calculation Order for Dynamic Calculations" on page 29-16.
- Since Hyperion Essbase calculates attribute data dynamically at retrieval time, attribute calculations do not affect the performance of the overall (batch) database calculation.
- Tagging base dimension members as Dynamic Calc may increase retrieval time.
- When a query includes the Sum member and an attribute member whose associated base member is tagged as two-pass, retrieval time may be slow.

- To maximize attribute retrieval performance:
  - Configure the outline as described in “Outline Performance Considerations” on page 10-14.
  - Drill down to the lowest level of base dimensions before retrieving data. For example, in Hyperion Essbase Spreadsheet Add-in, turn on the Navigate Without Data feature, drill down to the lowest level of the base dimensions included in the report, and then retrieve data.
  - When the members of a base dimension are associated with several attribute dimensions, consider grouping the members of the base dimension according to their attributes. For example, in the Sample Basic database, you could group all 8-ounce products. Grouping members by attribute may decrease retrieval time.

Using Attributes in Calculation Formulas

In addition to using the Attribute Calculations dimension to calculate attribute data, you can also use calculation formulas on members of standard or base dimensions to perform specific calculations on members of attribute dimensions; for example, to determine profitability by ounce for products sized by the ounce.

Note: You cannot associate formulas with members of attribute dimensions.

You can use the following functions to perform specific calculations on attributes:

To do this...	Use this function...
Generate a list of all base members with a specific attribute. For example, you can generate a list of members that have the Bottle attribute, and then increase the price for those members.	@ATTRIBUTE
Return the numeric value of the level 0 attribute member from a numeric, Boolean, or date attribute dimension that is associated with the base member being calculated. For example, you can return the numeric value of a size attribute (for example, 12 for the member 12 under Ounces) for the base member being calculated (for example, Cola).	@ATTRIBUTEVAL
Convert a date string to numbers for a calculation. For example, you can use @TODATE in combination with the @ATTRIBUTEVAL function to increase overhead costs for stores opened after a certain date.	@TODATE
Generate a list of all base dimension members associated with attributes that satisfy the conditions that you specify. For example, you can generate a list of products that are greater than or equal to 20 ounces, and then increase the price for those products.	@WITHATTR

Note: For syntax information and examples for these functions, see the online *Technical Reference* in the `DOCS` directory. For an additional example using @ATTRIBUTEVAL in a formula, see "Calculating an Attribute Formula" on page 27-6.

Chapter 11

Creating and Managing Aliases

An alias is an alternate name for a member or shared member. This chapter describes how to create and manage alias tables in an 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 “Rules for Naming Dimensions and Members” on page 8-15.
- You can select different aliases based on the combination of other dimension members. An alias based on a combination of members is called an *alias combination*. Table 11-1 shows members with a different alias for each market in which they are sold:

Table 11-1: 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 alias combinations, see “Creating Aliases for Member Combinations” on page 11-5.

- 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 11-2 shows some products in the Sample Basic database that have two descriptive alias names:

Table 11-2: 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 11-7.

Introducing Alias Tables

Aliases are stored in one or more tables as part of a database outline. When you create a database outline, Hyperion 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 11-2, 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 11-7 and "Setting Alias Tables" on page 11-7.

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 11-1 shows a sample alias table:

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

Figure 11-1: Sample Alias Table

Use the DISPLAYALIAS command in ESSCMD to view the all aliases in an alias table. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, "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 11-1.

➤ 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 select Edit > Properties to open the Member Properties dialog box.

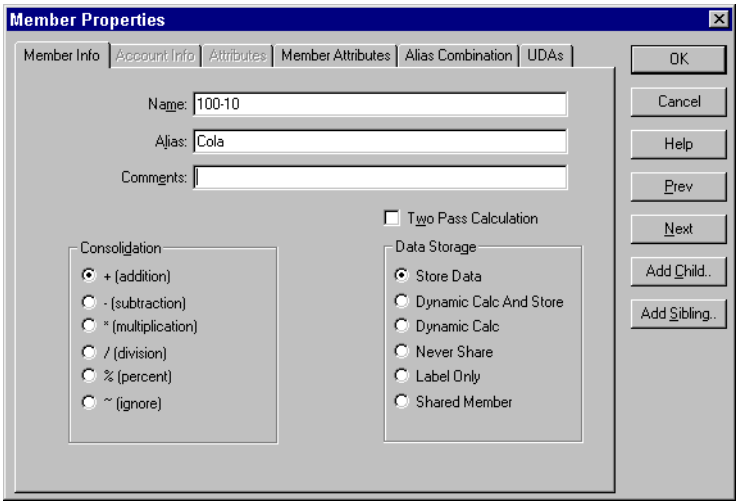


Figure 11-2: Member Properties Dialog Box

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 80 characters long.

Creating Aliases for Member Combinations

You can create aliases for non-attribute members that are based on member combinations. For more information on aliases, see “Introducing Aliases and Alias Tables” on page 11-1.

➤ To set an alias based on a member combination:

1. In Outline Editor, select the member for which the alias is to be defined; for example, 100-10.
2. Click the Data Dictionary button,  , press the Enter key or select Edit > Properties to open the Member Properties dialog box.
3. Click the Alias Combinations tab.

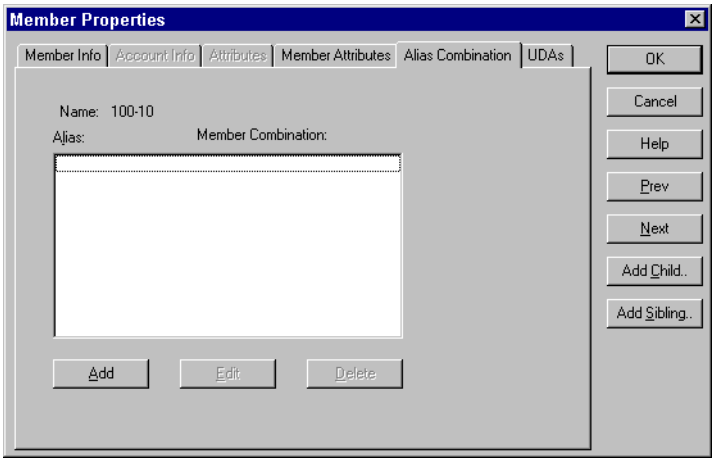


Figure 11-3: Alias Combinations Page

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:

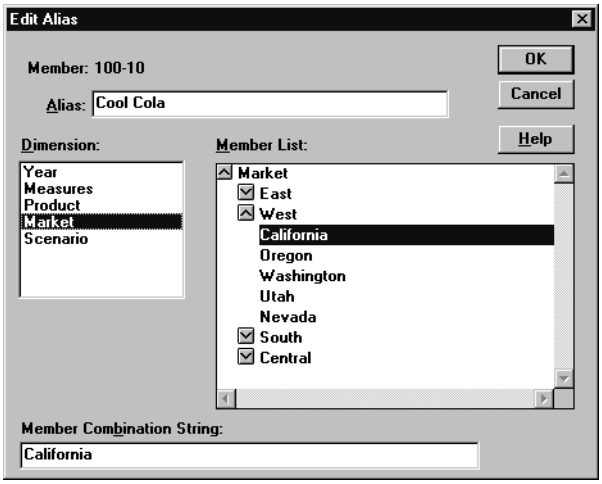


Figure 11-4: Edit Alias Dialog Box

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 select 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. Select Outline > Aliases > Create Table to open the Create Alias Table dialog box.

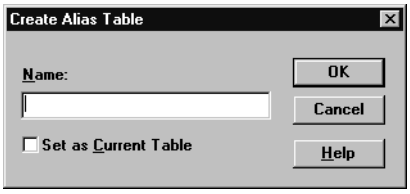


Figure 11-5: Create Alias Table Dialog Box

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 selecting Set as Current Table.
4. Click OK.

Setting Alias Tables

Hyperion 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 11-3.

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

1. Select Outline > Aliases > Set Table to open the Set Current Alias Table dialog box.

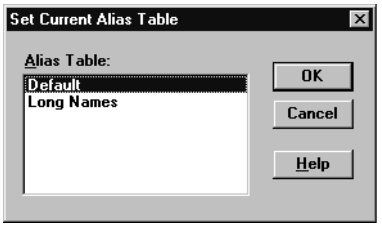


Figure 11-6: Set Current Alias Table Dialog Box

2. Select the alias table from the Alias Table list box. If the alias table that you want to use is not in the list box, you must create it. See "Creating New Alias Tables" on page 11-7.
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 the `DOCS` directory. See Chapter 44, "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. Select Outline > Aliases > Copy Table to open the Copy Alias Table dialog box.

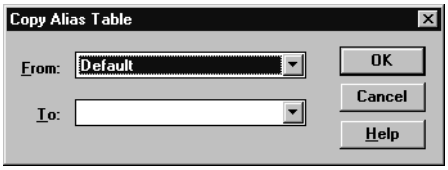


Figure 11-7: Copy Alias Table Dialog Box

2. Select the alias table to copy from the From list box.
3. Enter the new name of the alias table to copy it to in the To list box.
4. Click OK.

Renaming Alias Tables

➤ To change the names of existing alias tables:

1. Select Outline > Aliases > Rename Table to open the Rename Alias Table dialog box.

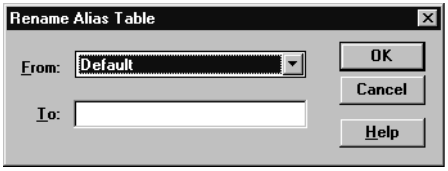


Figure 11-8: Rename Alias Table Dialog Box

2. Select the alias table to rename from the From list box.
3. Enter the new name of the alias table in the To list box.
4. Click OK.

Deleting and Clearing Alias Tables

You can delete an alias table from the outline or you can clear all of the aliases from an alias table without deleting the alias table itself.

Deleting Alias Tables

➤ To delete an alias table from the outline:

1. Select Outline > Aliases > Delete Table to open the Delete Alias Table dialog box.

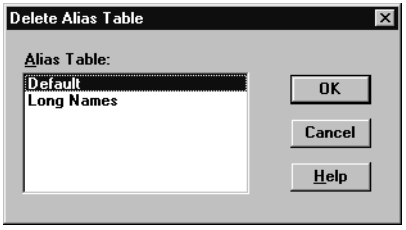


Figure 11-9: Delete Alias Table Dialog Box

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. Select Outline > Aliases > Clear Table to open the Clear Alias Table dialog box.

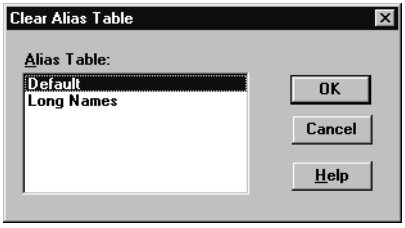


Figure 11-10: Clear Alias Table Dialog Box

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 11-1.

Importing an Alias Table

➤ To import an alias table:

1. Select File > Import > Alias Table to open the Import Server Alias Table Object dialog box.

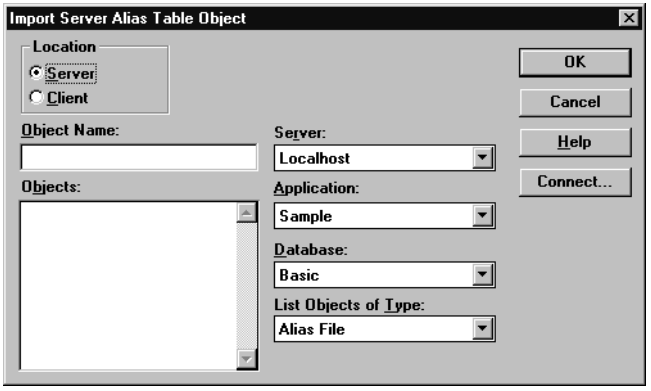


Figure 11-11: Import Alias Table Object Dialog Box

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.

   Select 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 the `DOCS` directory for information about these commands. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Exporting an Alias Table

To export an alias table:

1. Select File > Export > Alias Table to open the Export Server Alias Table Object dialog box.
2. Save the alias table to the desired location.

Use the LISTLINKEDOBJECTS and PURGELINKEDOBJECTS commands in ESSCMD to perform these tasks. See the online *Technical Reference* in the `DOCS` directory for information about these commands. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Chapter 12

Linking Objects to Hyperion 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?

Linked reporting objects are objects that you associate with specific data cells in a Hyperion 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 Hyperion Essbase database.

Hyperion Essbase supports the following types of linked objects:

Table 12-1: 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, an audio clip, or an HTML file (for example, `mypage.htm`).

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

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

How Do Linked Reporting Objects Work with Hyperion 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, Hyperion 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. Hyperion 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 Hyperion Essbase server.

If the object is a URL, Hyperion Essbase stores the URL string as part of the object description in the catalog entry. Hyperion Essbase does not check the syntax of the URL until the user tries to view it. At that time, Hyperion 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."

Note: Hyperion 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 Hyperion Essbase 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 12-2: 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 Hyperion Essbase stores linked files in a repository on the server, you might want to limit the size of files that users can link. This would prevent a user from taking up too much of the server's resources by storing extremely large objects. You can set the maximum linked file size for each application. If a user attempts to link a file that is larger than the limit, an error message displays.

Note: The maximum file size setting applies only to linked files and does not affect cell notes or URLs. The maximum cell note length is fixed at 599 characters. The maximum URL string length is fixed at 512 characters.

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

1. In Hyperion Essbase Application Manager, connect to the appropriate server and select the name of the application.
2. Select Application > Settings. The following dialog box displays:

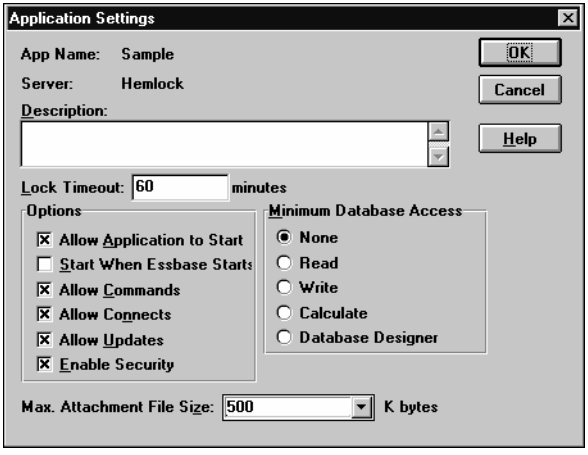


Figure 12-1: Application Settings Dialog Box

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 Hyperion Essbase 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. Select Database > Linked Reporting Objects. The following dialog box displays:

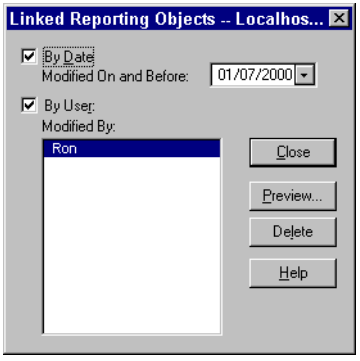


Figure 12-2: Linked Reporting Objects Dialog Box

3. Specify your selection criteria as follows:

   - To select by modification date, select By Date and enter a modification date. Objects modified on or before the date are selected.
   - To select by user name, select By User and select a user name. Objects last modified by the user are selected.

     If you select *both* options, objects matching both criteria are selected. If you select neither By Date nor By User, all objects in the database are selected.

4. To delete all objects that meet your criteria, click Delete. To see a list of the objects that meet the criteria, click Preview. If you click Preview, Hyperion Essbase displays the Linked Objects Browser:

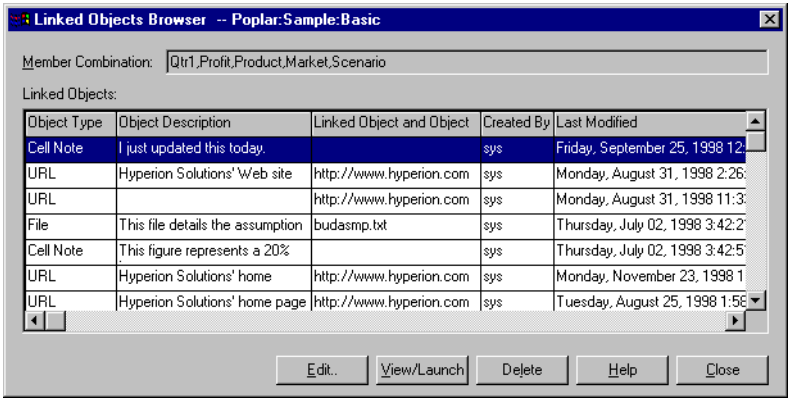


Figure 12-3: Linked Objects Browser

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 displays for your reference. The actions you can perform vary depending on the type of object you select.

If you select a linked file, you can:

- Click View/Launch to view the contents of the file. Hyperion 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 Hyperion 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.

Note: 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 Hyperion Essbase 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 the `DOCS` directory for information about these commands. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Chapter 13

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 Outline Editor.

- The data source can specify member names, their aliases, formulas and consolidation properties, the names of generations and levels, currency name and category, data storage properties, attributes, and user-defined attributes (UDAs).
- The rules file tells Hyperion Essbase which build method to use, specifies whether the data is in sorted or random order, and tells Hyperion Essbase how to transform the data before loading it. You use Data Prep Editor to create rules files. It is best to create a separate rules file for each dimension. For more information on data sources and rules files, see Chapter 20, “Introducing Data Loading.”

This chapter provides a background on dynamic dimension building.

- “Building Shared Members Using a Rules File” on page 13-32
- “Security and Multi-User Considerations” on page 13-46

For information on how to build dimensions dynamically, see Chapter 14, “Building Dimensions Using a Rules File.”

Workflow for Creating Dimensions Using Rules Files

This section describes the process for building dimensions using a rules file. For more information on data sources and rules files, see Chapter 20, “Introducing Data Loading.”

➤ 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 Data Prep Editor to create a dimension build rules file. Select View > Dimension Building Fields or click the Dimension Build button,  , on the toolbar to put Data Prep Editor into dimension build mode.

   Set field and record operations in the rules file. The field and record operations determine how Hyperion Essbase transforms the data before building the dimensions. For more information on how to set field and record operations, see Chapter 21, “Setting up a Rules File to Manipulate Records.”

3. Select Options > Dimension Build Settings to open the Dimension Build Settings dialog box. This dialog box contains three tabs, each of which sets up a different part of the rules file:
   - Use the Dimension Definition tab to name the dimension and define its properties.
   - Use the Dimension Build Settings tab to specify the build method and indicate how to sort the members.
   - Use the Global Settings tab to specify which alias table to update, whether Hyperion Essbase should select 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 21-21.
5. Perform the dimension build. See Chapter 23, "Performing a Data Load."

Introduction to Build Methods

The build method that you select determines the algorithm that Hyperion Essbase uses to add, change, or remove dimensions, members, and aliases in the outline. The kind of build method that you select depends on your data source.

Each record in a data source defines a single member of a dimension. Hyperion Essbase can read data sources in a variety of formats:

- Top-down data sources beginning with the most general information and progress to the most specific. Each record specifies the parent's name, the child's name, the children of that child, and so forth.
- Bottom-up data sources provide a complete description of the member, beginning with the most specific information and progressing to the most general. Each record specifies the name of the member, followed by the name of its parent, the name of its parent's parent, and so forth.
- Parent/child data sources specify the name of the parent and the name of the new child member, in that order, although they can specify other information as well.
- Other data sources consist of lists of new members; they do not specify where in the outline these members belong. Hyperion 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 13-1: 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 properties of existing dimensions and members	Generation references	The generation number for each field
Bottom-up data	Month, Quarter, Year	Create shared members that roll up into different generations Modify the properties of existing dimensions and members	Level references	The level number for each field
A parent followed by its child	Cola, Diet Cola	Create shared members that roll up into different generations Share non-leaf members Modify properties of existing dimensions and members	Parent/child references	Whether each field is the parent or child
A list of base dimension members and their attributes	Cola 16oz Can, Root Beer 14oz Bottle	Add members to the attribute dimension and associate them with members of the base dimension	Generation, level, or parent/child references, depending on the organization of the source data	The number for each field, as appropriate: • Generation number or level number of the associated member of the base dimension • Zero

Table 13-1: Build Method Guidelines (Continued)

If your data source contains...	Such as...	And you want to...	Use this build method...	And specify...
A list of new members	Jan, Feb, Mar, April	Add them all as children of an existing (possibly a "dummy") parent	Add as child of specified parent	
	800-10, 800-20	Add them at the end of the dimension	Add as sibling of lowest level	
	800-10, 800-20	Add each new member to a dimension with similar members	Add as sibling of member with matching string	

Note: Hyperion Essbase does not support concurrent attribute association with the Add as Sibling and Add as Child build methods.

Using Generation References

Top-down data sources are organized from the highest level down. Each record begins with the most general information and progresses to the most specific. The name of the member is at the end of the record. When building a dimension with a top-down data source, use the generation references build type. Use the rules file to specify the generation number and field type for each field in the data source. For more information about creating a rules file, see Chapter 21, "Setting up a Rules File to Manipulate Records."

Hyperion 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 13-1: Generations

The top half of Figure 13-2 shows a top-down data source used to build the Product dimension, `GENREF.TXT`. The bottom half of Figure 13-2 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 "Step 4: Setting Rules File Field Types" on page 14-12.

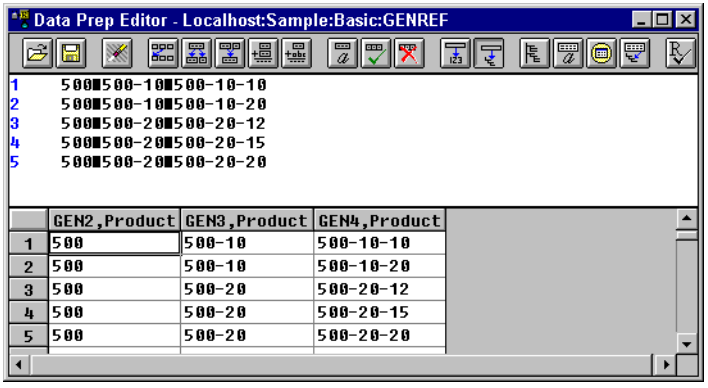


Figure 13-2: Rules File for Generation Build

Figure 13-3 shows the tree Hyperion Essbase builds from this data source and rules file:

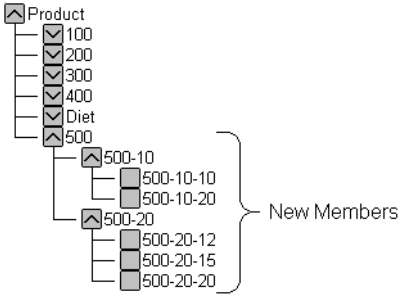


Figure 13-3: Generation References

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, Hyperion Essbase processes nulls as follows:

- If the null occurs where Hyperion Essbase expects a GENERATION field, Hyperion Essbase promotes the next GENERATION field in place of the missing one. For example:

  ```
  GEN2,Products  GEN3,Products  GEN4,Products
  100                           100-10a
  ```

  When Hyperion Essbase reads the above file, it promotes the GEN4 field (100-10a) to GEN3.

- If a null occurs directly before a secondary field, Hyperion Essbase ignores the secondary field. Secondary field options are alias, property, formula, duplicate generation, duplicate generation alias, currency name, currency category, attribute parent, UDA, and a name of an attribute dimension. For example:

  ```
  GEN2,Products   ALIAS2,Products   GEN3,Products   GEN4,Products
                                      100-10          100-10a
  ```

  When Hyperion 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 Hyperion Essbase expects a secondary field, Hyperion Essbase ignores that field and continues loading. For example:

```
GEN2,Products  ALIAS2, Products  GEN3,Products  GEN4,Products
100                              100-10         100-10a
```

  When Hyperion Essbase reads the above file, it ignores the ALIAS2 field.

Using Level References

In bottom-up data sources, each record defines a single member of a dimension. The definition begins with the most specific information about the member and provides progressively more general information. A typical record would specify the member itself, then the name of its parent, then its parent's parent, and so forth.

Levels are defined from a bottom-up hierarchical structure. In the outline in Figure 13-4, for example, the lowest level members are at the bottoms of the branches in the Product dimension.

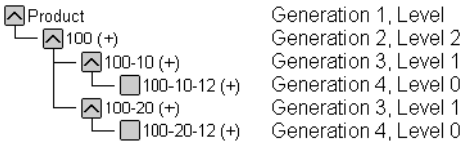


Figure 13-4: Generation and Level Numbers

To build the outline in Figure 13-4, you can use the bottom-up data source shown in Figure 13-5.

field 1	field 2	field 3
100-10-12	100-10	100
100-20-12	100-20	100

Figure 13-5: Bottom-up Data Source

In a level reference build, the lowest level members are sequenced left to right. Level 0 members are in the first field, level 1 members are in the next field, and so on. This is the opposite of how data is specified for generation references (top-down).

The rules file in Figure 13-6 uses the level reference build method to add members to the Product dimension in the Sample Basic database. The first column of the data source contains new members (100-10-12, 100-10-16, 100-20-12, and 100-20-16). Subsequent columns contain their parents. (Family-18 oz., and Family-24 oz.)

The rules file specifies the level number and field type for each field in the data source. For more information, see “Step 4: Setting Rules File Field Types” on page 14-12. To build the tree in Figure 13-7 for example, use Figure 13-6 to set up the data source, `LEVEL.TXT`, and rules file, `LEVEL.RUL` to match.

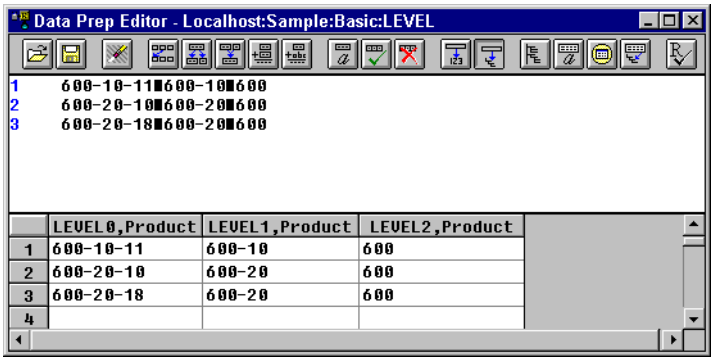


Figure 13-6: Rules File for Level Build

Figure 13-7 shows the tree Hyperion Essbase builds from this data source and rules file.

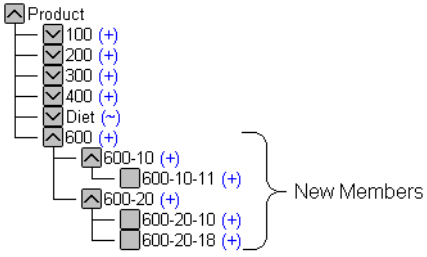


Figure 13-7: Levels

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, Hyperion Essbase processes nulls as follows:

- If the null occurs where Hyperion Essbase expects a LEVEL field, Hyperion Essbase promotes the next LEVEL field in place of the missing one. For example:

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

  When Hyperion 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, Hyperion Essbase ignores the secondary field. Secondary field options are alias, property, formula, duplicate level, duplicate level alias, currency name, currency category, attribute parent, UDA, and a name of an attribute dimension.

  For example:

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

  When Hyperion 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 Hyperion Essbase expects a secondary field, Hyperion Essbase ignores that field and continues loading. For example:

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

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

Using Parent/Child References

Use the parent/child references build method with data sources in which each record specifies the name of the new member and the name of the parent to which you want to add it.

Members in a database exist in parent/child relationships to one another. Figure 13-8 shows part of the Product dimension with its parent and children relationships identified.

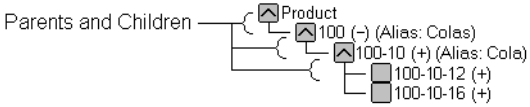


Figure 13-8: Parents and Children

Parent/child data sources must contain at least two columns: the parent column and the child column, in that order. They can include columns with other information as well (for example, the alias of the new member, its attributes or its properties). The data source *cannot* specify more than one parent or more than one child, and cannot reverse the order of the parent and child columns.

In a parent/child build, the rules file specifies which column is the parent and which the child. For more information, see “Step 4: Setting Rules File Field Types” on page 14-12. For example, the top half of Figure 13-9 shows a data source, `PARCHIL.TXT`, in which each record specifies the name of a parent and the name of its child, in that order. The bottom half of the figure shows the rules

file, PARCHIL.RUL, that specifies which column is the parent and which is the child. In addition to parent and child fields, this example demonstrates associating aliases with the parent field.

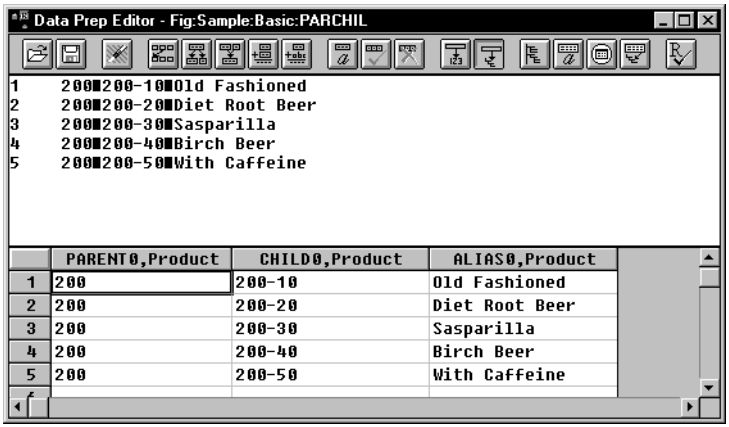


Figure 13-9: Rules Files for Parent/Child Build

Figure 13-10 shows the tree that Hyperion Essbase builds from this data source and rules file.

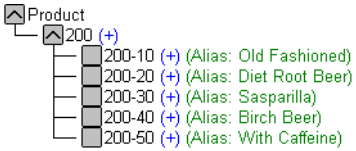


Figure 13-10: Parents and Children

Adding a List of New Members

If a data source consists of a list of new members, without specifying their ancestors, Hyperion Essbase must decide where in the outline to add the new members. Hyperion 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, Hyperion Essbase can:

- Add each new member as a sibling of the existing member whose text most closely matches its own.
- Add each new member as a sibling of the lowest existing member.
- Add all new members as children of a specified parent (generally a "dummy" parent).

After you select your build method, you must specify the dimension to which each field maps. See "Step 4: Setting Rules File Field Types" on page 14-12.

After Hyperion Essbase has added all the new members to the outline, use Outline Editor to examine the outline. If necessary, move the new members into their correct positions. See Chapter 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 Hyperion Essbase encounters a new member in a data source, it scans the outline for a member name with similar text. Hyperion Essbase then adds the new member as a sibling of the member with the closest string match.

For example, the data source in Figure 13-11, `SIBSTR.TXT`, contains two new members to add to the Product dimension in the Sample Basic database, 100-11 and 200-22. They are similar to strings in the Product dimension (they contain 3 digits, a dash, and two digits).

To add these members dynamically to the database, set the following values to match the rules file, `SIBSTR.RUL`, in Figure 13-11.

- For the Product column (field 1) on the Dimension Building Properties tab of the Field Properties dialog box:
  - Do not select a field type for the field.
  - Set the dimension for the field to Product. Field 1 is displayed as Product, as shown in Figure 13-11.
- For the other data fields that are irrelevant to this operation (fields 2 through 6), on the Global Properties tab of the Field Properties dialog box, select "Ignore field during dimension build."

- In the Dimension Build Settings dialog box, for the Product dimension, select the “Add as sibling of matching string” method.

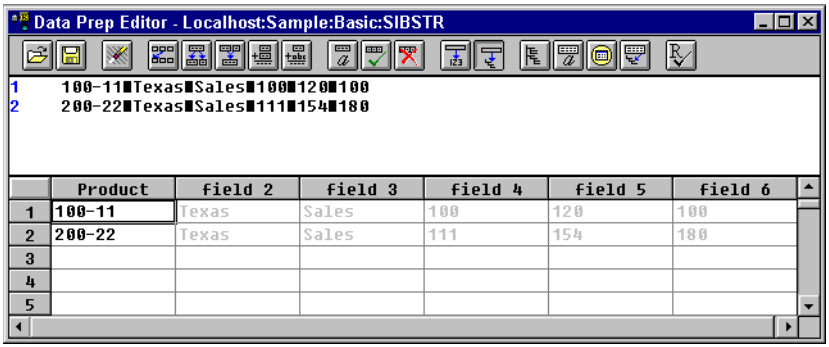


Figure 13-11: Rules File Fields Set to Add Members as Siblings with String Matches

When you load the data in Figure 13-11, Hyperion Essbase builds the following member tree:

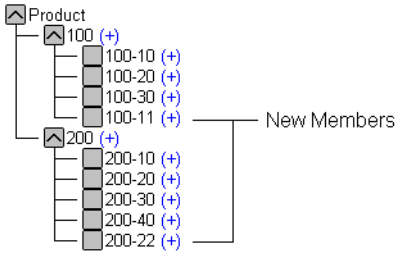


Figure 13-12: Adding Members as Siblings with String Matches Tree

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 Hyperion Essbase encounters a new member in a data source, it scans the outline for the lowest branch of members. Hyperion 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, Hyperion 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 13-13 contain new members (A100-10 and A100-99) to add to the Measures dimension in the Sample Basic database.

To add these members dynamically to the database, set the following values in the rules file:

- For the Measures column (field 3) on the Dimension Building Properties tab of the Field Properties dialog box:
  - Do not select a field type for the field.
  - Set the dimension for the field to Measures. Field 3 is displayed as Measures, as shown in Figure 13-13.
- For the other data fields that are irrelevant to this operation (fields 1, 2, 4, 5, and 6), on the Global Properties tab of the Field Properties dialog box, select "Ignore field during dimension build."

- In the Dimension Build Settings dialog box, for the Measures dimension, select the “Add as sibling of lowest level” build method.

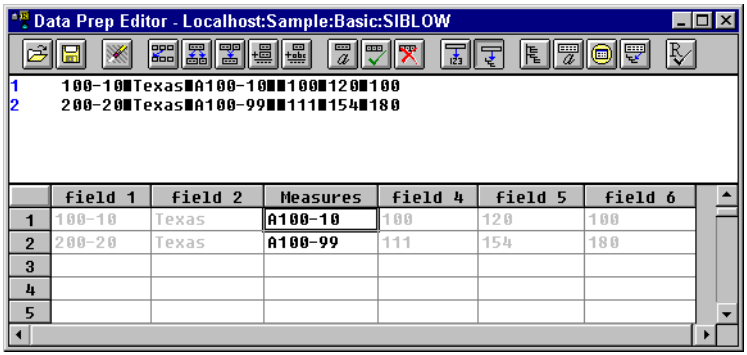


Figure 13-13: Rules File Fields Set to Add Members as Siblings of the Lowest Level

When you load the data in Figure 13-13, Hyperion Essbase builds the following member tree:

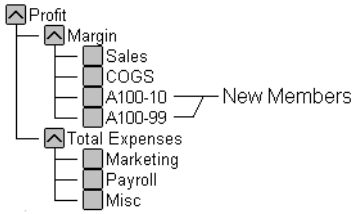


Figure 13-14: Adding Members as Siblings of the Lowest Level

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 Hyperion Essbase adds all new members to the outline, review the added members and move or delete them in Outline Editor.

When Hyperion 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 13-15, `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 Hyperion Essbase add these members dynamically to the database under the NewProducts member, set the following values:

- For the Product column (field 1) on the Dimension Building Properties tab of the Field Properties dialog box:
  - Do not select a field type for the field.
  - Set the dimension for the field to Product. Field 1 is displayed as Product, as shown in Figure 13-15.
- For the other data fields that are irrelevant to this operation (fields 2 through 6), on the Global Properties tab of the Field Properties dialog box, select "Ignore field during dimension build."
- In the Dimension Build Settings dialog box, for the Product dimension, select the "Add as child of build" method and type `NewProducts` in the associated text box.

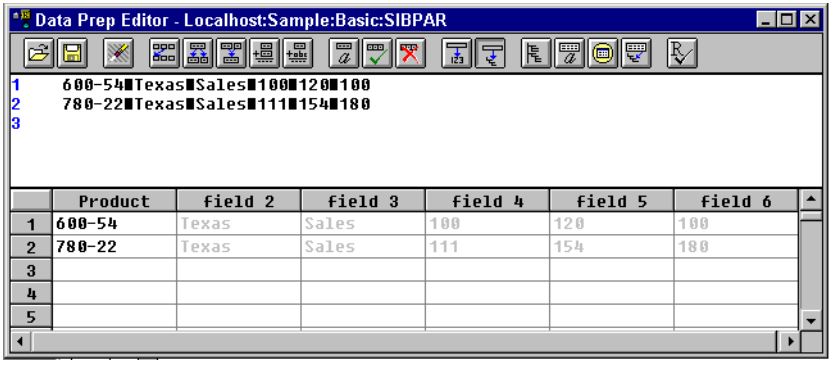


Figure 13-15: Rules File Fields Set to Add Members as a Child of a Specified Parent

When you load the data in Figure 13-15, Hyperion Essbase builds the following member tree:

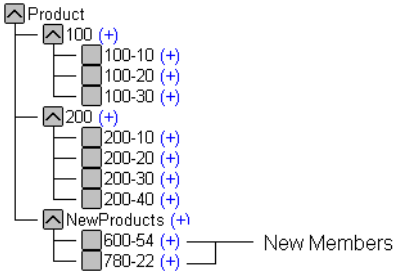


Figure 13-16: Adding Members as a Child of a Specified Parent

Building Attribute Dimensions and Associating Attributes

This section shows examples of rules files that are used to build attribute dimensions and to associate attributes with members of their base dimensions.

You can use rules files to build attribute dimensions dynamically, to add and delete members, and to establish or change attribute associations.

Working with attributes involves the three following operations:

- If the base dimension does not exist, you must build it.
- You must build the attribute dimension.
- You must associate members of the base dimension with members of the attribute dimension.

You can use any of three approaches to perform these operations.

When you use an all-at-once approach, you use a single rules file to build the base dimension and one or more attribute dimensions and to associate the attributes with members of the base dimension. Because this approach uses a single rules file, it can be the most convenient. Use this approach if the base dimension does not exist yet and each source data record contains all attribute information for each member of the base dimension.

If the base dimension is built in a separate step or already exists, you can build an attribute dimension and associate the attributes with the members of the base dimension in either of two ways:

- In a single step. You need only to define the attribute associations in the rules file. See "Associating Attributes" on page 13-19.
- In separate steps. Build the attribute dimension, and then associate the attribute members with members of the base dimension. You must use this approach when you build numeric attribute dimensions that are multilevel or that have members that represent different-sized ranges.

Building Attribute Dimensions

Before you build any attribute dimensions in a database, you must define the attribute member name formats for the outline. See "Working with the Names of Members of Attribute Dimensions" on page 10-15.

You can build attribute dimensions in either of the following two ways:

- The same way that you build standard dimensions, as described in "Workflow for Creating Dimensions Using Rules Files" on page 13-2.
- At the same time as you associate attributes with members of the base dimension, as described in "Associating Attributes" on page 13-19.

Hyperion Essbase does not support concurrent attribute association with the following three build methods: "Add as sibling of mbr with matching string," "Add as sibling of lowest level," and "Add as child of."

When you define the rules file for building attribute dimensions, be sure to specify the base dimension and the name of the attribute dimension file, as described in "Naming New Attribute Dimensions" on page 14-4.

Associating Attributes

Whether you build the attribute dimension at the same time as you associate its members with members of the base dimension or perform these tasks in separate steps, define the fields as described in this section.

However, when you are working with a multilevel attribute dimension or with an attribute dimension of the type numeric, Boolean, or date, the rules file requires an additional field. For a complete example of a multilevel situation, see “Working with Multilevel Attribute Dimensions” on page 13-22.

Each record in the source data must include at least a column for the member of the base dimension and a column for the member’s attribute value. In the same source data record you can include additional columns for other attributes that you want to associate with the member of the base dimension. You must position the field for the member of the base dimension before any of the fields for the members of the attribute dimension.

Define the field type for the attribute dimension member as the name of the attribute dimension, use the generation or level number of the associated member of the base dimension, and specify the base dimension name. For example, as shown in the `ATTRPROD.RUL` file in Figure 13-17, the field definition Ounces3,Product specifies that the field contains members of the Ounces attribute dimension. Each member in this field is associated with the data field that is defined as the generation 3 member of the base dimension Product. Based on this field definition, Hyperion Essbase associates the attribute 64 with product 500-10.

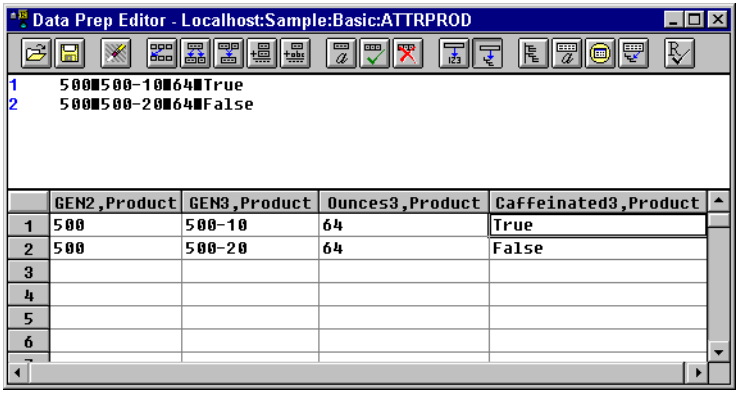


Figure 13-17: Rules File for Associating Attributes

To have Hyperion Essbase use the data attribute columns to build members in the attribute dimensions, on the Dimension Build Settings tab in the Dimension Build Settings dialog box, for the base dimension, leave the Do Not Create Mbrs option clear. See “Step 3: Specifying Changes to Dimensions” on page 14-10.

When you are working with numeric ranges, you may have to build attribute dimensions and perform associations in separate steps. See “Working With Numeric Ranges” on page 13-25.

The Caffeinated3,Product field in the example in Figure 13-17 shows how to associate attributes from additional single-level attribute dimensions. Because the base dimension is already specified, you need only to define an additional field for each attribute that you want to associate with the member of the base dimension.

The file in Figure 13-17 associates attributes as shown in the resulting outline in Figure 13-18. The members 500, 500-10, and 500-20 are new members of the base dimension, Product. The member 64 is a new member of the Ounces attribute dimension.

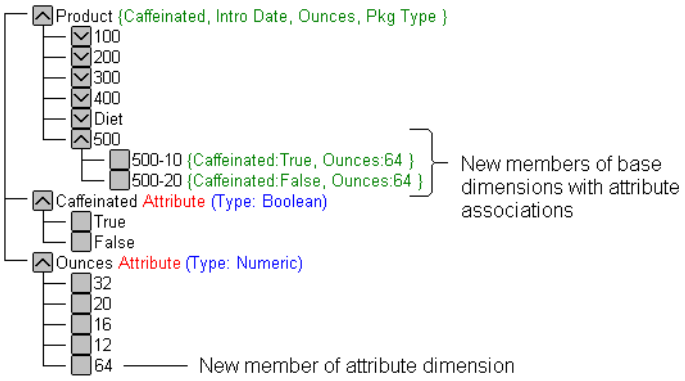


Figure 13-18: Associating Attributes

Updating Attribute Associations

You can also use the rules file shown in Figure 13-17 to change attribute associations. Make sure that you select “Allow Association Chgs” for the base dimension in the Dimension Build Settings tab of the Dimension Build Settings dialog box. See “Step 3: Specifying Changes to Dimensions” on page 14-10.

Working with Multilevel Attribute Dimensions

Multilevel, numeric, Boolean, or date attribute dimensions can have duplicate level 0 members. For example, associated with a Product dimension you can have a Size attribute dimension with two levels. Level 1 categorizes sizes by men or by women. The level 0 members (attributes) are the actual sizes. You can have a member named 8 under Women and member named 8 under Men.

When the attribute is part of a multilevel numeric, Boolean, or date attribute dimension, the source data must include columns for all generations or levels of the attribute dimension. In the rules file, you must make copies of all fields that comprise the levels of the attribute dimension. Define the first set of the attribute fields to build the attribute dimension. Define the second set of attribute fields to associate the attributes with the base dimension member. To ensure association with the correct attribute, to indicate the parent field for the attribute field, make a copy of the parent field and set the copy of the parent field as the field type Attribute Parent.

The position of the fields in the rules file is important.

- Place the copied attribute dimension field or fields that define the association immediately to the right of the field for the members of the base dimension.
- For a multilevel attribute dimension, place the attribute parent field immediately to the left of the field that is the child of the attribute parent.

The following procedure describes how to define the fields in the rules file to build this multilevel attribute dimension and associate its members with members of its base dimension. This example uses the level references build method.

1. In the rules file, in field 1 and field 2, define the attribute dimension fields in the same way that you define standard dimensions; specify level or generation type and number and dimension name.

   Hyperion Essbase uses these two fields to build the attribute dimension.

2. Define the fields for building the base dimension, in this example, the level 0 and level 1 fields for the Product dimension.

Figure 13-19 shows the fields of the rules file at this stage.

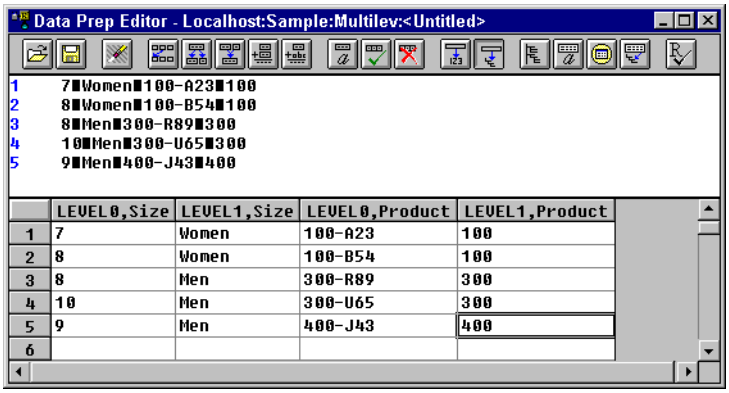


Figure 13-19: Defining Multilevel Attribute Dimensions Before Adding the Association Fields

3. To define the association, select Field > Create Using Join to make a copy of the field that contains the level 0 attribute. In this example, make a copy of field 1.
   - Use the attribute dimension name as the field type and specify the generation or level number of the member of the base dimension with which Hyperion Essbase associates the attribute; for example, Size0.
   - Specify the base dimension; for example, Product.
   - Select Field > Move to move the new field immediately to the right of the field for the base dimension with which Hyperion Essbase associates the attribute. In this example, move it to the right of the field Level0,Product.
4. Make a copy of the field containing the parent of the attribute field; in this example, make a copy of field 2.
   - Set the field type of this new field as Attribute Parent and specify the generation or level number of the base member with which you want Hyperion Essbase to associate the attribute; for example, ATTRPARENT0.
   - Specify the attribute dimension; for example, Size.
   - Move the ATTRPARENT field immediately to the left of the attribute association field that you created in step 3.

As shown in Figure 13-20, the rules file now contains the field definitions to build the attribute dimension Size and to associate the members of Size with members of the base dimension Product.

Data Prep Editor - Localhost:Sample:Multilev:multilev

```
1    7■Women■100-A23■100
2    8■Women■100-B54■100
3    8■Men■300-R89■300
4    10■Men■300-U65■300
5    9■Men■400-J43■400
```

	LEVEL0,Size	LEVEL1,Size	LEVEL0,Product	ATTRPARENT0,Size	Size0,Product	LEVEL1,Product
1	7	Women	100-A23	Women	7	100
2	8	Women	100-B54	Women	8	100
3	8	Men	300-R89	Men	8	300
4	10	Men	300-U65	Men	10	300
5	9	Men	400-J43	Men	9	400

Figure 13-20: Source Data and Rules File for Building a Multilevel Attribute Dimension

When you run a dimension build with the data shown in Figure 13-20, Hyperion Essbase builds the Size attribute dimension and associates its members with the members of the base dimension. Figure 13-21 shows the updated outline.

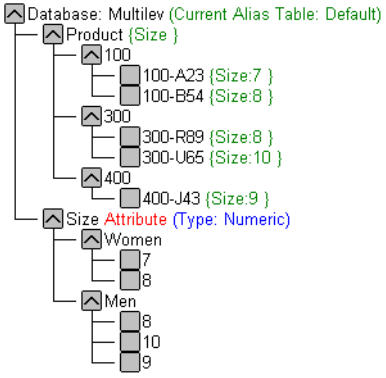


Figure 13-21: Multilevel Attribute Dimension

Working With Numeric Ranges

In many cases, you can use one rules file in a single dimension build operation to dynamically build attribute dimensions for numeric ranges and to associate the members of the base dimension with the ranges. However, in the following situations you must use two rules files, one to build the attribute dimension and one to associate the attributes with members of the base dimension:

- When the size of the range is different for different members. For example, you can define several shorter ranges for towns and cities with smaller populations, larger ranges for mid-sized cities, and ranges of 1,000,000 for cities with large populations.
- When the ranges are members of a multilevel attribute dimension. For example, the Population attribute dimension can have level 1 members that categorize the population ranges as Towns, Cities, and Metropolitan Areas.

The Population attribute dimension shown in Figure 13-22 demonstrates both situations. Population is a multilevel, numeric attribute dimension with level 0 members representing ranges of different sizes.

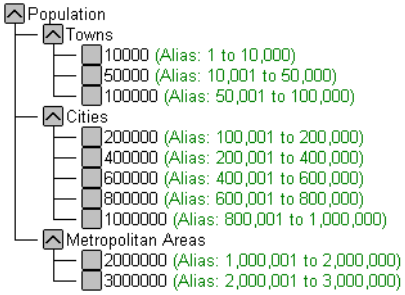


Figure 13-22: Numeric Attribute Dimension with Different-Sized Ranges

You must use one rules file to build this Population dimension and a second rules file to associate its members as attributes of members of the base dimension.

Building the Attribute Dimension

First, create a rules file that uses the generation, level, or parent/child build method to build the attribute dimension. In this rules file, be sure to specify the following:

- The name of the attribute dimension and its associated base dimension. See “Naming New Attribute Dimensions” on page 14-4.
- The fields for building the attribute dimension.

The source data must be in attribute sequence, in ascending order. If the ranges have different sizes, the source data must include a record for every attribute range.

Note: In later builds you cannot insert attribute members between existing members.

To use the generation method to build the outline in Figure 13-22, you must sequence the source data in ascending sequence, based on the numeric attribute value. Define the fields in a rules file as shown in Figure 13-23.

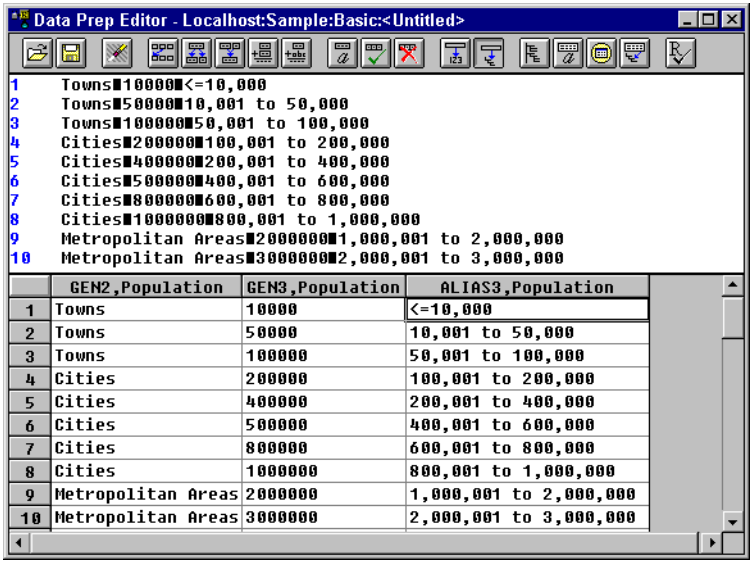


Figure 13-23: Rules File for Building a Numeric Attribute Dimension with Ranges

Figure 13-23 also shows how you can associate aliases with attributes.

Associating the Base Dimension Members with Their Range Attributes

After you build the numeric attribute dimension ranges, you need a rules file to associate the members of the base dimension with their attributes. The source data includes fields for the members of the base dimension and a field for the data values that Hyperion Essbase uses to associate the appropriate Population attribute.

Define the rules file as shown in Figure 13-24.

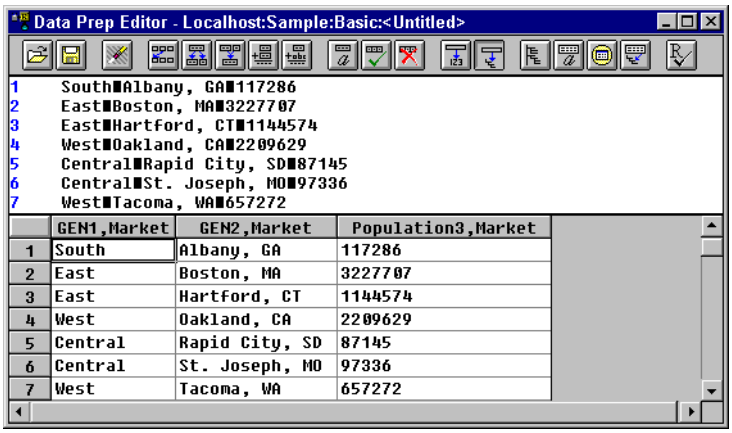


Figure 13-24: Rules File for Associating Numeric Range Attributes

When you define the association field (for example, Population3,Market) be sure to click the Ranges button to open the Numeric Range Rules dialog box. Select "Place attribute members within a range."

Note: Figure 13-24 includes a city, Boston, whose population of 3,227,707 is outside the existing ranges in the attribute dimension in Figure 13-22. (The ranges in Figure 13-22 go up only to 3,000,000.)

To allow for values in the source data that are outside existing ranges in the attribute dimension, enter a range size, such as 1000000, in the Numeric Range Rules dialog box. Hyperion Essbase uses the range size to add members to the attribute dimension above the existing highest member or below the existing lowest member, as needed.

CAUTION: After you associate members of the base dimension with members of the attribute dimension, be aware that if you manually insert new members into the attribute dimension or rename existing members of the attribute dimension, you may invalidate existing attribute associations.

Consider an example where numeric range attributes are defined as "Tops of ranges" and an attribute dimension contains members 100, 200, 500, and 1000. A base dimension member with the value 556 is associated with the attribute 1000. If you rename a member of the attribute dimension from 500 to 600, the base dimension member with the value 556 now has an invalid association. This base member is still associated with the attribute 1000 when it should now be associated with the attribute 600.

If you manually insert new members or rename existing members, to ensure associations are correct, you should rerun the dimension build procedure that associates the base members with the changed attribute dimension. For example, rerunning the attribute association procedure would correctly associate the member of the base dimension with the value 556 with the new attribute 600.

Working With Ranges

To ensure the validity of the attribute associations, you must be careful to select the correct dimension building options and to perform the builds in the proper sequence.

Adding or Changing Members of the Attribute Dimension: After you associate members of a base dimension with their numeric attribute ranges, if you manually insert new members or rename existing members in the attribute dimension, you should make sure that associations between attributes and base members are correct. To ensure that the associations are correct, you can do one of the following:

- Rerun the dimension build procedure that associates the base members with the changed attribute dimension.
- Through Outline Editor, manually review and fix, as needed, the associations of all base dimensions.

Deleting Members from the Attribute Dimension: You can delete all existing members of an attribute dimension so you can rebuild the dimension with the new data. Select "Delete all members of this attribute dimension" in the Numeric Range Rules dialog box. Hyperion Essbase uses the start value and range size value to rebuild the attribute dimension. To ensure proper attribute association, on the Dimension Build Settings tab in the Dimension Build Settings dialog box, for the base dimension you must select the "Allow Association Chgs" option.

Adding Members to the Base Dimension: You can use the same rules file to add new members simultaneously to the base dimension and to associate the new members with their numeric range attributes. In the Numeric Range Rules dialog box for the attribute dimension, be sure to provide a value for the range size.

If Hyperion Essbase encounters a new base dimension value that is greater than the highest attribute member by more than the range size or is lower than the lowest attribute member by more than the range size, it creates members in the attribute dimension for any intermediate ranges.

Consider the example, in Figure 13-22, where numeric range attributes are defined as "Tops of ranges". The highest value member of the Population attribute dimension is 3000000. If the source data includes a record with the population

4,420,000 and the range size is 1000000, Hyperion Essbase adds two members to the attribute dimension, 4000000 and 5000000, and associates the base member with the 5000000 attribute.

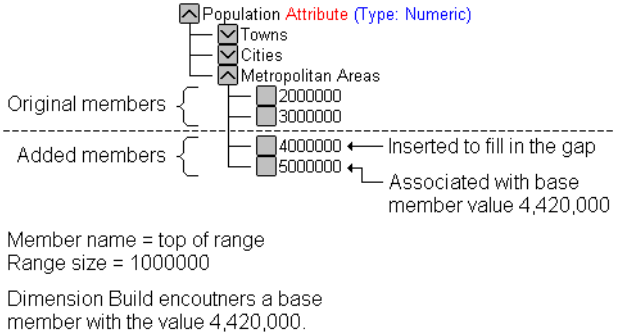


Figure 13-25: Dynamically Adding Attribute Range Members

When you add range members and base dimension members at the same time, Hyperion Essbase does not create aliases for the new members of the attribute dimension. If you want aliases that describe the range values for the new members of the attribute dimension, you must add the aliases in a separate operation.

Rules Summary for Building Attribute and Base Dimensions

The following list describes a few areas unique to defining and associating attributes through dimension build.

Getting Ready

- Before running a dimension build, you must define the attribute member name formats for the outline. See “Working with the Names of Members of Attribute Dimensions” on page 10-15.
- Defining new attribute dimensions in a rules file is different from defining new standard dimensions in a rules file. See “Naming New Attribute Dimensions” on page 14-4.

Defining Fields in Rules Files

Rules files that are used to build single-level attribute dimensions require fewer field types than rules files that build and associate members of multilevel attribute dimensions.

- For single-level attribute dimensions, define the field that contains the attribute values as the field to be associated with members of the base dimension. Dimension build also uses this field to add new members to the attribute dimension. See “Associating Attributes” on page 13-19.
- For multilevel attribute dimensions, Hyperion Essbase requires fields that define each generation or level in the attribute dimension in addition to fields that define the associations. Use the new field type, Attribute Parent, to identify fields that are parent members for the attribute members being associated. See “Working with Multilevel Attribute Dimensions” on page 13-22.

Controlling Adding New Attribute Members

When Hyperion Essbase encounters attribute data values that are not members of the attribute dimension, it automatically adds the values as new members. To prevent adding new members to attribute dimensions, you must select the Do Not Create Mbrs option for the attribute dimension in the Dimension Build Settings dialog box. See “Step 3: Specifying Changes to Dimensions” on page 14-10.

Controlling Associations

- Hyperion Essbase does not make changes to attribute associations unless you select the Allow Association Chgs option for the attribute dimension in the Dimension Build Settings page of the Dimension Build Settings dialog box. See “Step 3: Specifying Changes to Dimensions” on page 14-10.
- To enable automatic association of base members with attributes that represent ranges of values, make sure you have defined the size of the range in the Advanced Numeric Rules dialog box. See “Setting Field Information” on page 14-13.
- Hyperion Essbase does not support concurrent attribute association with the two Add as Sibling methods and the Add as Child build method.

Note: Because attributes are defined only in the outline, the data load process does not affect them.

Building Shared Members Using a Rules File

The data associated with shared members comes from another real member with the same name. The shared member stores a pointer to data contained in the real member; thus the data is shared between the members and is only stored one time.

In the Sample Basic database, for example, the 100-20 (Diet Cola) member rolls up into the 100 (Cola) family and the Diet family.

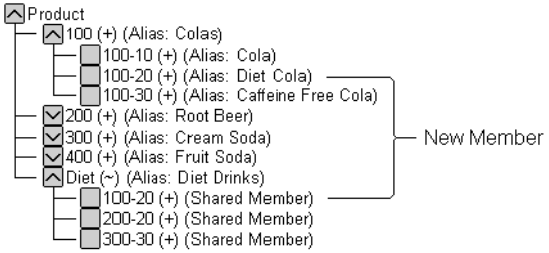


Figure 13-26: Shared Members in the Sample Basic Database

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 13-26, Diet Cola is shared by two members at generation 2 in the outline, but it could be shared by a member at generation 3 and a member at generation 4 as in Figure 13-34.

While creating shared members at different generations in the outline is easy in Outline Editor, they are a little more difficult to create using dynamic dimension building. You must pick your build method and format your data source carefully. The following sections describe how to build shared members in the outline using a data source and rules file.

Note: You should not create an outline where shared members are located before actual members in a dimension.

Sharing Members at the Same Generation

Members that are shared at the same generation roll up into the same branch. In the Sample Basic database, 100-20 (Diet Cola) is shared by two parents. Both parents roll up into the same branch, that is, the Product dimension, and both parents are at generation 2.

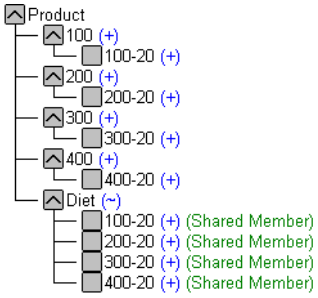


Figure 13-27: Members Shared at the Same Generation

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 13-33
- “Creating Shared Members Using Level References” on page 13-35
- “Creating Shared Members Using Parent/Child References” on page 13-36

Creating Shared Members Using Generation References

To create shared member parents at the same generation using the generation references build method, define the field type for the parent of the shared member as DUPGEN. A *duplicate generation* is a generation with shared members for children. Use the same GEN number as the primary member.

For example, to create the Diet parent and share the 100-20, 200-20, 300-20, and 400-20 members, use the sample file, `SHGENREF.TXT`, and set up the rules file so that the fields look like `SHGENREF.RUL`, shown in Figure 13-28. Remember 100 is the Cola family, 200 is the Root Beer family, 300 is the Cream Soda family and the -20 after the family name indicates a diet version of the soda.

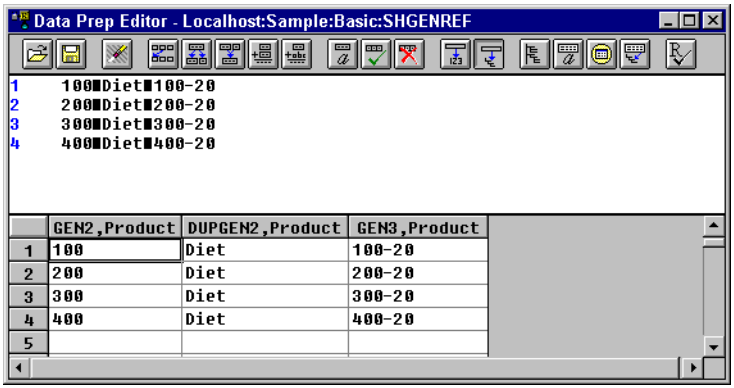


Figure 13-28: Sample Generation Shared Member Rules File

This builds the following tree:

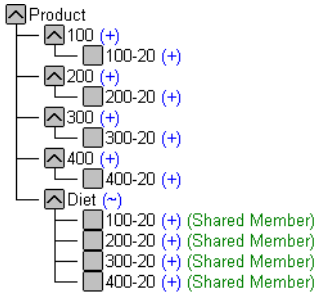


Figure 13-29: Sample Generation Shared Member Rules Tree

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, Hyperion Essbase creates a parent at the specified level and inserts the shared members under it.

For example, to create the shared 100-20 (Diet Cola), 200-20 (Diet Root Beer), 300-20 (Diet Cream Soda), and 400-20 (Fruit Soda) members in the Sample Basic database, use the sample file, `SHLEV.TXT`, and set up the rules file so that the fields look like `SHLEV.RUL` shown in Figure 13-30.

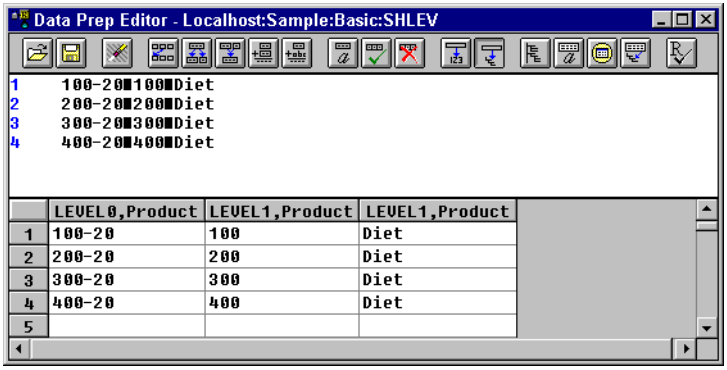


Figure 13-30: Sample Level Shared Member Rules File

This builds the following tree:

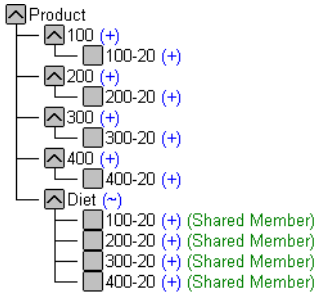


Figure 13-31: Sample Level Shared Member Rules Tree

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

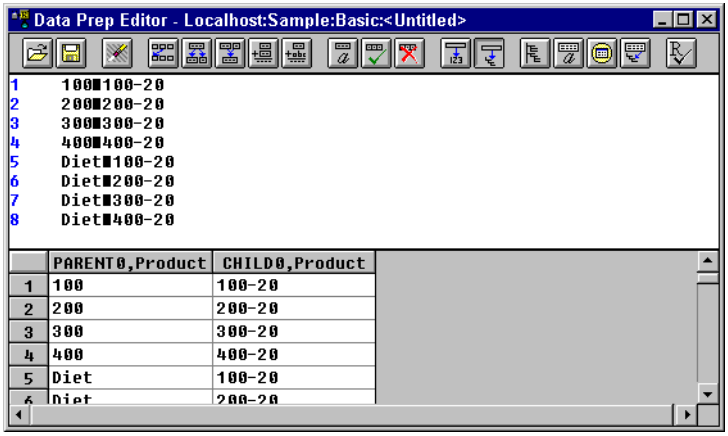


Figure 13-32: Sample Parent/Child Shared Members Rules File

This builds the following tree:

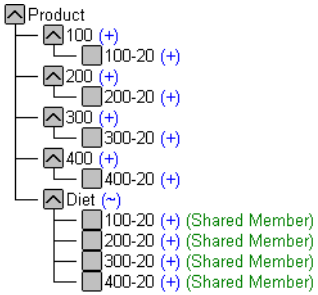


Figure 13-33: Sample Parent/Child Shared Member Rules Tree

Sharing Members at Different Generations

Sometimes you want shared members to roll up into parents at different generations in the outline. In Figure 13-34, for example, the shared members roll up into parents at generation 2 and generation 3 in the outline. This outline assumes that The Beverage Company (TBC) buys some of its beverages from outside vendors. In this case, it buys 200-20 (Diet Root Beer) from a vendor named Grandma's.

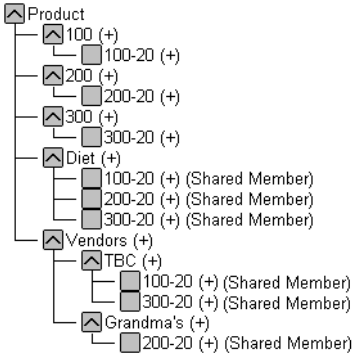


Figure 13-34: Members Shared at Different Generations

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

- "Creating Shared Members Using Level References" on page 13-38
- "Creating Shared Members Using Parent/Child References" on page 13-39

Creating Shared Members Using Level References

To create shared members at different generations using the level references build method, first make sure that both primary and secondary roll-ups are specified in one record. You can specify as many secondary roll-ups as you want, as long as the roll-ups are all in one record.

Define the field type for the shared member as LEVEL. Then enter the level number. While processing the data source, Hyperion 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 13-35.

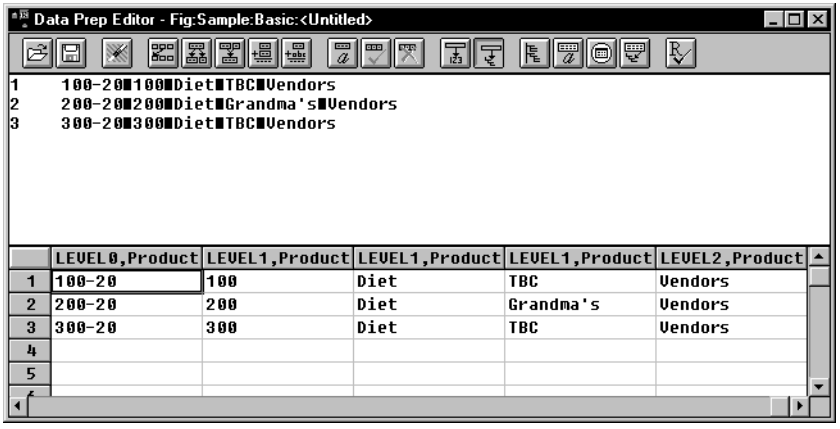


Figure 13-35: Level References Sample Rules File for Shared Members at Different Generations

This builds the tree in Figure 13-34.

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 selected in the Dimension Build Settings page of the Dimension Build Settings dialog box so that Hyperion Essbase creates duplicate members as shared when it encounters them under a new parent.

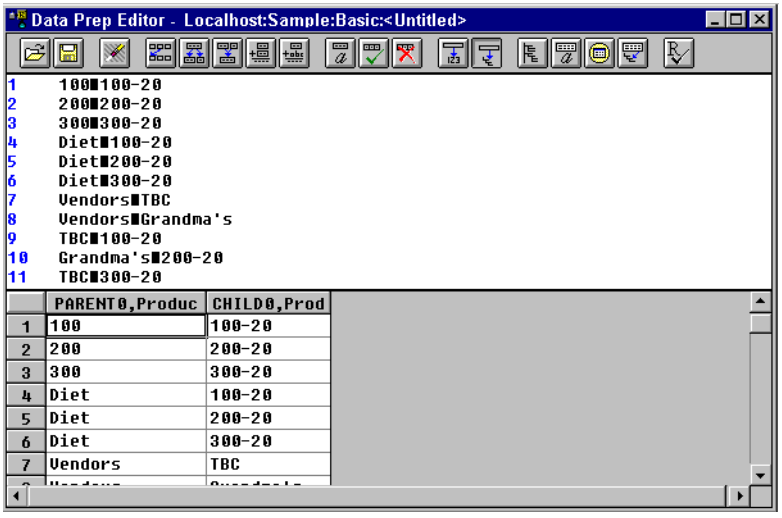


Figure 13-36: Parent/Child References Sample Rules File for Shared Members at Different Generations

This builds the tree in Figure 13-34.

Sharing Members with Branches

Sometimes you want to share non-leaf members (members that are not at the lowest generation). In Figure 13-37 for example, 100, 200, and 300 are shared by TBC and Grandma's. This outline assumes that TBC (The Beverage Company) buys some of its product lines from outside vendors. In this case, it buys 200 (all root beer) from a vendor named Grandma's.

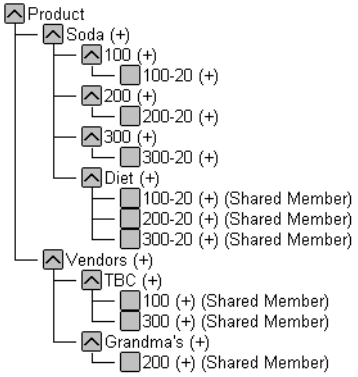


Figure 13-37: Members with Branches Shared at Different Generations

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

- "Creating Shared Members Using Level References" on page 13-40
- "Creating Shared Members Using Parent/Child References" on page 13-42

Creating Shared Members Using Level References

To create shared non-leaf members using the level references build method, first make sure that both primary and secondary roll-ups are specified in one record. You can specify as many secondary roll-ups as you want, as long as the roll-ups are all in one record.

Define the field type for the shared member's parent as duplicate level (DUPLEVEL). Then enter the level number. To create a shared member of the same generation, set the level number of the secondary roll-up to have the same

number of levels as the primary roll-up. While processing the data source, Hyperion Essbase creates a parent at the specified level and inserts the shared members under it.

For example, to share the product lines 100, 200, and 300 with a parent called Soda and two parents called TBC and Grandma's, use the sample data file and rules file shown in Figure 13-38. This data source and rules file only work if the Diet, TBC, and Grandma's members exist in the outline. The DUPLEVEL field is always created as a child of the dimension (that is, at generation 2), unless the named level field already exists in the outline.

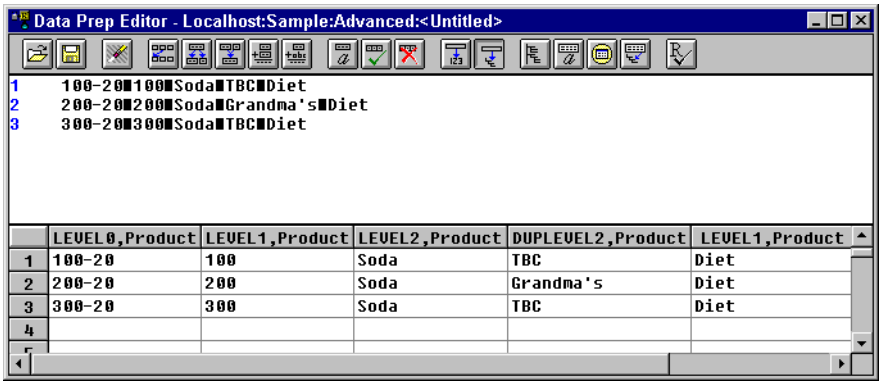


Figure 13-38: Level References Sample Rules File for Shared Members at Different Generations with Branches

This builds the tree in Figure 13-37.

Creating Shared Members Using Parent/Child References

To create shared members at the same generation using the parent/child references build type, define the PARENT and CHILD field types. Make sure that Do Not Share is not selected in the Dimension Build Settings page of the Dimension Build Settings dialog box so that Hyperion Essbase creates duplicate members as shared when it encounters them under a new parent.

The parent/child references build method is the most versatile for creating shared members. It does not have any restrictions on the position of the shared members in the outline, unlike the generation references and level references build methods.

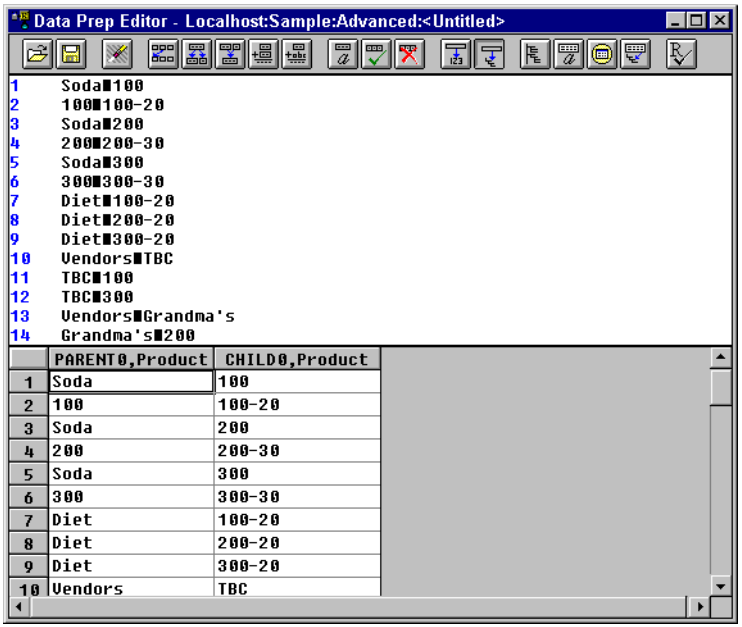


Figure 13-39: Parent/Child Sample Rules File for Shared Members at Different Generations with Branches

This builds the tree in Figure 13-37.

Building Multiple Roll-Ups Using Level References

To enable retrieving totals from multiple perspectives, you can also put shared members at different levels in the outline. Use the level references build method. The rules file, `LEVELMUL.RUL`, in Figure 13-40 specifies an example of build instructions for levels in the Product dimension.

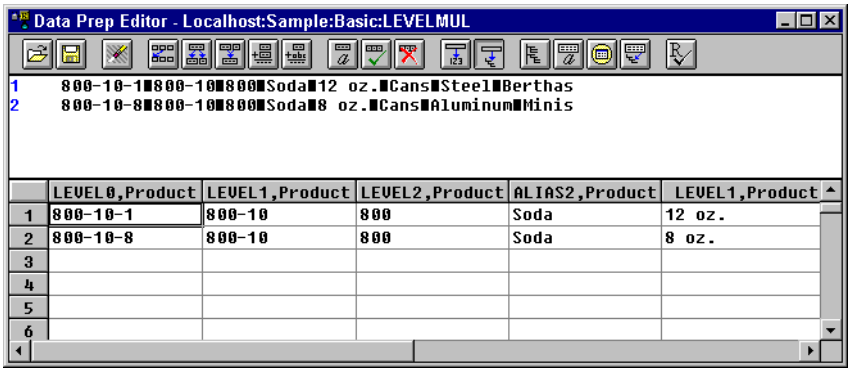


Figure 13-40: Rules File Fields Set to Build Multiple Roll-Ups Using Level References

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

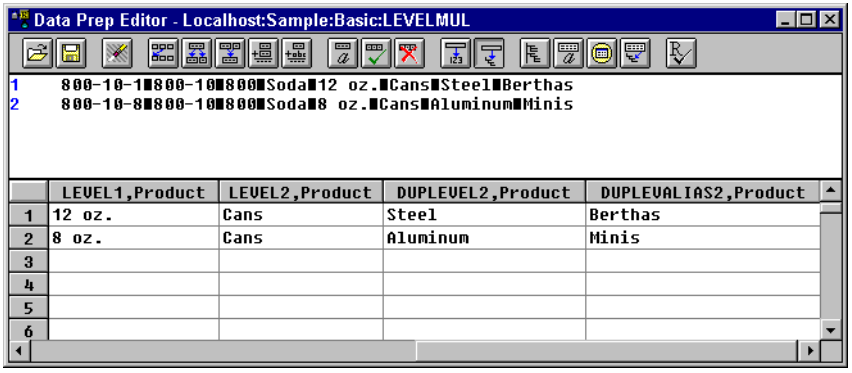


Figure 13-41: Scrolled Window

When you run the dimension build using the data in Figure 13-40, Hyperion Essbase builds the following member tree:

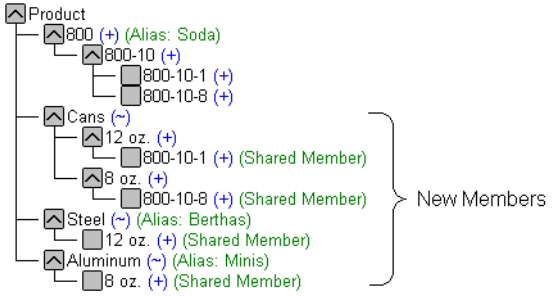


Figure 13-42: Multiple Roll-Ups

This example enables analysis not only by package type (Cans), but also by packaging material; for example, analysis comparing sales of aluminum and steel cans.

Since Product is a sparse dimension, you can use an alternative outline design to enable retrieval of the same information. Consider creating a multilevel attribute dimension for package type with Steel and Aluminum as level 0 members under Can. See “Analyzing Database Design” on page 5-10.

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

In many situations, the data for a dimension is in two or more data sources. If you are building dimensions from more than one data source and want to create multiple roll-ups, 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 selected so that Hyperion Essbase creates duplicate members as shared when it encounters them under a new parent.

For example, from the Product data source in Figure 13-44:

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

Figure 13-43: Soft Drinks Data Source

Hyperion Essbase builds the following tree:

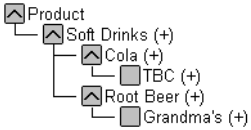


Figure 13-44: Soft Drinks Tree

A second data source relates the products to the vendors:

```
Vendor    TBC
Vendor    Grandma's
```

Figure 13-45: Second Shared Roll-Ups Data Source

To create a member tree where the products are shared by the Vendor member, load the second data source using the parent/child build method. Make sure that "Do Not Share" is not selected so that Hyperion Essbase creates duplicate members as shared when it encounters them under a new parent.

Hyperion Essbase builds the following tree:

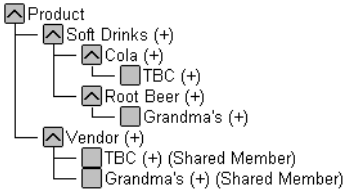


Figure 13-46: Shared Roll-Ups Tree

Security and Multi-User Considerations

Hyperion 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, Hyperion 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 17, “Managing Security at Global and User Levels.”

Chapter 14

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 13, “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 Data Prep Editor. This process includes defining new dimensions, specifying changes to existing dimensions, setting global options, and validating the dimension build rules.
- Create a dynamic reference in a rules file to a record in the data source that defines each field. This method enables you to use a single rules file with several different source files. To use this method, you must manipulate the date source to include this header record.

Validate the rules file and perform the dimension build. If you have problems validating the rules file or using it to build dimensions, this chapter also discusses debugging tips.

Use the BUILDDIM and INCBUILDDIM commands in ESSCMD to build dimensions dynamically. See the online *Technical Reference* in the `DOCS` directory for information about these commands. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Step 1: Defining Dimensions

➤ To define a dimension:

1. Name new dimensions and specify whether a standard dimension comes from the outline or a rules file.
2. Set the build type and the properties for new dimensions or change the properties of existing dimensions.

Naming New Dimensions

The processes for naming new standard dimensions and new attribute dimensions are different.

➤ To name a new dimension:

1. Select the application and database in the Application Desktop window in Hyperion Essbase Application Manager.
2. Click the Data Load Rules button,  .
3. Click New to open Data Prep Editor with a new rules file or Open to open an existing rules file.
4. Select View > Dimension Building fields or click the Dimension Build button,  , to make sure that Data Prep Editor displays dimension building fields and not data load fields.
5. Select Options > Dimension Build Settings to open the Dimension Build Settings dialog box. Select the Dimension Definition tab.

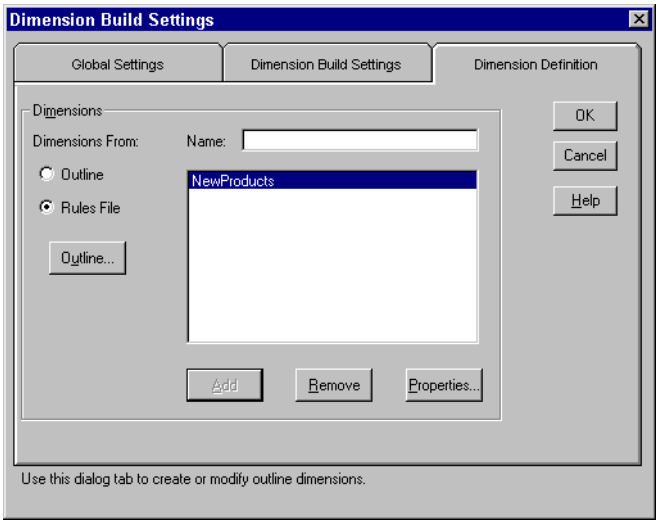


Figure 14-1: Dimension Definition Page

When Outline is selected, the large list box displays the names of all dimensions in the existing outline.

When Rules File is selected, the large list box displays the names of all new standard dimensions defined in the rules file. To see new attributes dimensions, you must follow the instructions in “Naming New Attribute Dimensions” on page 14-4.

Naming a New Standard Dimension

If you are naming a new attribute dimension to be associated with an existing standard dimension, proceed to “Naming New Attribute Dimensions” on page 14-4.

➤ To name a new standard dimension:

1. Select Rules File to indicate that the dimension is defined in the rules file.
2. Enter the name of the new dimension, such as `NewProducts`. See “Rules for Naming Applications and Databases” on page 7-12.
3. Click Add to add it to the end of the outline.

   If you are not also defining associated attribute dimensions, continue with setting the dimension properties. See “Setting Dimension Properties” on page 14-6.

Naming New Attribute Dimensions

➤ To create a new attribute dimension, the base dimension must already be defined in either the outline or the rules file. The base dimension must be a sparse dimension.

1. Select the base dimension in the list box and click Properties to open the Dimension Properties dialog box.
   - If the base dimension is defined in the outline, on the Dimension Definition page of the Dimension Build Settings dialog box, select Outline to display the base dimension name in the list box.
   - If the base dimension is defined in the rules file, on the Dimension Definition page of the Dimension Build Settings dialog box, select Rules to display the base dimension name in the list box

2. On the Dimension Properties page, click Attribute Dimensions to display the Define Attribute Dimensions dialog box.

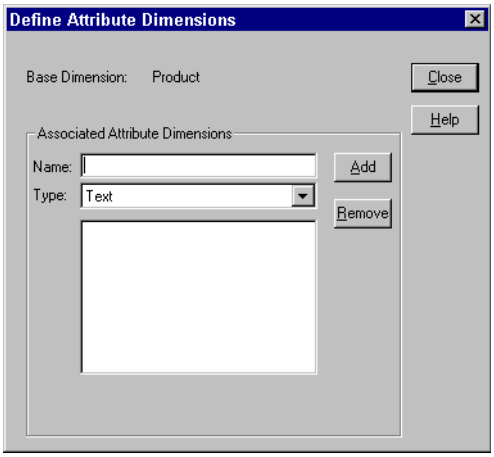


Figure 14-2: Defining Attribute Dimensions

The large list box lists the names of new attribute dimensions associated with the base dimension in the current rules file.

3. For each new attribute dimension that you define to associate with the base dimension:
   - Type the attribute dimension name; for example, Population.
   - Select the attribute dimension type; for example, Numeric.
   - Click Add.
4. To remove an attribute dimension from the list box:
   - Select the attribute dimension name.
   - Click Remove.
5. After adding all new attribute dimension names and specifying their types, click OK to close the dialog box and return to the Dimension Properties dialog box.
6. Click OK to close the Dimension Properties dialog box.

Setting Dimension Properties

➤ To set the properties of a standard dimension:

1. If the Dimension Build Settings dialog box is not open, select Options > Dimension Build Settings to open it.
   - Select the Dimension Definition page.
   - If the dimension exists in the outline, click the Outline option to display the names of the dimensions in the outline. If the list box is empty, click the Outline button to associate the dimension build rules file with an outline and display the dimensions in the list box.
   - If the dimension is new, click Rules File. The list box displays the new dimensions that you named.
2. Click the dimension name in the list box.
3. Click Properties to open the Dimension Properties dialog box.

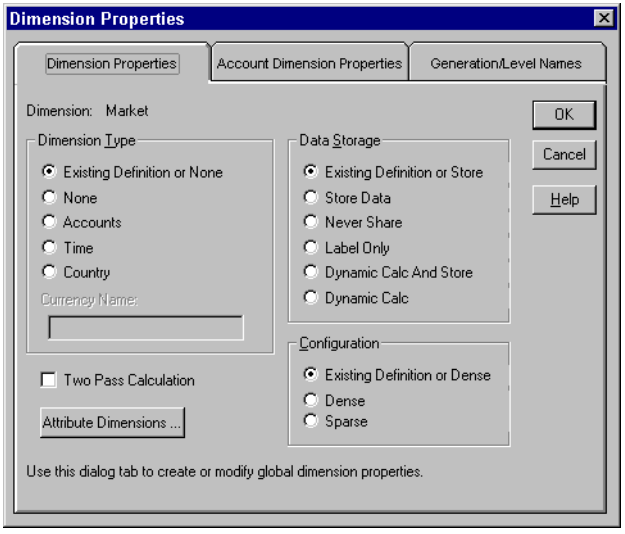


Figure 14-3: Dimension Properties Page

- Set the Dimension Type.
- Select the Data Storage property.

- Select a Dense or Sparse configuration. Base dimensions must be set as sparse.
- If you are not sure what settings to use, click Help.

4. For an accounts dimension, click the Account Dimension Properties tab.
   - Set the accounts dimension properties and click OK.
   - If you are not sure what settings to use, click Help.

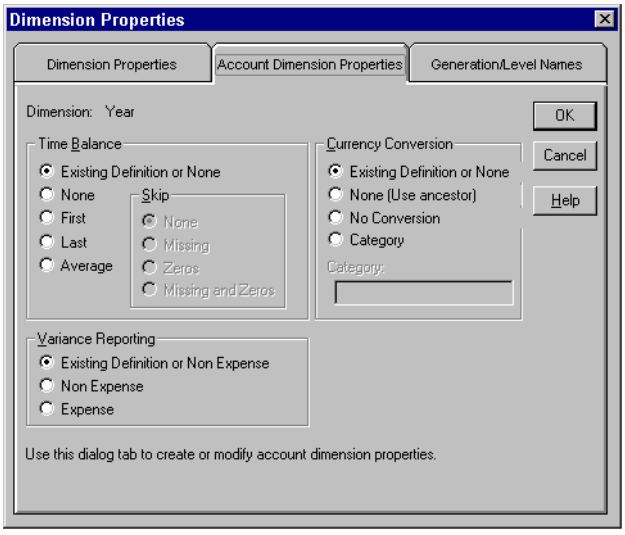


Figure 14-4: Account Dimension Properties Page

5. To name the generations and levels in the current dimension, select the Generation/Level Names tab.
   - Set the generation/level names and click OK.
   - If you are not sure which settings to use, click Help.

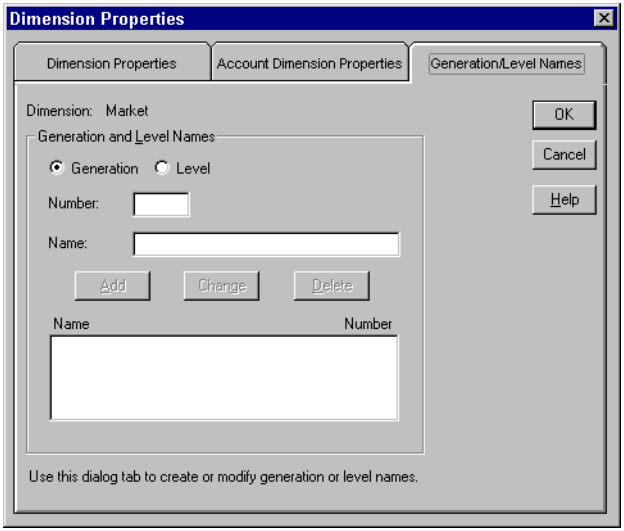


Figure 14-5: Generation/Level Names Page

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: Choosing the Build Method

➤ To specify the build method for the rules file:

1. If it is not showing, click the Dimension Build Settings tab of the Dimension Build Settings dialog box.

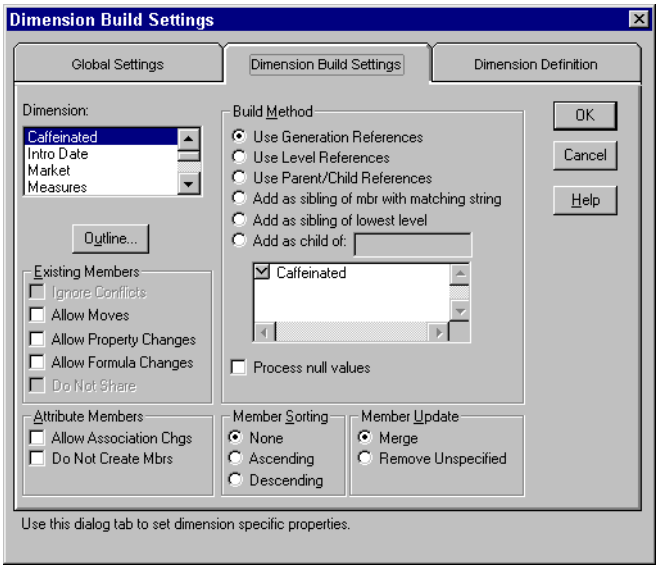


Figure 14-6: Dimension Build Settings Page

2. Select the dimension from the Dimension list. If the list is empty, click Outline to associate the dimension build rules file with an outline. The list includes new standard dimensions defined in the rules file.
3. Select the build method from the Build Method box. If you are not sure which method to use, see “Introduction to Build Methods” on page 13-3.

The following build methods require that you specify additional information:

Build Method	What to Specify
Use generation references	Whether to use null processing. See “Null Processing with Generation References” on page 13-7.
Use level references	Whether to use null processing. See “Null Processing with Level References” on page 13-10.
Add as child of	The parent to which the new members are added.

Step 3: Specifying Changes to Dimensions

➤ To specify the changes that you want Hyperion Essbase to make to dimensions in the outline:

1. On the Dimension Build Settings page of the Dimension Build Settings dialog box, select the dimension from the Dimension list.
2. Select the types of changes you will allow to existing members of the selected dimension in the outline.

Select 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.
Allow Moves	Allow a member and its descendants to be moved to a new parent in the same dimension. Hyperion Essbase cannot move the member to itself or to another member below it in the tree. Use Allow Moves to reorganize primary members in the outline to match the data source. To reorganize shared members, use Outline Editor.
Allow Property Changes	Allow changes to the properties, UDAs, and aliases of existing members.

Select the option...	When you want to...
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 select this box, each time a member is repeated with another parent, it is created as a shared member.

3. To sort the members of a dimension after building it, select either Ascending (A to Z) or Descending order (Z to A). To leave the members unsorted, select None.
4. By default, Hyperion Essbase merges new members found in the data source into the dimension. To remove existing members if Hyperion Essbase does not encounter them in the data source, select Remove Unspecified.
5. If the rules file works with attribute source data, select the attribute dimension and select the types of changes to allow:

Select the option...	When you want to...
Allow Association Chgs	Allow an existing association to be changed. For example, if the source data shows the Ounces attribute for product 100-10 as 8 and the existing outline shows the attribute as 12, Hyperion Essbase associates product 100-10 with the attribute 8.
Do Not Create Mbrs	Prevent creation of new members of the attribute dimension. For example, if the source data shows the Ounces attribute for product 100-10 as 8 and the Ounces attribute dimension does not include the member 8, Hyperion Essbase does not add the member 8 to the Ounces dimension.

6. Click OK.

Step 4: Setting Rules File Field Types

Each field defines a source data column that becomes a member in the outline, a property of a member, or information that helps to define an association; for example, associating an alias with a member or an attribute with a base dimension member.

Setting the field type tells Hyperion Essbase what kind of field to expect, such as a generation field or an alias field. At the same time, you specify the dimension for the member and its generation or level number.

To set the field types in Data Prep Editor, you must set Data Prep Editor to dimension building mode and you must open the data source.

- Select View > Dimension Building Fields or click the Dimension Build button,  , to ensure that Data Prep Editor is in dimension building mode.
- Select File > Open Data File or File > Open SQL, whichever is appropriate, to specify the source location and open the data source. For more details, see “Opening a Data Source” on page 21-5.

  As shown in Figure 14-7, Data Prep Editor displays the data source in the upper half of the window and rules fields in the lower half of the window.

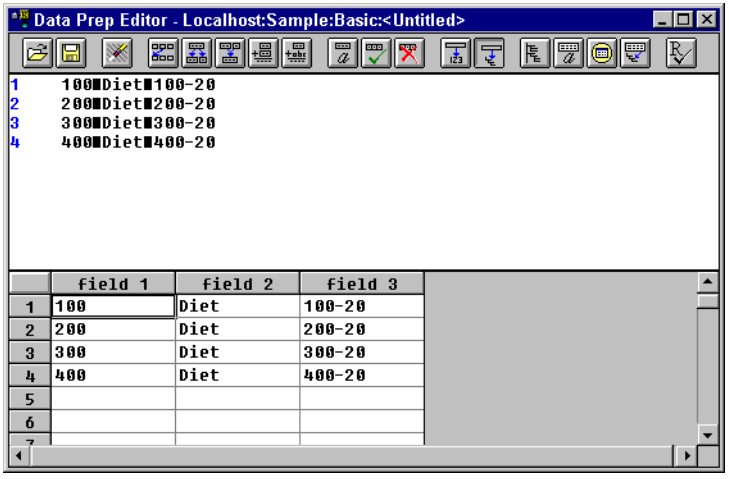


Figure 14-7: Data Prep Editor

If desired, you can customize Data Prep Editor. For example, you can hide the raw data or maximize the window, to set a better view of the rules fields. See "Customizing the Data Prep Editor" on page 21-4.

Setting Field Information

This section describes how to define the field type for each field in the rules file. Many of the details for each field depend on the nature of the data source and the build method to be used. See Chapter 13, "Introducing Dynamic Dimension Building," for explanations and examples of how the fields should be defined to build or modify outlines for various situations.

To map the rules file fields to the source data, you may need to manipulate how the data is used. For example, you may need to create a member name from two source data fields, or you may need to ignore a source data field. For information on mapping and manipulating fields, see "Manipulating Fields Using a Rules File" on page 22-1.

➤ To set field types:

1. In Data Prep Editor window, select the field for which you want the field type set.
2. Select Field > Properties or click the Field Properties button,  , to open the Field Properties dialog box.

3. Select the Dimension Building Properties tab.

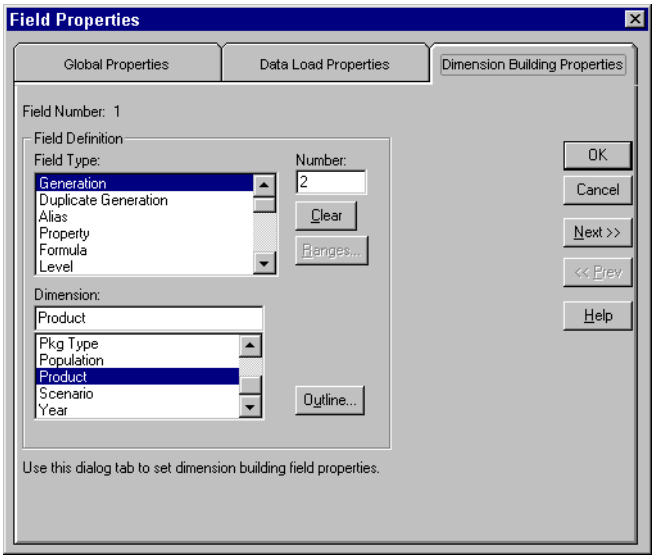


Figure 14-8: Dimension Building Properties Page

4. If the Dimension list is empty, click the Outline button to associate the rules file with an outline.

5. Select the field type from the Field Type list box. Table 14-1 lists valid field types for each build method.

Table 14-1: Defining Field Types in Rules Files

Field Type	What the Field Contains	Valid Build Methods
Alias	An alias	Generation, level, and parent/child references
Property	A member property	
Formula	A formula	
Currency name	A currency name	
Currency category	A currency category	
UDA	A user-defined attribute	
Attribute Parent	In an attribute dimension, the name of the parent member for the attribute member in the next field	
The name of a specific attribute dimension	A member of the specified attribute dimension. This member will be associated with the specified generation or level of the selected base dimension.	
Generation	Name of a member in a generation	Generation references
Duplicate generation	A member that is shared by more than one parent	
Duplicate generation alias	Alias for the new parent of the shared member as the member is created	
Level	Name of member in a level	Level references
Duplicate level	Name of member that has duplicate parents; that is, a member that is shared by more than one parent	
Duplicate level alias	Alias for the new parent of the shared member as the member is created	
Parent	Name of a parent	Parent/child reference
Child	Name of a child	

6. Enter the generation or level number in the Number text box.

   - If Field Type is the name of an attribute dimension, the generation or level number must correspond to the generation or level of the associated base member in the outline. For example, the 3 in OUNCES3,PRODUCT shows that the data values in the field are members of the Ounces attribute dimension associated with the third generation member of the Product dimension in the same source data record.
   - If Field Type is parent or child, enter 0 (zero) in the Number text box.
   - If Field Type is not the name of an attribute dimension nor parent nor child, the generation or level number must correspond to the generation or level of the member in the outline for which the field provides values. For example, the 3 in GEN3,PRODUCT shows that the data values in the field are third generation members of the Product dimension. The 2 in ALIAS2,POPULATION shows that the data values in the field are associated with the second generation or level member of the Population dimension.

As described in the following table, how you assign and sequence fields in the rules file can vary depending if the number is for a level or generation build.

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, Hyperion Essbase rejects the record unless you select Allow Moves. See “Step 3: Specifying Changes to Dimensions” on page 14-10. • Group level fields sequentially within a dimension. • Put the fields for each roll-up in sequential order. • Use a single record to describe the primary and secondary roll-ups. • Put attribute association fields after the base field with which they are associated and specify the level number of the associated base dimension member; for example: `LEVEL3, PRODUCT  OUNCES3,PRODUCT  LEVEL2,PRODUCT`

Generation	• If GEN numbers don’t start at 2, the first member in the specified generation must exist in the outline.
	• GEN numbers must form a contiguous range. For example, if GEN 3 and GEN 5 exist, you must also define GEN 4.
	• Put DUPGEN fields immediately after GEN fields.
	• Put DUPGENALIAS fields immediately after DUPGEN fields.
	• Group GEN fields sequentially within a dimension; for example:
	`GEN2,PRODUCT  GEN3,PRODUCT  GEN4,PRODUCT`
	• Put attribute association fields after the base field with which they are associated and specify the generation number of the associated base dimension member; for example:
	`GEN2, PRODUCT  GEN3,PRODUCT  OUNCES3,PRODUCT`

7. In the Dimension box, enter the dimension for which the field provides values, or select the dimension name from the Dimension list. For attribute associations, select the base dimension from the Dimension list.

8. To define ranges for members of the numeric attribute dimension specified in the Field Type list box, click Ranges. Otherwise, proceed to step 14.

   Data Prep editor displays the Numeric Range Rules dialog box.

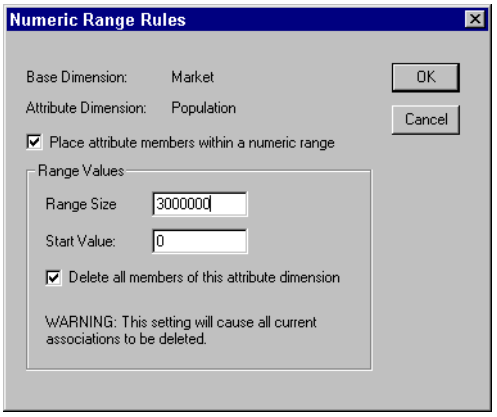


Figure 14-9: Defining Range Size for Numeric Attribute Dimensions

   For information about using attribute dimension members to represent ranges of base member values, see “Assigning the Names of Members of Numeric Attribute Dimensions to Ranges of Values” on page 10-23 and “Working With Numeric Ranges” on page 13-25.

9. To enable automatic range building for the specified numeric attribute dimension members, click “Place attribute members within a member range.”

10. In the Range Size text box, enter the numeric value of the range size for each member; for example, 3000000.

11. In the Start Value text box, enter a numeric value for the name of one member of the numeric attribute dimension.

    This member becomes the pivot point upon which Hyperion Essbase uses the range size to create other range members above and below the specified start value. Hyperion Essbase uses the start value only when it builds an attribute dimension and associates its members to members of a base dimension in the same build operation.

Assume, for example, that you set the range size as 10 and the start value as 5. As it processes the data source, Hyperion Essbase may build the following members of the numeric attribute dimension: 5-, 5, 15, 25, and so on.

Note: Although you enter the names of negative numeric attributes with the minus sign before the number, for example -5, Hyperion Essbase creates the member name with the minus sign after the number, for example 5-.

The start value can be a positive or negative whole number, zero, or a decimal value. To enter a negative number, type the minus sign in front of the number; for example, -15.

Tip: If you want a numeric range member named 0, specify 0 as the start value. For example, if the range size is 3000000, Hyperion Essbase builds the following members of the numeric dimension: 0, 3000000, 6000000, and so on.

12. To remove and re-create all members of the specified attribute dimension and reassociate them with the members of the base dimension, select "Delete all members of this attribute dimension."

CAUTION: All base member associations with the attribute dimension are lost. To enable the dimension build to re-create the associations, the source data must include all members of the base dimension that you want to be associated with members of this attribute dimension.

---

13. Click OK to close the Numeric Range Rules dialog box.
14. To move to the next field, click Next.
15. When you are finished setting the field types, click OK.

Note: If needed, move the fields to the required locations. The required location of fields depends on the build method and what you want to achieve through the dimension build operation. For specific requirements, see the sixth step under "Setting Field Information" on page 14-13 and "Working with Multilevel Attribute Dimensions" on page 13-22.

Step 5: Setting Global Options

Global options affect *all* dimensions in the rules file. Generally, you build one dimension per rules file. The global build options include:

- Whether to configure dimensions as dense or sparse automatically or to use the dense/sparse configuration defined in the outline or rules file
- Which alias table to update
- How to combine field select/reject criteria between fields

➤ To set global build options:

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

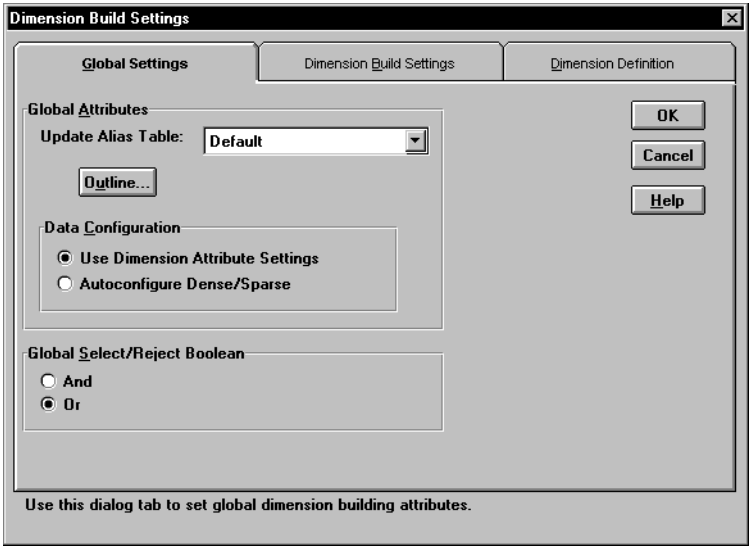


Figure 14-10: Global Settings Page

2. Select which alias table to update with new aliases from the data source. If you do not specify an alias table, Hyperion Essbase updates the Default table.
3. Select either:
   - The Use Dimension Property Settings to keep using the current data configuration or use the one specified in the rules file.
   - The Autoconfigure Dense/Sparse to let Hyperion Essbase determine the data configuration automatically. For more information on dense and sparse dimensions, see “Sparse and Dense Dimensions” on page 4-2.

   When you change the configuration settings, Hyperion Essbase restructures the database.
4. Select either And or Or to determine how Hyperion Essbase combines select and reject criteria. For more information, see “Defining Multiple Select and Reject Criteria” on page 21-18.
5. Click OK to save the changes.

Step 6: Validating Dimension Build Rules

➤ To validate a rules file, make sure that the rules file is open and associated with an outline. If you’re building dimensions by altering the data source (using dynamic references), open the data source.

1. Select View > Dimension Building Fields or click the Dimension Build button,  , to make sure that Data Prep Editor is in dimension building mode.

2. Select Options > Validate or click the Validate Rules button,  , to validate the rules file against the outline. When Hyperion Essbase finishes the validation, the Validate Rules dialog box displays.

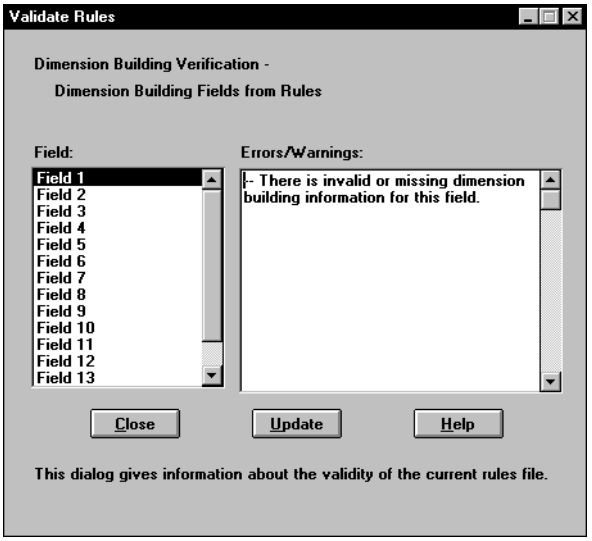


Figure 14-11: Validate Rules Dialog Box

If the rules file is correct, you can use it perform a dimension build. For more information, see “Performing Dimension Builds” on page 14-26.

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 14-11, for example, Field 1 is invalid.

   Make sure that the field name is valid.

   - Are the reference numbers sequential?
   - Are there repeated generations?
   - Is the field type valid for the build method?
   - Are the fields in correct order?

- Does the child field have a parent field?
- Do all dimension names exist in the outline or the rules file?

4. Validate the file again. Return to step 1.

Manipulating the Data Source

You can also build dimensions or change the properties of existing members in a dimension by adding information to the data source. You can add:

- Header information to specify the dimension and field types
- Member codes that set member properties

Using Dynamic References

You can dynamically build dimensions by adding header information to the top of the data source and specifying the location of the header information in the rules file as a dynamic reference. Figure 14-12 contains an example of a header record.

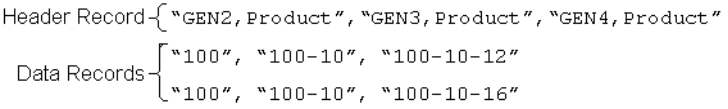


Figure 14-12: Header Record

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

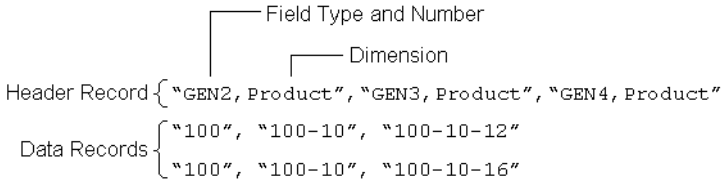


Figure 14-13: Header Record with Three Field Definitions

If the file delimiter is a comma, enclose each field definition in quotation marks ("").

After you set the header record in the data source, you must use a dynamic reference to specify the location of the header record in the rules file. If a rules file contains a dynamic reference, Hyperion 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
- PROPERTY
- ALIAS
- FORMULA
- CURNAME
- CURCAT
- UDA
- ATTRPARENT
- The name of an attribute dimension.

For each field type that you set, you must also enter a field number. When the field type is the name of an attribute dimension, the field number cannot be greater than 9. For more information on field numbers, see “Step 4: Setting Rules File Field Types” on page 14-12.

Setting Member Properties

You can modify the properties of both new and existing members during a dimension build by setting the properties directly in the data source. Put properties in the field directly following the field they modify. For example, to specify that the Margin% member not roll up into its parent and not be shared, use the following data source:

```
Margin%  Margin%  Sales  ~ N
```

Set the field type for the properties field to Property. To set the field type to property, see “Manipulating the Data Source” on page 14-23.

The following table lists all member codes used to assign properties to members in the data source.

Code	Description
%	Express as a percentage of the current total in a consolidation
*	Multiply by the current total in a consolidation
+	Add to the current total in a consolidation
-	Subtract from the current total in a consolidation
/	Divide by the current total in a consolidation
~	Exclude from the consolidation
A	Average time balance item (applies to accounts dimensions only)
B	Exclude data values of zero or `#MISSING` in the time balance (applies to accounts dimensions only)
E	Expense item (applies to accounts dimensions only)
F	First time balance item (applies to accounts dimensions only)
L	Last time balance item (applies to accounts dimensions only)
M	Exclude data values of `#MISSING` from the time balance (applies to accounts dimensions only)
N	Never allow data sharing
O	Label only (store no data)
T	Require a two-pass calculation (applies to accounts dimensions only)
V	Create as Dynamic Calc And Store
X	Create as Dynamic Calc
Z	Exclude data values of zero from the time balance (applies to accounts dimensions only)

Performing Dimension Builds

When you have a valid dimension build rules file, you can 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 23, “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 the `DOCS` directory for information about these commands. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

- Using Outline Editor. For more information, see “Updating Dimensions in Outline Editor” on page 14-26.

Updating Dimensions in Outline Editor

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

➤ To update dimensions using Outline Editor:

1. Open the outline in Outline Editor. See “Opening Outlines” on page 8-2.

2. Select File > Update Outline to open the Outline Update dialog box.

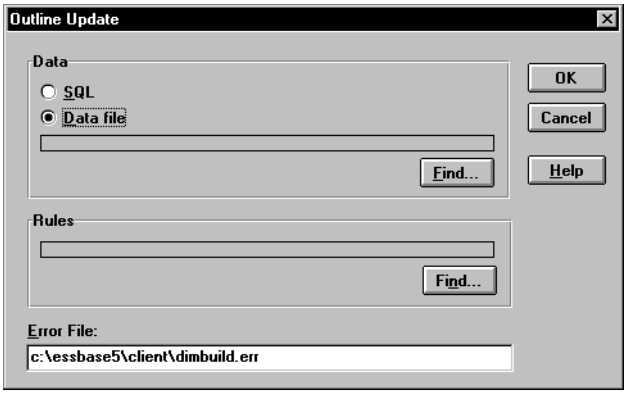


Figure 14-14: Outline Update Dialog Box

3. Select the kind of data source by choosing SQL or Data File.
4. If you chose SQL, all of the information you need is stored in the rules file. Skip to step 7.
5. If you select Data File, click Find to select the data source. The Open Server Data File Object dialog box displays.

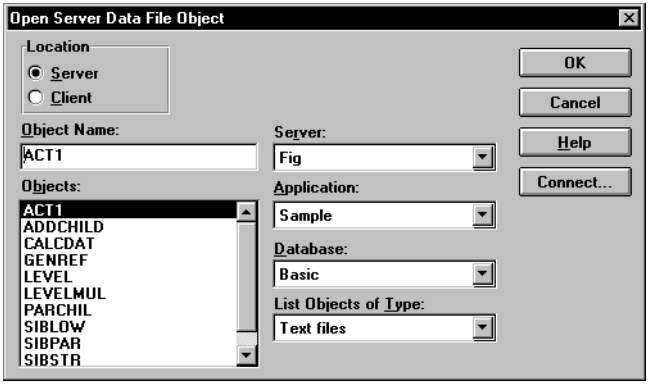


Figure 14-15: Open Server Data File Object Dialog Box

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

   If you select Server, the data source must reside in the database directory under `\ESSBASE\APP\`*`application_name`*`\`*`database_name`*, where *application_name* and *database_name* represent the name of your application and database. Type the name of the data source in the Object Name text box or select it from the Objects list box.

   If you select Client, the data source may reside in either the application or database directory under `\ESSBASE\CLIENT` or on the drives accessible from the client file system. Click File System to select a data source from a standard Open Client Data Files dialog box. Select the data source to open.

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

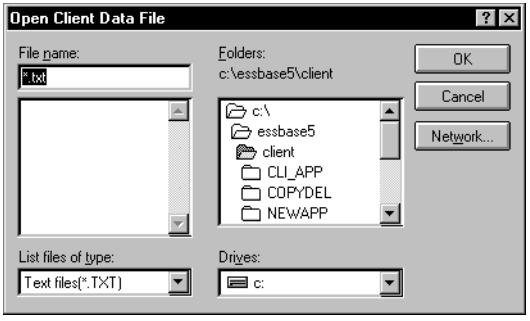


Figure 14-16: Open Client Data File Dialog Box

7. Select the dimension build rules file to load by clicking Find and then selecting the rules file in the Open Server Data File Object or Open Client Data File dialog box as described in step 5.
8. Click OK. Hyperion 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. Perform the dimension build using Outline Editor.
3. Move the updated outline back to the server using standard Windows tools.

Debugging Dimension Builds

Hyperion Essbase displays the results of dimension builds in the Dimension Build Completed dialog box as shown in Figure 14-17.

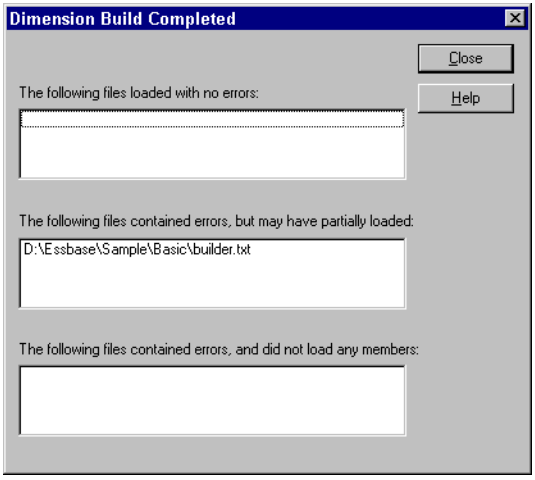


Figure 14-17: Dimension Build Completed Dialog Box

The Dimension Build dialog box displays the results in three different windows.

- The top window lists files that loaded completely.
- The middle window lists files that may have partially loaded.
- The bottom window list files that did not load at all.

If errors occurred during dimension building, Hyperion Essbase logs them in the `\ESSBASE\CLIENT\DIMBUILD.ERR` file.

➤ To find data errors and correct 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 14-18 is an example of an error log that was generated during a dimension build.

```
\\Record #1 - Child Member 600 with no parent (3331)
600     600-10
\\Record #1 - Child Member 600-10 with no parent (3331)
600     600-10
\\Record #2 - Child Member 600 with no parent (3331)
600     600-20
\\Record #2 - Child Member 600-20 with no parent (3331)
600     600-20
\\Record #3 - Child Member 600 with no parent (3331)
600     600-30
\\Record #3 - Child Member 600-30 with no parent (3331)
600     600-30
```

Figure 14-18: Error Log

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

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

   - Can you validate the rules file? See "Step 6: Validating Dimension Build Rules" on page 14-21.
   - Is the data source available? See Chapter 24, "Debugging and Optimizing Data Loads."
   - Is the server available? See Chapter 24, "Debugging and Optimizing Data Loads."

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

4. Reprocess the records that failed.

How Hyperion Essbase Builds Dimensions

Sometimes, you can track down problems with dimension builds by understanding how Hyperion Essbase initializes the rules file and processes the data source. Hyperion 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 Hyperion 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 15

Estimating Disk and Memory Requirements for a Database

This chapter helps you estimate disk and memory requirements and tells you about database settings, including how to change settings. This chapter contains the following sections:

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

For information about specifying and changing Hyperion Essbase kernel settings, see Chapter 41, "Specifying Hyperion Essbase Kernel Settings."

Understanding How Hyperion Essbase Stores Data

This topic describes the units of storage that you need to know about in order to size a database. The discussion assumes that you are familiar with basic concepts of the Hyperion Essbase kernel. See Chapter 40, "Introducing the Hyperion Essbase Kernel," to learn about the Hyperion Essbase kernel.

A Hyperion Essbase database consists of many different components. In addition to an outline file and a data file, Hyperion Essbase uses various files and memory structures to manage data storage, calculation, and retrieval operations. Table 15-1 describes the major components that you must consider when you estimate the disk and memory requirements of a database.

Table 15-1: Storage Units Relevant to Calculation of Disk and Memory Requirements

Storage Unit	Description	Disk	Memory
Outline	A structure that defines all elements of a database. The number of members in an outline determine the size of the outline.	✓	✓
Data files	A group of files in which Hyperion Essbase stores data values in data blocks. Named `ESSxxxxx.PAG`, where *xxxxx* is a number. Hyperion Essbase increments the number, starting with `ESS00001.PAG`, on each disk volume.	✓	
Data block	A subdivision of a data file. A multidimensional array that represents all cells of all dense dimensions relative to a particular intersection of sparse dimensions.	✓	✓
Index files	A group of files that Hyperion Essbase uses to retrieve data blocks from data files. Named `ESSxxxxx.IND`, where *xxxxx* is a number. Hyperion Essbase increments the number, starting with `ESS00001.IND`, on each disk volume	✓	✓
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. Hyperion Essbase allocates this memory at startup of the database.		✓
Data file cache	A buffer in memory that holds data files. Hyperion Essbase allocates this memory during data load, calculation, and retrieval operations, as needed.		✓
Data cache	A buffer in memory that holds data blocks. Hyperion Essbase allocates this memory during data load, calculation, and retrieval operations, as needed.		✓
Calculator cache	A buffer in memory that Hyperion Essbase uses to create and track data blocks during calculation operations.		✓

When you load data to a database, Hyperion Essbase creates and populates index and data files on the disk. When you access data, Hyperion Essbase executes an index search and loads index information and data into memory buffers called caches.

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

The calculator cache is an optional cache used for calculations. For information on the calculator cache, see "Setting Memory Cache Sizes" on page 33-14.

Determining Disk Space Requirements

Hyperion Essbase requires disk space for basic server software and for each database. Each database comprises many files including the data files, the index files, the overhead files and the outline file. The following subtopics enable you to estimate the size of each of these database components. After you determine each individual component size, see "Estimating Disk Space for a Database" on page 15-12, to approximate the amount of disk space needed for an entire database.

General Disk Space Requirements for a Database

This discussion begins with guidelines for calculating adequate disk space for a single database. To estimate overall disk space requirements, see "Total Disk Requirement" on page 15-13.

The total amount of disk space required for a database depends on the following factors:

- The database design, including the number of sparse and dense dimensions and how the dimensions are populated with data.
- The number of existing data blocks in the database. An *existing block* contains at least one non-missing value.
- The number of data values (cells) that populate the data blocks.
- The number of members in the database outline.
- Whether data blocks are compressed, and if they are compressed, the compression scheme used. See "Data Compression" on page 40-8.

- The Isolation Level settings. If you select *committed* access or set the Commit Blocks and Commit Rows settings at 0 under *uncommitted* access (the default isolation level), you need to allow space on the disk for double the size of the database. Hyperion Essbase helps ensure data integrity and enforces transaction semantics by maintaining redundant data. See “About Isolation Levels” on page 42-2.
- Whether you are using Hyperion Essbase partitioning to store a replicated partition. See “Considerations When Using Partitioning” on page 15-12.
- Whether you are using stored linked reporting objects. See “Considerations When Using Linked Reporting Objects” on page 15-13.
- The number of stored members. Dynamic Calc, label only, and shared members and attribute dimensions and their members are not stored and thus do not require storage space.The term *stored member*, which is used later in this chapter, excludes all such members. Dynamic Calc And Store members do use storage space and are designated as stored members.

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

Note: The database sizing calculations that follow assume an ideal scenario with an optimum database design and unlimited disk space.

Determining the Expanded Block Size of Stored Data

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 stored data block, follow these steps:

1. Multiply the number of stored members (including Dynamic Calc And Store members) in each dense dimension together.

   For example, the Sample Basic database contains the following:

   - Twelve stored members from the Year dimension
   - Eight stored members from the Measures dimension
   - Two stored members from the Scenario dimension

   Thus, the potential population of a data block in Sample Basic is:

   ```
   12 x 8 x 2 = 192 data cells
   ```

2. Multiply the figure derived in Step 1 by the size of each cell (8 bytes); for example:

   ```
   192 data cells x 8 bytes = 1,536 bytes
   ```

Estimating the Size of Compressed Data Blocks

The use of compression (whether it is used and, if it is used, what type is used) affects the actual disk space consumed by a data file. For more information about data compression, see “Data Compression” on page 40-8.

The following formula provides a rough estimate of compressed block size. It assumes that bitmap compression is enabled.

```
Compressed block size = Expanded block size x Block density
```

For example, assume an expanded block size of 1,536 bytes and a block density of 25%:

```
(1,536 bytes x .25) = 384 bytes
```

To determine block density, you can load the database and in Hyperion Essbase Application Manager, select Database > Information, select the Statistics tab, and look at the Block Density label. If you want to estimate block density prior to loading data, estimate the ratio of existing data values to potential data values.

Note: Actual block density varies widely from block to block. The calculations in this discussion are only for estimation purposes.

Estimating the Number of Data Blocks

To estimate the number of data blocks, you must consider the potential number of data blocks and the density of the database.

Determining the Potential Number of Data Blocks

When you determine the potential number of data blocks, you assume that there are data values for all combinations of stored members; for example, for Sample Basic, you assume that all products are sold in all markets. Stored members include neither attribute dimensions and their members nor members tagged as label only, shared, or Dynamic Calc.

To determine the potential, or maximum, *number* of blocks, multiply the number of stored members in each sparse dimension together.

For example, the Sample Basic database contains the following:

- 19 stored members from the Product dimension
- 25 stored members from the Market dimension

Thus, the Sample Basic database has:

```
19 x 25 = 475 potential data blocks.
```

Estimating the Actual Number of Data Blocks

Product and Market are sparse dimensions because not every product is sold in every market. Therefore, the actual number of blocks is far less than the potential number of data blocks.

If the database is already loaded, you can see the existing number of blocks and the potential number of blocks on the Statistics page of the Database Information dialog box of Hyperion Essbase Application Manager. Instead of estimating a number, you can use the existing number of data blocks in later calculations.

It is very difficult to determine the size of a database without loading the database. If you must determine size before loading the database, estimate the percentage of cells that contain data values and multiply this percentage against the potential number of data blocks. The percentage can vary based on the sparsity or density of the data blocks. This guideline provides only a rough estimate.

The following examples assume 100,000,000 potential data blocks:

- Extremely sparse: Only 5 percent of potential data cells exist.

  ```
  .05 (estimated percentage) x 100,000,000 (potential data
  blocks) = 5,000,000 data blocks
  ```

- Sparse: 15 percent of potential data cells exist.

  ```
  .15 (estimated percentage) x 100,000,000 (potential data
  blocks) = 15,000,000 data blocks
  ```

- Dense: 50 percent of potential data cells exist.

  ```
  .50 (estimated percentage) x 100,000,000 (potential data
  blocks) = 50,000,000 data blocks
  ```


Estimating the Size of the Compressed Data Files

To estimate the space required to store the data files (`ESSxxxxx.PAG`), multiply the compressed block size by the number of blocks. For this example and the remaining examples, consider the number of data blocks from the previous example calculated at a density of 15%.

For example:

```
384 (compressed block size) x 15,000,000 (number of data blocks)
= 5,760,000,000 bytes
```

Note: If compression is not used, substitute the expanded block size for the compressed block size in this formula.

Estimating the Size of the Index

The following formula helps you estimate the total size of the index, including all index files. The formula provides only a rough estimation.

```
Number of blocks x 112 bytes
```

This formula is based on the size of an index entry, which is 112 bytes.

For example, assuming a database with 15,000,000 blocks, you should allow about 1,680,000,000 bytes for the entire Hyperion Essbase index:

```
15,000,000 bytes x 112 = 1,680,000,000 bytes
```

If the database is loaded, select Database > Information and look at the Files tab to determine the actual size of the index.

Estimating Fixed-Size Overhead

The following subtopics show you how to calculate fixed-size overhead. You should use one of two methods of calculation, depending on whether the database uses bitmap compression, run-length encoding (RLE), or no compression.

Bitmap Compression

Begin the calculation of fixed-size overhead for a database that uses bitmap compression by using this formula:

```
((Expanded block size in bytes / 64) + 72) x Number of blocks
```

The formula may be explained as follows:

- Expanded block size—Although this subtopic assumes that you have compression enabled, the formula starts with the expanded block size. An *expanded block* is a non-compressed block. To estimate expanded block size, see “Determining the Expanded Block Size of Stored Data” on page 15-5.
- 64—The compression bitmap uses one bit for each cell in a block. You divide the expanded block size by 8 to get the number of cells; the quotient equals the number of bits in the bitmap. You divide this value by 8 to get the number of bytes; hence we divide expanded block size by 64 to obtain the bitmap size (fixed-size overhead) for each block.
- 72—The block header size is 72 bytes.

Then, round the value produced by the formula up to the nearest multiple of eight.

For example, assume the expanded block size is 4,802 bytes and that there are 15,000,000 blocks.

1. Calculate the formula:

   ```
   (4,802 / 64) + 72 = 147.03 bytes
   ```

2. Round the result up to the nearest multiple of eight.

a. Divide the result by 8	147.03 / 8 = 18.38
b. Use the whole number only	18
c. Add 1	18 + 1 = 19
d. Multiply by 8	19 x 8 = 152

   The result is 152 bytes per block.

Finally, calculate the fixed-size overhead for the entire database by multiplying the overhead per block by the number of blocks:

Bitmap overhead using bitmap compression =

```
152 x 15,000,000 bytes = 2,280,000,000 bytes
```


Run-Length Encoding (RLE) or No Compression

To calculate the fixed-size overhead for a database that uses RLE or no compression, use the total amount of overhead per block, 72 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.

Assuming 15,000,000 blocks, bitmap overhead using RLE or no compression =

```
72 x 15,000,000 bytes = 1,080,000,000 bytes
```

Calculating a Fragmentation Allowance

If you are using bitmap or RLE compression, a certain amount of fragmentation occurs. The amount of fragmentation is based on individual database and operating system configurations and cannot be precisely predicted.

As a rough estimate, calculate 20% of the compressed database size. For example, a compressed database size of 5,769,000,000 bytes produces the following calculation:

```
5,769,000,000 bytes (database size) x .2 = 1,152,000,000 bytes
```

Calculating a Data Recovery Area

For recovery purposes, Hyperion Essbase maintains a data recovery area on the disk. The size of this area increases until the database is restructured. To calculate the size of the data recovery area, multiply by 2 the sum of the sizes of the index, the overhead, the compressed data blocks, and the fragmentation allowance.

```
(Index size + Fixed size overhead
+ Size of compressed data blocks
+ Fragmentation allowance) x 2
```

Estimating the Size of the Outline

Factors that affect the size of an outline include the following:

- The number of members in the outline
- The length, in characters, of member names and aliases
- The number of members in each base dimension
- The number of members in each attribute dimension

First, estimate the main area of the outline by multiplying the number of members by an estimated number of bytes between 350 and 450. If the database includes few aliases or very short aliases and short member names, use a smaller number within the range. If you know that the names or aliases are very long, use a larger number within the range. (The maximum size for a member name and for an alias is 80 characters.) The following example uses a number in the middle of the range and assumes the outline has 26,000 members. This formula provides only a rough estimate.

```
400 bytes x 26,000 members = 10,400,000 bytes
```

Then, if the outline includes attribute dimensions, you must also calculate an area that stores attribute association information. Calculate the size of this area for each base dimension and add the sum of these areas to the basic outline size you already estimated.

To calculate the attribute area in an outline for a base dimension in bytes, multiply the number of members of the base dimension by the sum of the number of members of all attribute dimensions associated with the base dimension, and then divide by 8.

For example, assume that the base dimension Product (with 23,000 members) has two attribute dimensions associated with it: Ounces (with 20 members) and Pkg Type (with 50 members). The calculation for the attribute area in the outline for the Product dimension is:

```
(23,000 x (20 + 50)) bits / 8 bits/byte = 201,250 bytes
```

Assume another base dimension, Market (with 2,500 members), is associated with the 12-member attribute dimension, Population. The attribute area in the outline for the Market dimension is equal to (2,500 x 12) / 8, or 3,750 bytes.

The total estimated size of the outline is equal to the sum of 10,400,000 bytes + 201,250 bytes + 3,750 bytes which is 10,605,000 bytes.

Allowing for Restructuring and Migration Work Areas

During restructuring, Hyperion Essbase uses a restructuring work area on the disk.

During migration from prior releases of Hyperion Essbase, for recovery purposes, Hyperion Essbase creates a copy of the database in a migration work area.

To create these temporary work areas, Hyperion Essbase may require disk space equal to the size of the entire database, including the outline. Because migration and restructuring do not occur at the same time, a single allocation can represent both requirements.

Estimating Disk Space for a Database

After you estimate the size of each database component, you are ready to calculate an estimate of disk space that you need for the database.

The process for determining the total disk space needed for a database is shown in Table 15-2. It is assumed that the database does not contain linked reporting objects and that bitmap compression is enabled.

Table 15-2: Estimating Disk Space for a Database

	Component	Example Value
	Size of data files (If blocks are compressed, use compressed block size.)	5,760,000,000
+	Estimated size of index	1,680,000,000
+	Fixed size overhead (header and bitmap)	2,280,000,000
+	Fragmentation allowance	1,152,000,000
=		10,872,000,000
x 2	To allow for the data recovery area	
=		21,744,000,000
+	Estimated size of outline	10,605,000
=		21,754,605,000
x 2	To allow for the restructuring and migration work area	
=	**Total estimated disk space needed**	43,509,210,000 bytes

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, Hyperion Essbase temporarily retains duplicate blocks before committing transactions.

When you use any type of partition, Hyperion Essbase stores certain data (such as connection and time stamp information) at both the source and target databases. However, the amount of storage Hyperion 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

You can use the Linked Reporting Objects (LRO) to associate objects, such as flat files, with data cells.

If the linked object associated with a cell is a flat file, Hyperion 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 the linked object is a file (as opposed to a cell note). In Hyperion Essbase Application Manager, select Application > Settings and enter the desired limit in the Max. Attachment File Size text box.

You can use the SETAPPSTATE command in ESSCMD to limit the size of a linked object. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

The Hyperion Essbase kernel stores information about linked reporting objects in a Linked Reporting Objects catalog. A catalog entry is stored as an index page. For every catalog entry, Hyperion Essbase uses 8 KB if direct I/O is used; if buffered I/O is used, Hyperion Essbase uses the index page size. For every cell note, Hyperion Essbase uses 600 bytes. The cell note is stored in the catalog.

For more information, see Chapter 12, "Linking Objects to Hyperion Essbase Data."

Total Disk Requirement

The previous calculations in this chapter estimate the data storage requirement for a single database. Often, more than one database resides on the server.

The total Hyperion Essbase data storage requirement on a server includes the Hyperion Essbase software and the sum of the requirements of all the databases to be stored on the server.

Allow 35 to 50 MB for the base installation of Hyperion Essbase server software and sample applications. The allowance varies by platform. For details, see the *Hyperion Essbase Installation Guide*.

Determining Memory Requirements

The minimum memory requirement for running Hyperion Essbase is 64 megabytes (MB) per database. On UNIX systems, the minimum requirement is 128 MB per database. Based on the number of databases and the database operations on the server, the amount of memory you require may be more.

Hyperion Essbase uses buffers called *caches* to hold data and index information in memory while database operations are being performed. To optimize your database performance, Hyperion Essbase enables you to define cache settings and sizes.

This topic provides guidelines for determining the cache settings and sizing your memory requirements. It is recommended that you load your databases, define the cache settings based on the guidelines that follow, and run the basic calculation and retrieval operations for each database. Depending on the amount of memory that is available, you may need to fine tune the cache settings to optimize performance.

The following subtopics enable you to estimate the size of each of the individual database components that require memory resource. After you determine the individual component sizes, to approximate the total memory requirement, see "Estimating Total Memory Requirements" on page 15-26.

General Memory Requirements Per Database

At startup of a database, Hyperion Essbase sets aside memory for the following components:

- Hyperion Essbase code and static data (approximately 10 MB)
- The database outline (350–450 bytes per member)
- The index cache
- Cache-related overhead

How much additional memory is required for normal operation of a database, after it is started, depends on the following factors:

- Different operation types and their associated cache allocations. Data load, data retrieval, and calculation operations set aside memory for the data file, data, and calculation caches plus some overhead associated with the caches.

- Database workload; for example, the complexity of a calc script or the number and complexity of data queries (see "Estimating Additional Memory Requirements for Database Operations" on page 15-20)
- The number of members in the database outline (to view this information in Hyperion Essbase Application Manager, select Database > Information and click the Statistics tab).
- The number of Dynamic Calc members in the outline, including members of attribute dimensions

In addition, the following database settings affect memory usage:

- The isolation level settings (see "Specifying Isolation Level" on page 41-15)
- Synchronization thresholds (see "Specifying Isolation Level" on page 41-15)
- The sizes of the retrieval buffer and the retrieval sort buffer (see Chapter 38, "Optimizing Your Reports")
- The number of data blocks defined using the CALCLOCKBLOCK setting in the ESSBASE.CFG file and the SET LOCKBLOCK setting, which specifies which CALCLOCKBLOCK setting to use (see "Locking Blocks During Calculation" on page 33-22)

Sizing Your Index, Data File, and Data Caches

Cache size settings can have a significant impact on database and general server performance. For general performance information regarding cache sizes, see Chapter 40, "Introducing the Hyperion Essbase Kernel."

This topic provides information about setting up your cache sizes with recommendations for ideal starting sizes to optimize performance. The exact setting you should use for each cache also depends on data distribution and the dense/sparse configuration of the database. For definitions of index cache, index page, data file cache, and data cache, see "Understanding How Hyperion Essbase Stores Data" on page 15-1.

If your memory resource is restricted, you can optimize your performance by adjusting your settings.

Note: If you are migrating from a previous release of Hyperion Essbase, see the *Hyperion Essbase Start Here* booklet for important information on migrating existing cache settings.

Using Cache Memory Locking

Before setting cache sizes, you need to determine whether or not you want to enable cache memory locking. The cache memory locking setting locks the memory used for the index cache, data file cache, and data cache into physical memory.

If you want to give the Hyperion Essbase kernel priority usage of the physical memory, enable cache memory locking. If you enable cache memory locking, leave sufficient physical memory available for non-Hyperion Essbase kernel use. If you do not want to give the Hyperion Essbase kernel priority usage of the physical memory, disable cache memory locking. By default, cache memory locking is turned off.

For information about how to enable cache memory locking, see "Enabling Cache Memory Locking" on page 41-12.

Note: In order to take advantage of the cache memory locking feature on Solaris, you must run the Bourne shell script, *root.sh*, after installing. This sets the server to run in Superuser mode so it can lock memory. For more information, see the *Hyperion Essbase Installation Guide*.

Estimating Cache Sizes

If you are using Hyperion Essbase for the first time, cache sizes are automatically set to the default values shown in Table 15-3. If you are migrating from a previous version of Hyperion Essbase, the new data file cache is set to the default value and the other cache settings from that version are retained when you migrate. See the *Hyperion Essbase Start Here* booklet for migration information.

To help you estimate cache memory requirements, Table 15-3 shows default and recommended cache settings. These caches are listed in priority order from highest priority to lowest priority. For example, if memory resources are restricted, allocating adequate memory to the index cache is top priority, followed by the index page, the data file cache, and so on.

The needs for each site and even for a particular database can vary. Depending on the complexity and type of each operation, Hyperion Essbase allocates as much memory for data file cache and data cache as needed. Use the recommended values in this table to estimate enough memory for *optimal* performance.

Table 15-3: Recommendations for Cache-Related Settings

Cache-Related Settings in priority order	Minimum Value	Default Value	Recommended Value
Index Cache Size	1024 KB (1048576 bytes)	10240 KB (10485760 bytes)	Combined size of all ESS*.IND files, if possible; as large as possible otherwise. Do not set this cache size higher than the total index size, as no performance improvement results. (To determine the total index size see "Estimating the Size of the Index" on page 15-8.)
Index Page Size	1 KB (1024 bytes)	• Direct I/O: 8 KB (8192 bytes) • Buffered I/O: 1 KB (1024 bytes)	• If direct I/O, Hyperion Essbase ignores the Index Page Size setting and always employs an index page size of 8 KB. • If buffered I/O and you are migrating, keep the pre-Release 6 value. Otherwise, set to 8 KB.
Data File Cache Size	80 KB (8388608 bytes)	320 KB (33554432 bytes)	Combined size of all ESS*.PAG files, if possible; as large as possible otherwise.
Data Cache Size	30 KB (3145728 bytes)	30 KB (3145728 bytes)	0.125 * data file cache size value. Allocate more if: • Many concurrent users are accessing different data blocks • Calc scripts contain functions on sparse ranges and the functions require all those members in memory (for example, @RANK and @RANGE) **Note:** If migrating, keep the pre-Release 6 value.
Cache Memory Locking	Not applicable	Disabled	To give the Hyperion Essbase kernel priority usage of system RAM, enable cache memory locking. Otherwise, disable it.

For instructions on setting index cache, index page, data file cache, and data cache sizes and on defining cache memory locking settings, see “Specifying and Changing Hyperion Essbase Kernel Settings” on page 41-3.

An additional cache, the calculator cache, is used only during calculations. Consider the calculator cache memory requirements when you size memory requirements for database operations (see “Estimating Additional Memory Requirements for Calculations” on page 15-25).

Fine Tuning Your Cache Settings

After using the database under your installation’s usual conditions, check to see how Hyperion Essbase is performing. You may need to adjust the settings.

Testing Your Cache Settings

You can check Hyperion Essbase performance in several ways. Chapter 46, “Using Diagnostics to Monitor Performance,” provides more information, but here is an example:

Check the Run-time page of the Database Information dialog box (select Database > Information from the Hyperion Essbase Application Manager menu). In particular, check the hit ratios:

- The Hit Ratio on Index Cache setting indicates the Hyperion Essbase kernel’s success rate in locating index information in the index cache without having to retrieve another index page from disk.
- The Hit Ratio on Data File Cache setting indicates the Hyperion Essbase kernel’s success rate in locating data file pages in the data file cache without having to retrieve the data file page from disk.
- The Hit Ratio on Data Cache setting indicates the Hyperion Essbase kernel’s success rate in locating data blocks in the data cache without having to retrieve the block from the data file cache.

A higher hit ratio on the index and data file caches indicates better performance; 100 is the highest possible value.

Determining Which Settings to Change

The sizes of the index cache and the data file cache are the most critical Hyperion Essbase cache settings. In general, the larger these buffers, the less swapping activity occurs; however, it does not always help performance to set cache sizes larger and larger. Read this entire section to understand the cache size considerations.

The advantages of a large index cache start to level off after a certain point. At any given time, when the index cache size equals or exceeds the index size (including all index files on all volumes), performance does not improve. However, to account for future growth of the index, you can set the index cache size larger than the current index size. See “Estimating the Size of the Index” on page 15-8 for an example of estimating index size.

Because the index cache is filled with index pages, for optimum use of storage, set the size of the index cache to be a multiple of the size of the index page.

If possible, set the data file cache to equal the size of the stored data, which is the combined size of all `ESS*.PAG` files. Otherwise, the data file cache should be as large as possible. If you want to account for future growth of stored data, you can set the data file cache size larger than the current size of stored data.

The data cache should be about 0.125 times the data file cache. However, certain calculations require a larger data cache size. If many concurrent users are accessing different data blocks, this cache should be larger.

In general, if you have to choose between allocating memory to the data file cache or allocating it to the data cache, choose the data file cache. If you are migrating from a previous version of Hyperion Essbase, see the *Hyperion Essbase Start Here* booklet for important migration information.

Sizing Cache-Related Overhead

In addition to the memory used by the caches, Hyperion Essbase uses some additional memory while it works with the caches. To calculate these areas, use the formulas in Table 15-4.

Table 15-4: Estimating Cache-Related Overhead

Overhead	Formula	When Used
Index cache	Index cache size * .5	At startup
General cache	((# of server threads allocated to the Hyperion Essbase server process * 3) * 256) + 5242880 bytes	At startup
Data file cache	(Data file cache size/8192) * 110	During calculations, retrievals, and data loads
Data cache	(Data cache size/Potential stored block size * 8) * 160	During calculations, retrievals, and data loads

To determine the potential stored block size, see “Determining the Expanded Block Size of Stored Data” on page 15-5.

To determine the # of server threads initially allocated to the Hyperion Essbase server process, see the SERVERTHREADS setting in the online *Technical Reference* in the `DOCS` directory

Estimating Additional Memory Requirements for Database Operations

In addition to startup memory requirements and cache requirements, daily operations such as queries and periodic operations such as calculations require additional memory. To estimate actual memory requirements, you must also evaluate the memory used during these operations.

Estimating Additional Memory Requirements for Data Retrievals

Hyperion Essbase processes requests for database information (queries) from a variety of sources. For example, Hyperion Essbase processes queries from the Hyperion Essbase Spreadsheet Add-in and from Hyperion Essbase Report Writer. Hyperion Essbase uses additional memory when it retrieves the data for these queries, especially when Hyperion Essbase must perform dynamic calculations to retrieve the data. This section describes Hyperion Essbase's memory requirements for query processing.

Hyperion Essbase is a multithreaded application in which queries get assigned to threads. Threads are created when Hyperion Essbase is started. In general, once a thread is created, it exists until you shut down the Hyperion Essbase server (for more information, see Chapter 45, "Running Hyperion Essbase, Applications, and Databases").

As Hyperion Essbase processes queries, it cycles through the available threads. For example, assume 20 threads are available at startup. As each query is processed, Hyperion Essbase assigns each succeeding query to the next sequential thread. Hyperion Essbase cycles back to the beginning, assigning the 21st query to the first thread.

A thread allocates some memory while processing a query and frees most of this allocated memory when the query is completed. However, the thread holds on to a portion of the memory for possible use in processing subsequent queries. As a result, after a thread has processed its first query, the memory held by the thread is greater than it was when Essbase first started.

The maximum amount of memory required for query processing occurs when the maximum number of simultaneous queries that will occur are being processed and all threads have been assigned to at least one query by Hyperion Essbase.

In the example, the maximum amount of memory used for queries occurs when at least 20 queries have been processed and the maximum number of simultaneous queries are in process.

Calculating the Maximum Amount of Additional Memory Required

To estimate query memory requirements, this topic describes a method whereby you observe the memory used during queries. To calculate the maximum possible use of memory for query processing, you summarize the memory used by queries that will be run simultaneously plus the extra memory acquired by threads that are waiting for queries.

The final formula uses the following variables:

- Maximum number of possible concurrent queries (*Max#ConcQueries*)
- Maximum memory usage for any query *during* processing (*MAXAdditionalMemDuringP*)
- Total number of threads (*Total#Threads*)
- Maximum memory usage for any query *after* processing (*MAXAdditionalMemAfterP*)

Determining the Total Number of Threads

The potential number of threads available is based on the number of licensed ports that are purchased. The actual number of threads available depends on settings you define for the Agent or the server. See “Multithreading” on page 45-15. Use this value for *Total#Threads* in step 8 below.

Estimating the Maximum Number of Concurrent Queries

Based on your knowledge of your company and the requirements for each database, determine the maximum number of concurrent queries and use this value for *Max#ConcQueries* in step 8 below. This value cannot exceed the value for *Total#Threads*.

Estimating the Maximum Memory Usage for Any Query Before and After Processing

The memory usage of individual queries depends on the size of each query and the number of data blocks that Hyperion Essbase needs to access to process each query. To estimate the memory usage, calculate the additional memory Hyperion Essbase uses while processing and after processing each query.

Decide on several queries that you expect to use the most memory. Consider queries that must process large numbers of members; for example, queries that perform range or rank processing.

For each query, complete the following steps:

1. Start the Hyperion Essbase application.
2. Using memory monitoring tools for the operating system, note the memory used by the Hyperion Essbase server *before* processing the query. Use the value associated with the ESSSVR process.

   Use this value for *MemBeforeP*.
3. Run the query.
4. Using memory monitoring tools for the operating system, *while* processing the query, note the peak memory usage of the Hyperion Essbase server. This value is associated with the ESSSVR process.

   Use this value for *MemDuringP*.
5. Using memory monitoring tools for the operating system, *after* the query is completed, note the memory usage of the Hyperion Essbase server. This value is associated with the ESSSVR process.

   Use this value for *MemAfterP*.
6. For each query, calculate the following two values:
   - Additional memory used while Hyperion Essbase processes the query (*AdditionalMemDuringP*):

     AdditionalMemDuringP = MemDuringP - MemBeforeP
   - Additional memory used after Hyperion Essbase has finished processing the query (*AdditionalMemAfterP*):

     AdditionalMemAfterP = MemAfterP - MemBeforeP

7. When you have completed the above calculations for all the relevant queries, compare all results to determine the following two values:
   - The maximum *AdditionalMemDuringP* used by a query: (*MAXAdditionalMemDuringP*)
   - The maximum *AdditionalMemAfterP* used by a query: (*MAXAdditionalMemAfterP*)
8. Insert the values from step 7 into the formula in following statement:

   The amount of additional memory required for data retrievals will not exceed:

   *Max#ConcQueries*MAXAdditionalMemDuringP* + (*Total#Threads* - *Max#ConcQueries*)**MAXAdditionalMemAfterP.*

Because this calculation assumes that all of the concurrent queries are maximum-sized queries, the result may exceed your actual requirement. It is very difficult to estimate the actual types of queries that will be run concurrently.

To adjust the memory used during queries, you can set values for the retrieval buffer and the retrieval sort buffer. For information, see "Setting the Retrieval Buffer Size" on page 38-2 and "Setting the Retrieval Sort Buffer Size" on page 38-4.

Note: If you cannot perform this test with a query, you can calculate a very rough estimate for the operational requirement of a query by summarizing the following values:

- The size of the data file cache (see "Estimating Cache Sizes" on page 15-16)
- The size of the data file cache overhead (see "Sizing Cache-Related Overhead" on page 15-20)
- The size of the data cache (see "Estimating Cache Sizes" on page 15-16)
- The size of the data cache overhead (see "Sizing Cache-Related Overhead" on page 15-20)
- The size of the retrieval buffer (see "Setting the Retrieval Buffer Size" on page 38-2)
- (If sorting is involved in the query) The size of the retrieval sort buffer (see "Setting the Retrieval Sort Buffer Size" on page 38-4)

- (If the query uses dynamic calculations) The number of data blocks specified by the SET LOCKBLOCK command. To calculate the memory requirement in bytes, multiply the specified number of data blocks by the size of the data blocks (see “Estimating the Size of Compressed Data Blocks” on page 15-5)

Estimating Additional Memory Requirements for Calculations

For existing calc scripts, you can use the memory monitoring tools provided for the operating system on the server to observe memory usage. Run the most complex calculation and take note of the memory usage both before and during running the calculation. Calculate the difference and use that figure as the additional memory requirement for the calc script.

To understand calculation performance, see Chapter 33, “Optimizing Calculations.”

If you cannot perform a test with a calc script, you can calculate a very rough estimate for the operational requirement of a calculation by summarizing the following values:

- The size of the data file cache (see “Estimating Cache Sizes” on page 15-16)
- The size of the data file cache overhead (see “Sizing Cache-Related Overhead” on page 15-20)
- The size of the data cache (see “Estimating Cache Sizes” on page 15-16)
- The size of the data cache overhead (see “Sizing Cache-Related Overhead” on page 15-20)
- The size of the calculator cache (see “Setting Memory Cache Sizes” on page 33-14)
- The size of the outline. Hyperion Essbase uses approximately 30 bytes of memory per member of the database outline (see “Setting Memory Cache Sizes” on page 33-14)

- The number of data blocks specified by the CALCLOCKBLOCK command. To calculate the memory requirement in bytes, multiple the specified number of data blocks by the expanded size of the data blocks (see “Determining the Expanded Block Size of Stored Data” on page 15-5)

Note: The size and complexity of the calc scripts that you run affects the memory requirement. However, how this affects the amount of memory required is difficult to estimate.

Estimating Total Memory Requirements

To estimate the total memory required for a server, you must consider the following factors:

- The total start-up memory requirements of all Hyperion Essbase databases that are running simultaneously on the server. To be running, a database has been started and not stopped. A running database need not have operations performed on it.
- Additional operational requirements of the databases at peak usage.

The basic formula for estimating the total start-up memory needed per database is shown in Table 15-5.

Table 15-5: Formula for Calculating Database Start-Up Memory Requirements

	Component	Size (in bytes)
+	Code and static data	1,024,000 bytes
+	Size of outline (see “Estimating the Size of the Outline” on page 15-10)	(See “Estimating the Size of the Outline” on page 15-10)
+	Index cache size	(See “Sizing Your Index, Data File, and Data Caches” on page 15-15)
+	Index cache-related overhead	(See “Estimating Cache Sizes” on page 15-16)
+	Other cache-related overhead used at startup	(See “Estimating Cache Sizes” on page 15-16)
=	**Total database memory used at startup**	

Additional operational requirements for memory may include memory used during the following operations:

The maximum operational requirements for queries	(See "Estimating Additional Memory Requirements for Database Operations" on page 15-20)
The maximum operational requirement for calculations	(See "Estimating Additional Memory Requirements for Calculations" on page 15-25)

➤ To estimate the total Hyperion Essbase memory requirement on a server, perform the following steps:

1. As shown in Table 15-5, calculate the start-up memory requirement for each database.
2. Calculate the operational memory requirement for each database.
3. Determine the different combinations (sets) of databases that will run concurrently on the server. For example, if at one time, only Sample Basic and Demo Basic will run concurrently, Sample Basic and Demo Basic would comprise one set.
4. Summarize the start-up memory requirement for each set of databases.
5. For each set of databases, determine which operations will be performed concurrently.
6. To determine the combination of operations that use the most memory, summarize the additional memory usage for each combination of operations. Use the additional memory figures you observed or calculated, as described in "Estimating Additional Memory Requirements for Data Retrievals" on page 15-21 and "Estimating Additional Memory Requirements for Calculations" on page 15-25. Select the largest requirement.

   This is the total additional operational memory requirement.
7. Add the operational memory requirement to the start-up memory requirement.

   This is an estimate of the memory needed for the Hyperion Essbase databases on the server.

8. To estimate the total memory requirement, add the base memory used by the operating system to the result from step 7.
9. Compare the result from step 8 with the total available RAM on the server.

   If cache memory locking is enabled, the total memory requirement should not exceed two-thirds of available RAM; otherwise, system performance is severely degraded. If cache memory locking is disabled, the total memory requirement should not exceed available RAM.

   If there is insufficient memory available, you can redefine your cache settings and recalculate the memory requirements. This can be an iterative process. For guidelines, see “Fine Tuning Your Cache Settings” on page 15-18. In some cases, you may need to purchase additional RAM.

Chapter 16

Building and Maintaining Partitions

When you build a new partition, each database in the partition uses a partition definition file to record all information about the partition, such as its data source and data target and the areas to share. This chapter describes how to create a replicated, transparent, or linked partition.

CAUTION: You must design partitions carefully. We strongly recommend that you read Chapter 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.

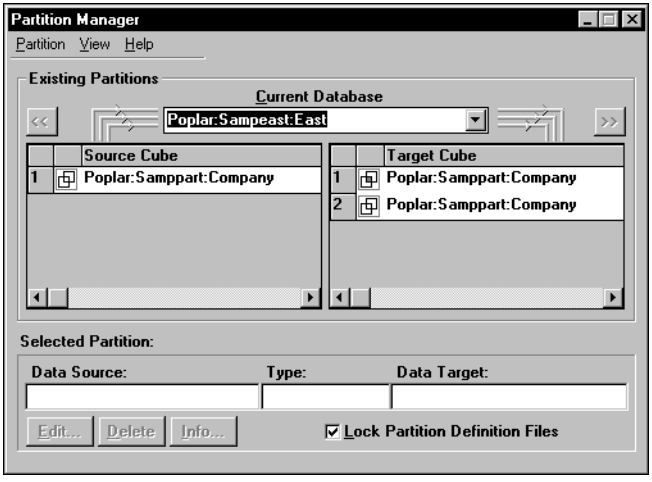


Figure 16-1: Partition Manager

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 16-2 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.

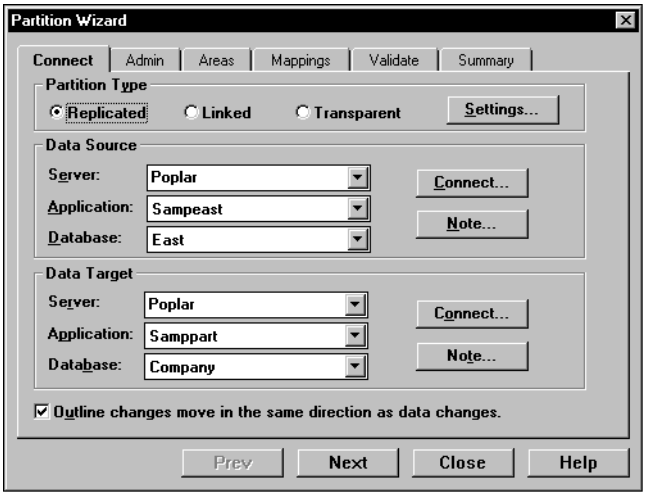


Figure 16-2: Connect Page of the Partition Wizard

Note: For more information on using Dynamic Time Series members in partitions, see "Using Dynamic Time Series Members in Partitions" on page 30-16.

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 6-10.

The examples used in this section are based on the example described in "Scenario 1: Partitioning an Existing Database" on page 6-33. 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 16-5
- "Setting the User Name and Password" on page 16-6

- "Defining the Areas in a Partition" on page 16-8
- "Mapping Data Source Members to Data Target Members" on page 16-11
- "Validating the Partition" on page 16-38
- "Saving the Partition Definition" on page 16-40

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 16-3.
2. Choose Replicated from the Connection page of the Partition Wizard.
3. Click the Settings button to open the Replication Properties dialog box.

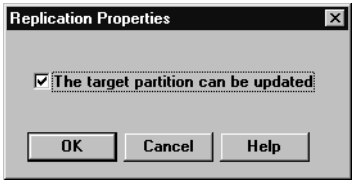


Figure 16-3: Replication Properties Dialog Box

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:

   - Hyperion 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.

5. Click OK.
6. Choose the server, application, and database names for the data source and the data target. If you are not connected to the appropriate server, click Connect. For our example, choose Sampeast and East as the data source and Samppart and Company as the data target. The server should correspond to the server on which your Samppart and Sampeast applications reside.

   Note: Do not use network aliases when creating partitions unless you are *certain* that they are propagated to all computers on your system. If you're not certain, use the full server name. This is especially important for linked partitions, because this is the host name that clients connected to the data target use to find the data source.

7. If desired, enter notes about the data source or the data target by clicking the Note button to open the Note dialog box. Enter your notes and click OK.
8. To propagate changes in the data source's outline to the data target's outline, check the Outline changes move in the same direction as data changes box. To propagate changes from the data target's outline to the data source's outline, clear the box. If you're not sure which outline to use as the source outline, see "Introducing Outline Synchronization" on page 16-47.

Setting the User Name and Password

You must specify a user name and password for Hyperion Essbase to use when communicating between the data source and the data target. Hyperion 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 6-30.

➤ To set the user name and password for the data source and the data target:

1. From the Connection page of the Partition Wizard, click Next or Admin to move to the Admin page.

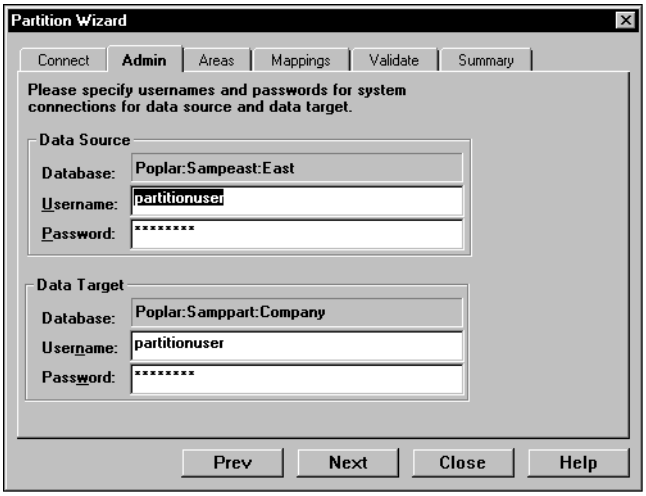


Figure 16-4: Admin Page of the Partition Wizard

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.

Note: The cell count does not include those of attribute dimensions.

➤ To define an area in a partition:

1. From the Admin page of the Partition Wizard, click Next or Areas to move to the Areas page.

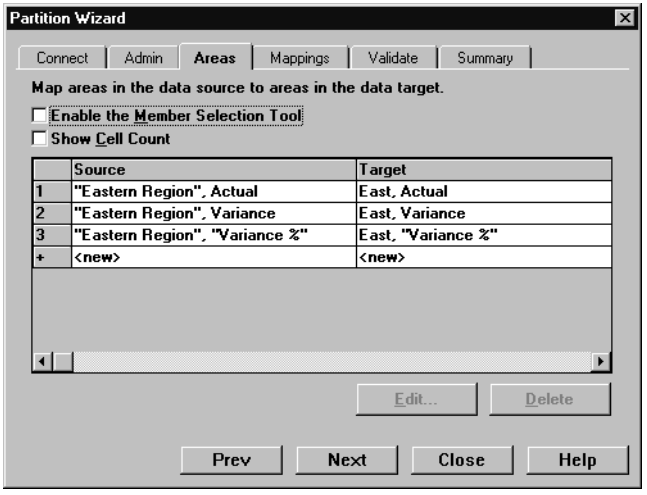


Figure 16-5: Areas Page of Partition Wizard

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 6-9.

   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 16-5) 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 Hyperion 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 Hyperion Essbase functions, see the online *Technical Reference* in the `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 16-5) to open the Member Selection dialog box. For more information about the parts of the Member Selection dialog box, click Help.

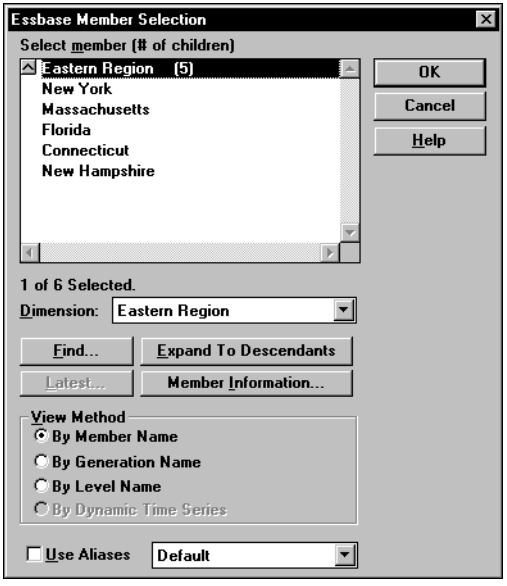


Figure 16-6: Member Selection Dialog Box

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. Hyperion 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 16-6. 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 16-9.

Mapping Data Source Members to Data Target Members

Hyperion Eassbase 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, Hyperion Essbase automatically maps the members. Skip to "Validating the Partition" on page 16-38.

Note: Use dimension member names instead of their aliases to create a valid partition.

It is recommended that the data source member names and the data target member names are the same. This reduces the maintenance requirements for the partition, especially when the partition is based on member attributes. See "Introducing Outline Synchronization" on page 16-47 for more information.

Mapping Members with Different Names

If the data source outline and data target outline contain different members or the members have different names in each outline, you must map the data source members to the data target members. In the following example, the first two member names are identical but the third member name is different:

Source	Target
Product	Product
Cola	Cola
Year	Year
1998	1998
Market	Market
East	East_Region

Because you know that East in the data source corresponds to Eastern_Region in the data target, map East to Eastern_Region. Then, all references to Eastern_Region in the data target point to East in the data source. For example, if the data value for Cola, 1998, East is 15 in the data source, the data value for Cola, 1998, Eastern_Region is 15 in the data target.

Mapping Data Cubes with Extra Dimensions

There can be instances when the number of dimensions in the data source and in the data target vary. The following example illustrates a case where there are more dimensions in the outline of the data source than in the outline of the data target:

Source	Target
Product	Product
Cola	Cola
Market	Market
East	East
Year	
1999	
1998	
1997	

If you want to map member 1997 of the Year dimension from the data source to the data target, you can map it to Void in the data target. But first, you must define the areas of the data source to share with the data target in the Areas tab of the Partition Wizard as shown in Figure 16-7:

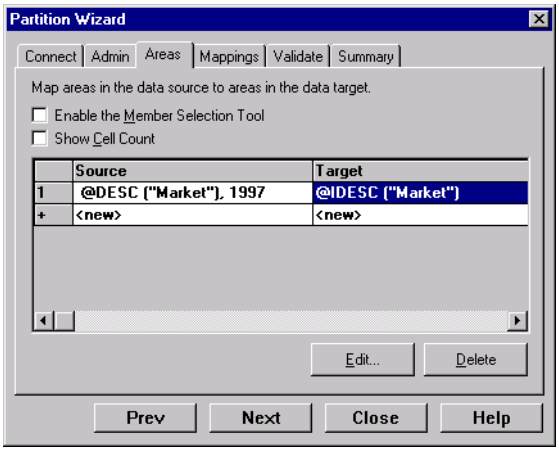


Figure 16-7: Defining Areas for Data Cubes with Extra Dimensions

You can then map the data source member to Void in the data target in the Mappings tab of the Partition Wizard, as follows:

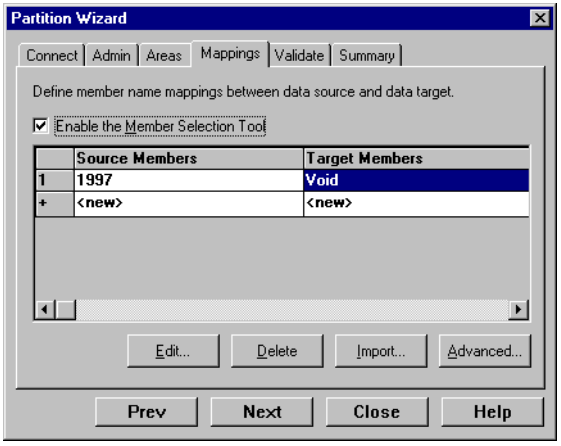


Figure 16-8: Mapping Data Cubes with Extra Dimensions

If you do not include at least one member from the extra dimension in the Areas definition, you will receive an error message when you attempt to validate the partition.

Note: When you map a member from an extra dimension, the partition results reflect data only for the mapped member. In the above example, the Year dimension contains three members: 1999, 1998, and 1997. If you map member 1997 from the data source to the data target, then the partition results reflect Product and Market data only for 1997. Product and Market data for 1998 and 1999 will not be extracted.

The following example illustrates a case where the data target includes more dimensions than the data source:

Source	Target
Product	Product
Cola	Cola
	Market
	East
Year	Year
1997	1997

In such cases, you must first define the shared areas of the data source and the data target as shown in Figure 16-9:

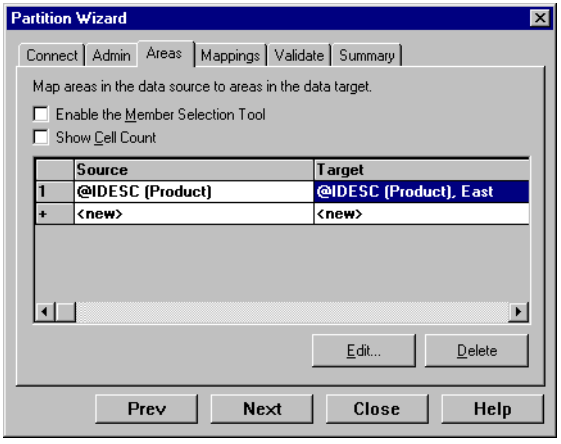


Figure 16-9: Defining Areas for Data Cubes with Extra Dimensions

You can then map member East from the Market dimension of the data target to Void in the data source as follows:

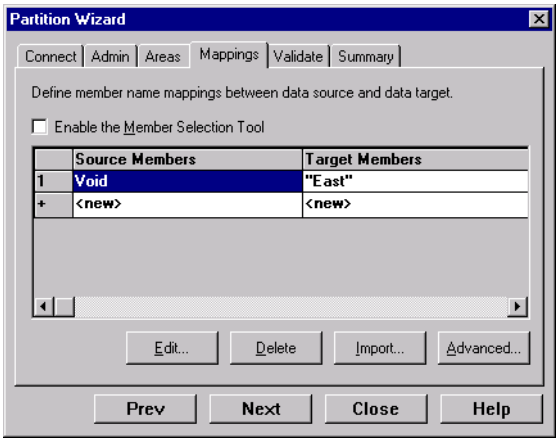


Figure 16-10: Mapping Extra Dimensions in Partitioning

If member East from the Market dimension in the data target is not included in the target areas definition, you will receive an error message when you attempt to validate the partition.

➤ To map member names:

1. From the Areas page of the Partition Wizard, click Next or Mappings to move to the Mappings page.

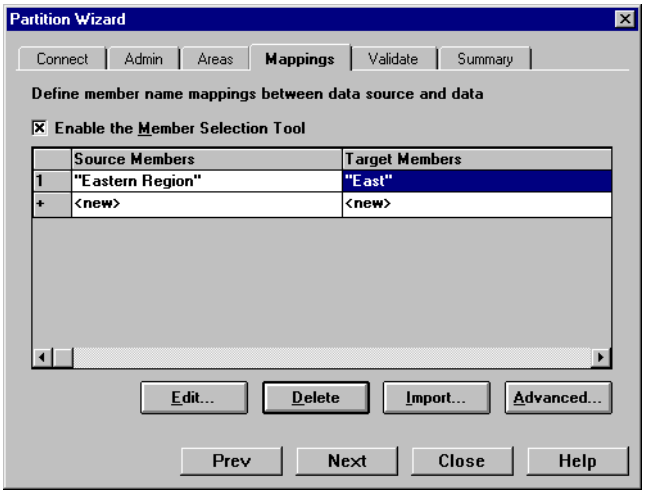


Figure 16-11: Mappings Page of the Partition Wizard

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 Mappings” on page 16-17.
   - Select the member name in the Member Select dialog box. See “Entering an Area Using the Member Selection Dialog Box” on page 16-10.
   - Import the member mappings from an external data file. See “Importing Member Mappings” on page 16-19.

Note: You cannot map attributes dimension members in replicated partitions. For more information, refer to “Rules for Transparent Partitions” on page 6-16. You can, however, map attributes in transparent and linked partitions. For information on using attributes in partitions, see “Using Attributes in Partitions” on page 6-28. For information on mapping attributes, see “Mapping Attributes Associated with Members” on page 16-28.

Manually Defining Mappings

➤ To enter the member in the data source and the member that it maps to in the data target:

1. In the Mappings page of the Partition Wizard, make sure that the Enable the Member Selection Tool box is *cleared*.
2. Double-click the New field in the Source Members box (Figure 16-11) to open the Member Name dialog box.
3. Enter the member name for the data source and click OK. For our example, enter:

   ```
   "Eastern Region"
   ```

4. Double-click the New field in the Target Members box to open the Member Name dialog box.
5. Enter the member name to map it to in the data target and click OK. For our example, enter:

   ```
   East
   ```

Using the Member Selection Dialog Box to Enter Mappings

➤ To choose the member in the data source and the member that it maps to in the data target from the Member Selection dialog box:

1. In the Mappings page of the Partition Wizard, make sure that the Enable Member Selection Tool box is checked.

2. Double-click the New field in the Source Members box (Figure 16-11) to open the Member Selection dialog box. For more information about the parts of the Member Selection dialog box, click Help.

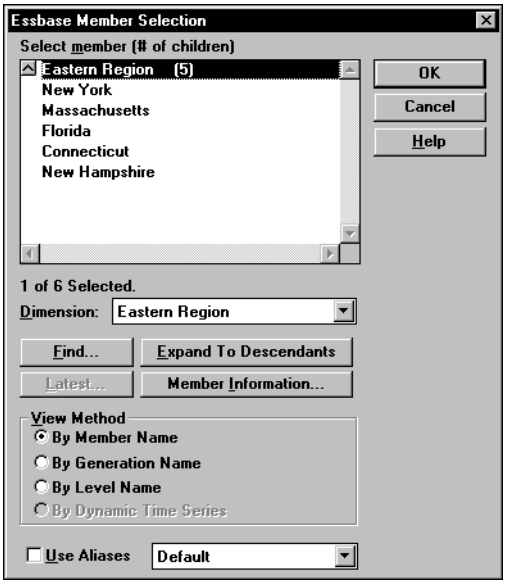


Figure 16-12: Member Selection Dialog Box

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.
6. When you finish, click OK.

Note: For the linked partition described in "Scenario 3: Linking Two Databases" on page 6-38, you must also map the Package dimension to `Void`.

You can only use the Member Selection dialog box to enter new members, not to edit existing members. To edit existing members, use the Member Name dialog box. See "Manually Defining an Area" on page 16-9.

Importing Member Mappings

➤ To import the names from an external file:

1. In the Mappings page of the Partition Wizard, click Import to open the Import Member Mappings dialog box.

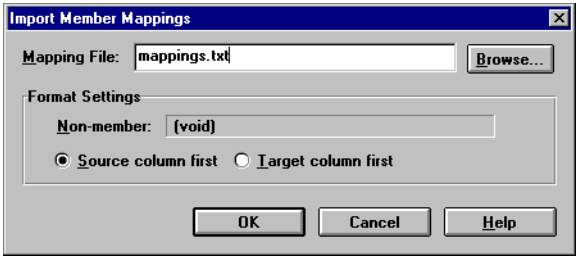


Figure 16-13: Import Member Mappings Dialog Box

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):

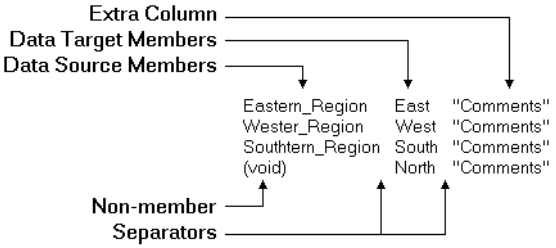


Figure 16-14: Member Mapping Import File

- 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 Hyperion Essbase database or on another Hyperion Essbase server. For more information, see “Transparent Partitions” on page 6-15.

The examples used in this section are based on the example described in “Scenario 1: Partitioning an Existing Database” on page 6-33. 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 16-21
- “Setting the User Name and Password” on page 16-21
- “Defining the Areas in a Transparent Partition” on page 16-22
- “Mapping Data Source Members to Data Target Members” on page 16-25
- “Validating the Partition” on page 16-38
- “Saving the Partition Definition” on page 16-40

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 16-3.
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 16-47.

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 16-6.

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 16-8.

➤ To define an area in a transparent partition:

1. From the Admin page of the Partition Wizard, click Next or Areas to move to the Areas page.

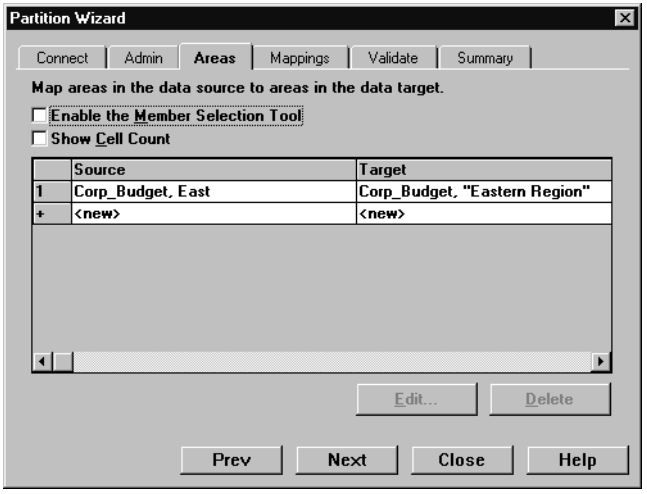


Figure 16-15: Areas Page of Partition Wizard

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 6-9.

   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 Hyperion 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 Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

4. Repeat steps 2 and 3 for the Target box. For our example, enter:

   ```
   Corp_Budget, "Eastern Region"
   ```

Using the Member Selection Dialog Box to Enter an Area

➤ To choose the members that make up an area from the Member Selection dialog box:

1. In the Areas page of the Partition Manager, make sure that the Enable Member Selection Tool box is checked.

2. Double-click the New field in the Source box to open the Member Selection dialog box. For more information about the parts of the Member Selection dialog box, click Help.

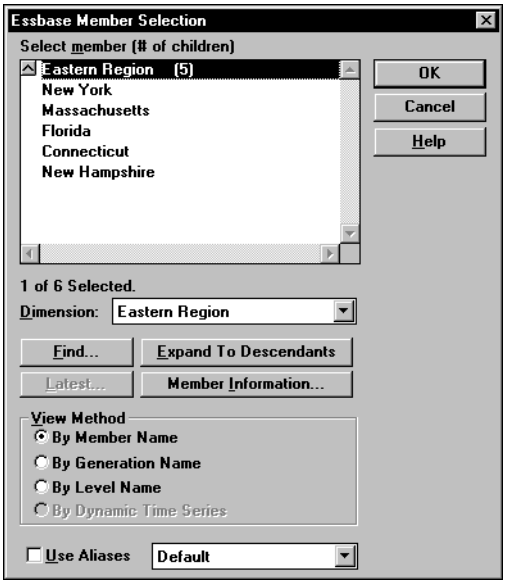


Figure 16-16: Member Selection Dialog Box

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. Hyperion 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 16-9.

Mapping Data Source Members to Data Target Members

Hyperion 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, Hyperion Essbase automatically maps the members. Skip to “Validating the Partition” on page 16-38.

For a detailed introduction to member mapping, see “Mapping Data Source Members to Data Target Members” on page 16-11.

For information on using attributes in partitions, see “Using Attributes in Partitions” on page 6-28. For information on mapping attributes associated with members of a base dimension, see “Mapping Attributes Associated with Members” on page 16-28.

Note: When you create a replicated or transparent partition using a shared member, use the real member names in the mapping. Hyperion Essbase maps the real member, not the shared one, from the data source.

➤ To map member names for a transparent partition:

1. From the Areas page of the Partition Wizard, click Next or Mappings to move to the Mappings page.

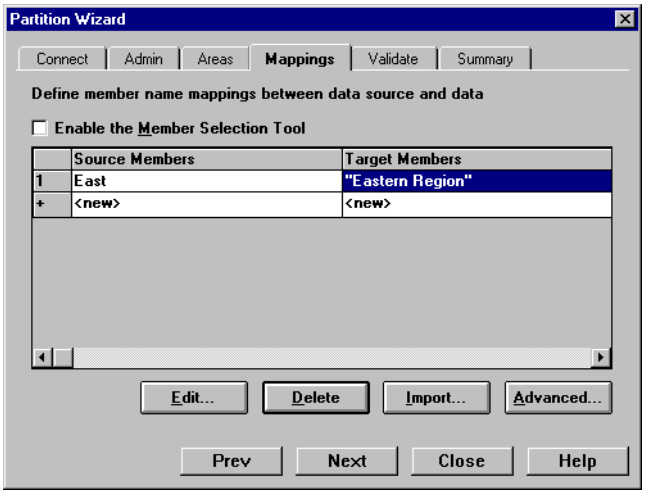


Figure 16-17: Mappings Page of the Partition Wizard

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 16-27.
   - Select the member name in the Member Select dialog box. See “Using the Member Selection Dialog Box to Enter Mappings” on page 16-17.
   - Import the member mappings from an external data file. See “Importing Member Mappings” on page 16-19.

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 16-17) to open the Member Name dialog box.
3. Enter the member name for the data source and click OK. For our example, enter:

   ```
   East
   ```

4. Double-click the New field in the Target Members box to open the Member Name dialog box.
5. Enter the member name to map it to in the data target and click OK. For our example, enter:

   ```
   "Eastern Region"
   ```

Using the Member Selection Dialog Box to Enter Mappings

➤ To choose the member in the data source and the member that it maps to in the data target from the Member Selection dialog box:

1. In the Mappings tab of the Partition Wizard, make sure that the Enable Member Selection Tool box is checked.
2. Double-click the New field in the Source Members box (Figure 16-17) 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 16-9.

Mapping Attributes Associated with Members

You must accurately map attribute dimensions and members from the data source to the data target to ensure that partitioning is valid.

In the following example, the outline for the data source contains a Product dimension with a member 100 (Cola). Children 100-10 and 100-20 are associated with member TRUE of the Caffeinated attribute dimension, and child 100-30 is associated with member FALSE of the Caffeinated attribute dimension.

The data target outline has a Product dimension with a member 200 (Cola). Children 200-10 and 200-20 are associated with member Yes of the With_Caffeine attribute dimension, and child 200-30 is associated with No of the With_Caffeine attribute dimension.

First define the areas to be shared from the data source to the data target in the Areas tab of the Partition Manager as shown in Figure 16-18:

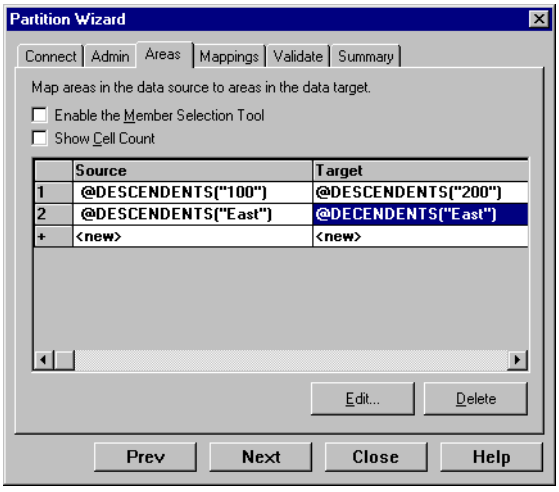


Figure 16-18: Defining Areas for Mapping Attributes

Then map attributes in the Mapping tab of the Partition Manager as follows:

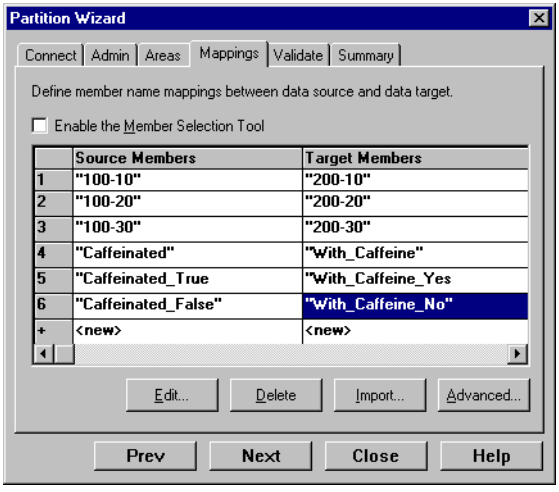


Figure 16-19: Mapping Attributes

If you map attribute Caffeinated_True to attribute With_Caffeine_No, you receive an error message during validation. You must associate caffeinated cola from the data source to caffeinated cola in the data target.

Note: You can only map attributes for transparent and linked partitions. An attempt to map attributes in replicated partitions results in an error message.

There can be instances where an attribute dimension or an attribute member exists in the outline of the data source but not in the outline of the data target, or vice versa. For example:

Source	Target
Caffeinated	
True	
False	

In such cases, you have the following choices:

- Create the Caffeinated attribute dimension and its members in the outline of the data target and associate them with the Product dimension. You can then map the attributes from the data source to the data target.
- Map the Caffeinated attribute dimension in the data source to Void in the data target.

For more information on mapping, see “Mapping Data Cubes with Extra Dimensions” on page 16-12. For more information on attributes, see Chapter 10, “Working with Attributes.” For more information on using attributes in partitions, see “Using Attributes in Partitions” on page 6-28.

Note: You *must* validate the partition to ensure that the results are accurate. When you create a replicated or transparent partition using a shared member, use the real member names in the mapping. Hyperion Essbase maps the real member, not the shared one, from the data source.

Creating a Linked Partition

A linked partition sends users from a cell in one database to a cell in another database. This gives users a different perspective on the data. For more information, see “Linked Partitions” on page 6-23.

The examples used in this section are based on the example described in “Scenario 3: Linking Two Databases” on page 6-38. This example links the Product dimension in the Sample Basic database to the TBC Demo database. This scenario is *not* shipped with Hyperion Essbase.

Creating a linked partition involves the following steps:

- “Specifying the Partition Type and Connection Information” on page 16-31
- “Setting the User Name and Password” on page 16-32
- “Defining the Areas in a Partition” on page 16-33
- “Mapping Data Source Members to Data Target Members” on page 16-35
- “Validating the Partition” on page 16-38
- “Saving the Partition Definition” on page 16-40

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 16-3.
2. Choose Linked from the Connection page of the Partition Wizard.
3. Click the Settings button to open the Link Properties dialog box.

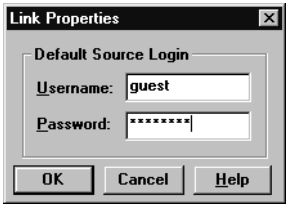


Figure 16-20: Link Properties Dialog Box

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 6-32.
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 Hyperion Essbase.

   Note: Do not use network aliases when creating partitions unless you are *certain* that they are propagated to all computers on your system. If you're not certain, use the full server name. This is especially important for linked partitions, because this is the host name that the data target uses to find the data source.

6. If desired, enter notes about the data source or the data target by clicking the Note button to open the Note dialog box. Enter your notes and click OK. These notes display in the linked objects box to help users select the database to drill across to.
7. To propagate changes in the data source's outline to the data target's outline, check the "Outline changes move in the same direction as data changes" box. To propagate changes from the data target's outline to the data source's outline, clear the box. If you're not sure which outline use as the source outline, see "Introducing Outline Synchronization" on page 16-47.

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 16-6.

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 16-8.

➤ To define an area in a linked partition:

1. From the Admin page of the Partition Wizard, click Next or Areas to move to the Areas page.

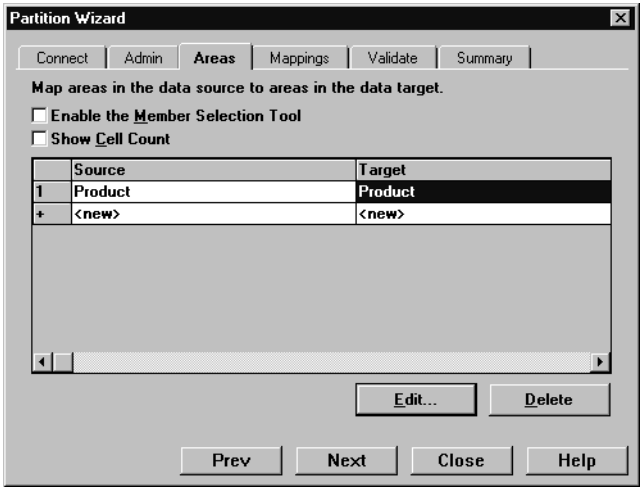


Figure 16-21: Areas Page of Partition Wizard

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 6-9.

   There must be a one-to-one correspondence between cells in the areas for the data source and the data target on each line.

Manually Defining an Area

➤ To enter the members in an area:

1. In the Areas tab of the Partition Manager, make sure that the Enable the Member Selection Tool box is *cleared*.
2. Double-click the New field in the Source box to open the Area Definition dialog box.
3. Enter the member name for the data source and click OK. For our example, enter:

   ```
   Product
   ```

   Note: You can enter Hyperion 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 Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

4. Repeat steps 2 and 3 for the Target box. For our example, enter:

   ```
   Product
   ```

Using the Member Selection Dialog Box to Enter an Area

➤ To choose the members that make up an area from the Member Selection dialog box:

1. In the Areas tab of the Partition Manager, make sure that the Enable Member Selection Tool box is checked.
2. Double-click the New field in the Source box to open the Member Selection dialog box. For more information about the parts of the Member Selection dialog box, click Help.
3. Choose the dimension and member to partition. For our example, choose the Product dimension.

4. Click Add. Hyperion 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 16-9.

Mapping Data Source Members to Data Target Members

Hyperion 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, Hyperion Essbase automatically maps the members. Skip to "Validating the Partition" on page 16-38.

For a detailed introduction to member mapping, see "Mapping Data Source Members to Data Target Members" on page 16-11.

For information on using attributes in partitions, see "Using Attributes in Partitions" on page 6-28. For information on mapping attributes associated with members of a base dimension, see "Mapping Attributes Associated with Members" on page 16-28.

➤ To map member names for a linked partition:

1. From the Areas page of the Partition Wizard, click Next or Mappings to move to the Mappings page.

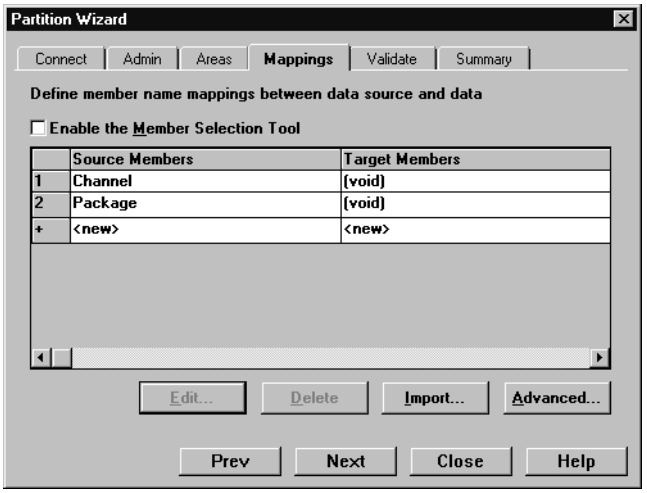


Figure 16-22: Mappings Page of the Partition Wizard

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 Mappings” on page 16-17.
   - Select the member name in the Member Select dialog box. See “Entering an Area Using the Member Selection Dialog Box” on page 16-10.
   - Import the member mappings from an external data file. See “Importing Member Mappings” on page 16-19.

Manually Defining Mappings

➤ To enter the member in the data source and the member that it maps to in the data target:

1. In the Mappings page of the Partition Wizard, make sure that the Enable the Member Selection Tool box is *cleared*.
2. Double-click the New field in the Source Members box (Figure 16-22) 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 6-38, you must also map the Package dimension to `Void`.

Using the Member Selection Dialog Box to Enter Mappings

➤ To choose the member in the data source and the member that it maps to in the data target from the Member Selection dialog box:

1. In the Mappings page of the Partition Wizard, make sure that the Enable Member Selection Tool box is checked.
2. Double-click the New field in the Source Members box (Figure 16-22) 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 6-38, 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 16-9.

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 16-40.

➤ To validate a partition:

1. From the Mappings page in the Partition Wizard, click Next or Validate to move to the Validate page.

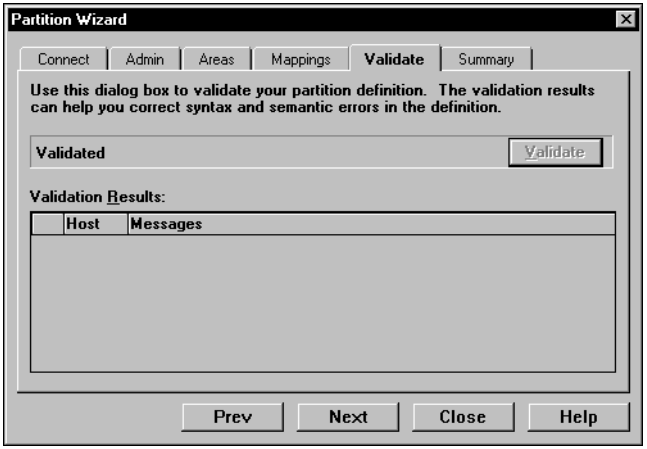


Figure 16-23: Validate Page of the Partition Wizard

2. Click Validate. Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

When Hyperion Essbase validates a partition definition, it checks on the server for the data source and the data target to ensure that:

- The area definition is valid (contains no syntax errors).
- The specified data source members are valid and map to valid members in the data target.
- All connection information is correct; that is, the server names, database names, application names, user names, and password information.
- For linked partitions, the default user name and password that you provide are correct.
- For replicated and transparent partitions, a replication target does not overlap with a replication target; a replication target does not overlap with a transparent target; a transparent target does not overlap with a transparent target; and a replication target does not overlap with a transparent target.
- For replicated and transparent partitions, the cell count for the partition is the same on the data source and the data target.
- For replicated and transparent partitions, the area dimensionality matches the data source and the data target.

Note: You must validate a transparent partition that is based on attribute values to ensure that the results are complete. Hyperion Essbase does not display an error message when results are incomplete.

Saving the Partition Definition

You can save partition definitions even if they have not been validated.

➤ To save a partition definition:

1. From the Validate page of the Partition Wizard, click Next or Summary to open the Summary page.

Figure 16-24: 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.

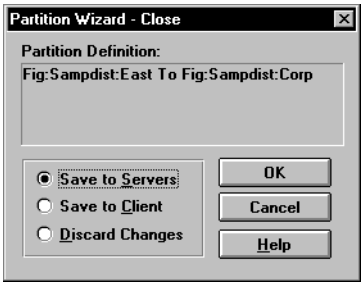


Figure 16-25: Partition Wizard Close Dialog Box

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 16-58.

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 Hyperion 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 16-38.

If, however, you need to control how Hyperion 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 Hyperion Essbase cannot determine how to map multiple members in the data source to a single member in the data target, you must logically determine how to divide your data until you can apply one mapping rule to that subset of the data. Then use that rule in the context of area-specific mapping to map the members.

Example 1: Advanced Area-Specific Mapping

The data source and data target contain the following dimensions and members:

Source	Target
Product	Product
Cola	Cola
Market	Market
East	East
Year	Year
1998	1998
1999	1999
	Scenario
	Actual
	Budget

The data source does not have a Scenario dimension. Instead, it assumes that past data is actual data and future data is forecast, or budget, data.

You know that 1998 in the data source should correspond to 1998, Actual in the data target and 1999 in the data source should correspond to 1999, Budget in the data target. So, for example, if the data value for Cola, East, 1998 in the data source is 15, then the data value for Cola, East, 1998, Actual in the data target should be 15.

Because mapping works on members, not member combinations, you cannot simply map 1998 to 1998, Actual. You must define the area (1998 and 1998, Actual) and then create area-specific mapping rules for that area.

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 16-8.

   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 16-11.
   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 Hyperion Essbase maps the 1998 values to 1998, Actual and the 1999 values to 1999, Budget.

Example 2: Advanced Area-Specific Mapping

You can also use advanced area-specific mapping if the data source and data target are structured very differently but contain the same kind of information.

This strategy works, for example, if your data source and data target contain the following dimensions and members:

Source	Target
Market	Customer_Planning
NY	NY_Actual
CA	NY_Budget
	CA_Actual
	CA_Budget
Scenario	
Actual	
Budget	

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 16-11.

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 16-8.

   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 Hyperion 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, Hyperion Essbase maps Actual to NY_Actual for the entire Actual, NY area and Actual to CA_Actual for the entire Actual, CA area. Hyperion Essbase maps the Actual member to different members depending on the area.

Specifying Area-Specific Mapping

➤ To specify area-specific mapping:

1. Click Advanced in the Mappings tab of the Partition Wizard to open the Area Specific Member Mapping dialog box.

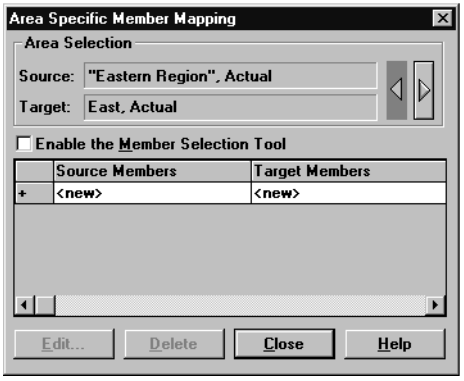


Figure 16-26: Area Specific Member Mapping Dialog Box

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, Hyperion 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, Hyperion 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 Hyperion Essbase does try to make whatever changes it can to replicated and transparent partitions when the outlines are not synchronized, Hyperion 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. Hyperion Essbase tracks changes that you make to your outlines and provides tools to make it easy to keep your outlines synchronized. This section describes the steps you need to take to synchronize your outlines.

Source Outline and Target Outline

Before you can synchronize your outlines, you must determine which outline is the source outline and which is the target outline.

- The *source outline* is the outline that outline changes are taken from.
- The *target outline* is the outline that outline changes are applied to.

By default, the source outline is from the same database as the data source; that is, outline and data changes flow in the same direction. For example, if the East database is the data source and the Company database is the data target, then the default source outline is East.

You can also use the data target outline as the source outline. You might want to do this if the structure of the outline (its dimensions, members, and properties) is maintained centrally at a corporate level, while the data values in the outline are maintained at the regional level (for example, East). This allows the database administrator to make changes in the Company outline and apply those changes to each regional outline when she synchronizes the outline.

To set the source outline, see "Specifying the Partition Type and Connection Information" on page 16-5.

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: Hyperion Essbase updates as many changes as possible to the target outline. If Hyperion 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 Hyperion 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...	Hyperion Essbase...
Make changes to the source outline	1. Records the changes in a change log file named ESS*xxxxx*.CHG, where *xxxxx* is the number of the partition. If you have more than one partition on a source outline, Hyperion 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.

When You...	Hyperion Essbase...
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. Hyperion 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, Hyperion 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.
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.

CAUTION: 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 Hyperion Essbase desktop.
2. Choose Database > Synchronize Outline to open the Synchronize Outline dialog box. Click the Help button for detailed information about each item in the Synchronize Outline dialog box.

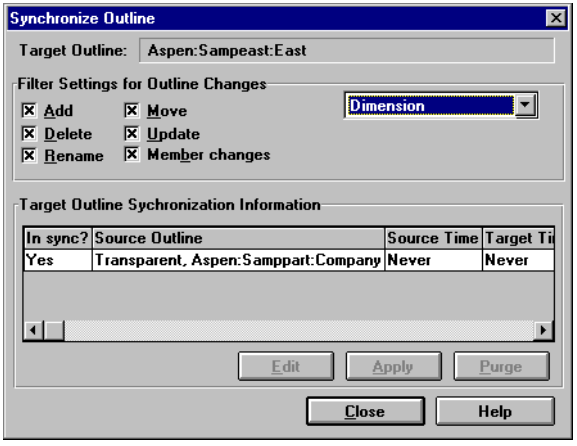


Figure 16-27: Synchronize Outline Dialog Box

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 Properties from the list box. Select the types of changes to retrieve by checking their boxes.

Note: The Apply button is enabled only when outline synchronization is necessary. If the database outlines are identical, the Apply button is not enabled.

CAUTION: 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 Hyperion 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.

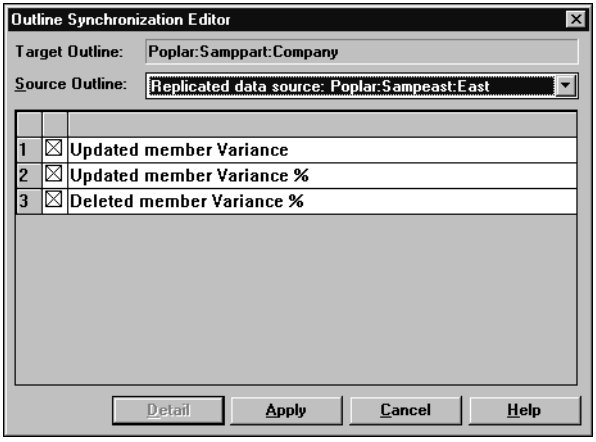


Figure 16-28: Synchronization Editor Dialog Box

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.

CAUTION: 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. Hyperion Essbase propagates the accepted change records.

d. Proceed to Step 7.

5. Click Apply to open the Apply Outline Changes dialog box.

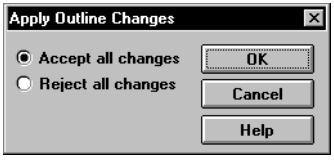


Figure 16-29: Apply Outline Changes Dialog Box

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, Hyperion 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.

   Hyperion Essbase deletes all records from the change log file that have been applied or rejected. If all records have been applied or rejected, Hyperion Essbase deletes the change log file as well. Hyperion 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.

Note: Hyperion 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 16-38.

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.

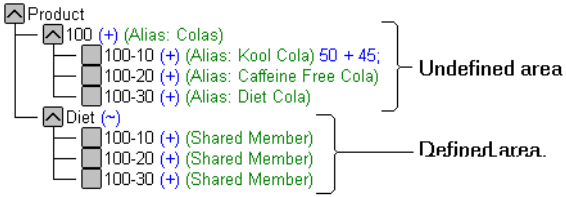


Figure 16-30: Shared Members and Outline Synchronization

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 16-30, 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 is not

propagated to the target outline because these members are not defined in the partition area and are not associated with a member that is in the defined partition area.

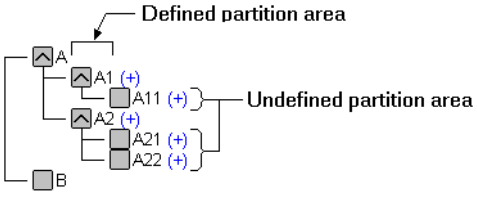


Figure 16-31: Implied Shared Members and Outline Synchronization

The reverse is true again. If A1 is not defined in the partition area and its implied shared member is, then any change to A1 is propagated to the target outline.

You can synchronize outlines using ESSCMD. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Task:	ESSCMD to Use:
Retrieve the outline change log from the source outline. You can choose specific changes from the source outline. You can choose to: • Get all changes from the outline. If you say Yes, Hyperion Essbase retrieves all changes. • Get all dimension changes. If you say Yes, Hyperion Essbase retrieves all changes and proceeds to the next step. If you say No, Hyperion 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, Hyperion Essbase retrieves all changes and proceeds to the next step. If you say No, Hyperion Essbase asks you which member changes to retrieve. You can retrieve: new members, deleted members, renamed members, or moved members. • Get all member property changes. If you say Yes, Hyperion Essbase retrieves all changes and proceeds to the next step. If you say No, Hyperion Essbase asks you which member property 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 Hyperion 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.

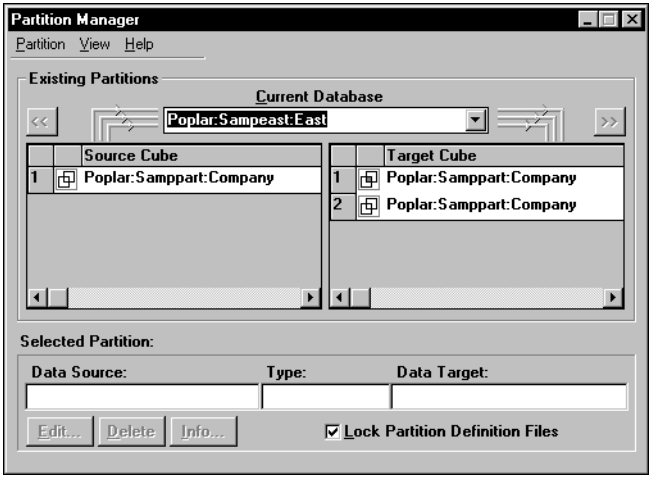


Figure 16-32: Partition Manager

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 Chapter 16, "Building and Maintaining Partitions."
Delete the partition	Delete. Hyperion Essbase deletes the partition definition from the partition's `.DDB` file on the data source and data target servers.
View more information about the partition	Info to open the Partition Information dialog box.  You can also use the PRINTPARTITIONDEFFILE command in ESSCMD. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, "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. Hyperion 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.

Hyperion 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, Hyperion 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, Hyperion Essbase updates the Budget, Actual, Variance, and Variance % members in the Samppart Company database.

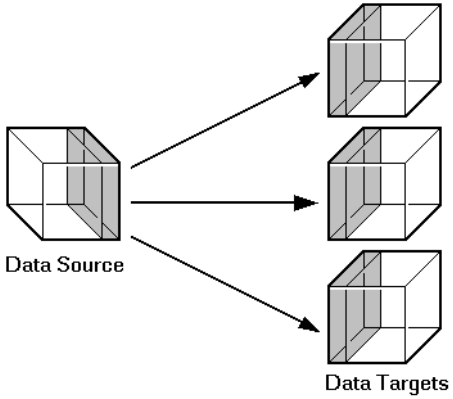


Figure 16-33: Put to Targets

- From all data sources connected to a data target. For example, if you replicate from all data sources connected to the Samppart Company database, Hyperion Essbase pulls the Budget, Actual, Variance, and Variance % members from the Sampeast East database and updates them in the Samppart Company database.

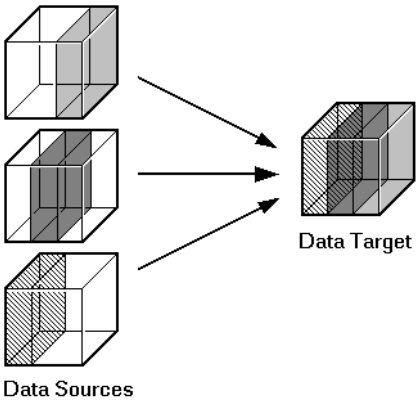


Figure 16-34: Get from Sources

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 Hyperion Essbase desktop.

   If you opened the data source, choose Database > Replicate Data > To Targets. If you opened the data target, choose Database > Replicate Data > From Sources.

This opens the Data Replication dialog box.

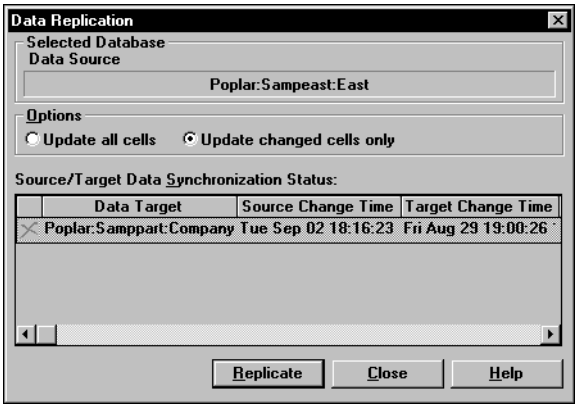


Figure 16-35: Data Replication Dialog Box

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 16-58.
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 the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Task:	ESSCMD to Use:
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.

Symptom	Possible Causes	Solutions
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.
Hyperion 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? Hyperion Essbase only propagates changes to the source outline. Does the outline change affect a defined partition? Hyperion 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 17, "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 18, "Controlling Access to Database Cells," describes how to define security filters and assign them to users.
- Chapter 19, "Security Examples," contains detailed examples that illustrate how to implement a security system for Hyperion Essbase OLAP Server applications.

Chapter 17

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 Hyperion 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 Hyperion Essbase Security System.

This chapter contains the following sections:

Multi-Layered Security and Privileges

The Hyperion 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 combine the following layers of security:

- User and Group layer: Define privileges and access levels for individual users and groups of users. When higher, these privileges take precedence over global access settings (applied to the server, applications, and databases).
- Global Access layer: Define privileges and access at the level of the server, application, or database. User and group access conforms to these global settings except where higher privileges are granted at the user and group layer.

- Database Filter layer: Define specific database access levels that users and groups can have for particular database members, down to the individual data value (cell). To learn more about the Database Filter layer, see Chapter 18, “Controlling Access to Database Cells.”

The ESSBASE.SEC Security File

All information about users, groups, passwords, privileges, filters, applications, databases, and their corresponding directories is stored in the `ESSBASE.SEC` file in your `$ARBORPATH\Bin` directory. Each time you successfully start the Agent, a backup copy of the security file is created as `ESSBASE.BAK`.

If you attempt to start the Agent and can’t get a password prompt or your password is rejected, no `.BAK` file is created. You can restore from the last successful startup by copying `ESSBASE.BAK` to `ESSBASE.SEC`. Both files are in the `BIN` directory where you installed the Hyperion Essbase server.

Privileges Available at Global and User Levels

Various types of management privileges can be assigned at 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. When higher, 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 17-1: 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]

Table 17-1: Global and User Management Privileges (Continued)

	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)
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. Groups are collections of users that share the same minimum privileges. Users inherit all privileges of the group, and can additionally have access to privileges that exceed those of the group. Users and groups are managed on a server–by–server basis: users defined on a server exist for all applications and databases on the server.

User and Group Types

One major way to assign privileges to users and groups is to define user and group 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 17-6 and Figure 17-8, respectively).

In the Application Manager, users and groups can have one of four types 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 17-9.

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 Hyperion Essbase on the server is designated the Hyperion Essbase System Supervisor for that server. Hyperion 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.

Figure 17-1: 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.

Figure 17-2: 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.

Figure 17-3: 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 17-29.

Figure 17-4: 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 17-35 or “Database Designer Privilege” on page 17-36.

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.

   Hyperion Essbase displays the following dialog box:

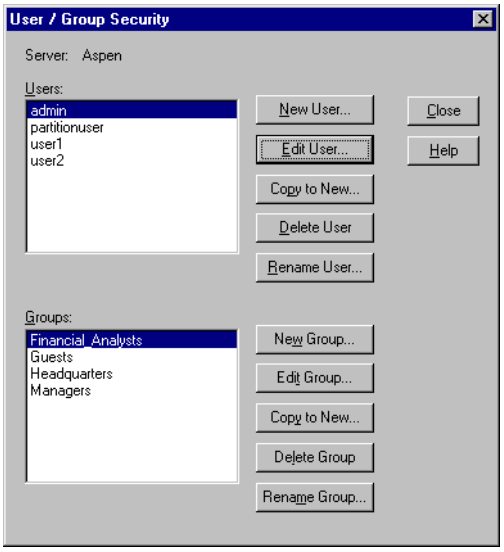


Figure 17-5: User/Group Security Dialog Box

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 to the right of each list box, which are displayed, 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 17-9, “Copying an Existing Security Profile” on page 17-16, “Deleting Users and Groups” on page 17-19, or “Renaming Users and Groups” on page 17-21.

For information about lock management and password/user name management, see “Managing User Activity at the Server Level” on page 17-36.

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 the `DOCS` directory for information about these commands. See Chapter 44, “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 define 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 18, “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. Hyperion Essbase displays the following dialog box:

Figure 17-6: 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, Hyperion 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 are not 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.)

   At the next login attempt, the user is prompted to change the password in the Change Password dialog box, shown in Figure 17-9.
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 type 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 17-5. 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.

   Hyperion Essbase displays the following dialog box:

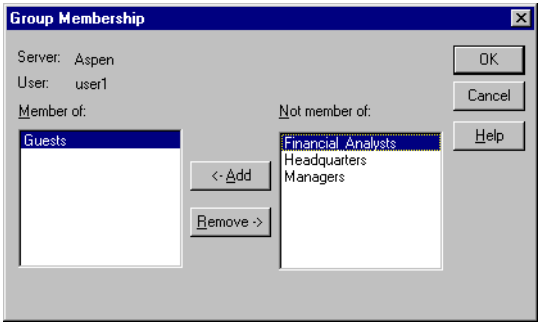


Figure 17-7: Group Membership Dialog Box

   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 17-31 and "Defining Global Database Access" on page 17-34 for more information.

11. Click OK to add the new user to the server. Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, “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 the `DOCS` directory for information about these commands. See Chapter 44, “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.

   Hyperion Essbase displays the Edit User dialog box:

Figure 17-8: Edit User Dialog Box

2. To change the user's password, enter the new password in the Password text box. Hyperion 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 are not 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:

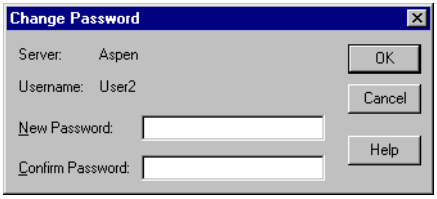


Figure 17-9: Change Password Dialog Box

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. Hyperion 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 17-21.

To change a user's password, you can also use the SETPASSWORD command in ESSCMD. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, "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.

   Hyperion Essbase displays the User/Group Security dialog box (see Figure 17-5).

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 17-21.

Hyperion Essbase displays the following dialog box:

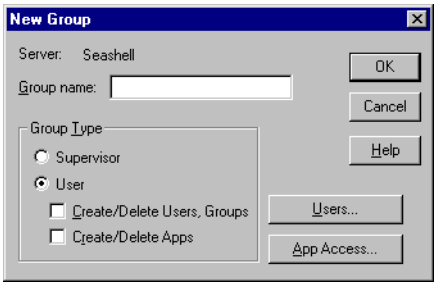


Figure 17-10: New Group Dialog Box

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.

     Note: You cannot add users to a group having higher privileges than your own.

   - For more information on using the Group Membership dialog box to assign users to groups, click Help or see the instructions accompanying Figure 17-7.
   - 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 17-31 and "Defining Global Database Access" on page 17-34 for more information.
7. Click OK to add the new group to the server. Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, "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 the `DOCS` directory for information about this command. See Chapter 44, "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.

   Hyperion Essbase displays the User/Group Security dialog box (see Figure 17-5).

2. Select the name of the user whose profile you want to copy.

3. Click Copy to New.

   Hyperion Essbase displays the following dialog box:

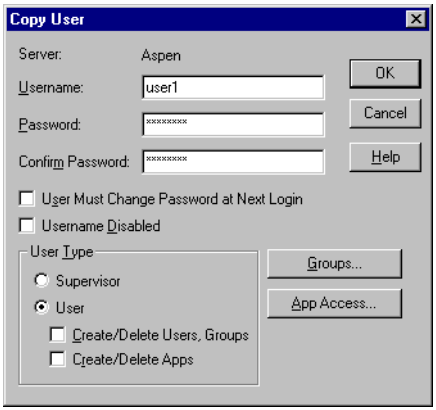


Figure 17-11: Copy User Dialog Box

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, Hyperion Essbase masks your typing with asterisks.

6. Retype the user's password in the Confirm Password text box. You must type the password exactly as you did in the Password text box.

   Note: Passwords are not case-sensitive.

7. To force the new user to change 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. Hyperion 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.

   Hyperion Essbase displays the User/Group Security dialog box (see Figure 17-5).

2. Select the name of the group you want to copy.

3. Click Copy to New.

   Hyperion Essbase displays the following dialog box:

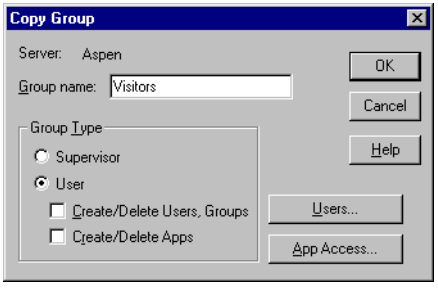


Figure 17-12: Copy Group Dialog Box

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 17-7.
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. Hyperion Essbase updates the server security file with your changes.

Deleting Users and Groups

➤ To delete a user:

1. Choose Security > Users/Groups.

   Hyperion Essbase displays the User/Group Security dialog box (see Figure 17-5).
2. Select the name of the user you want to delete in the Users list box.
3. Click Delete User.

   Hyperion Essbase displays the following confirmation box:

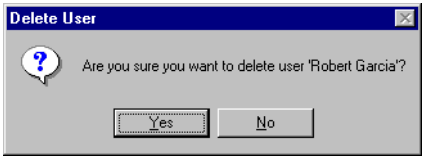


Figure 17-13: Delete User Confirmation Box

4. Click Yes to delete the user, or click No to cancel the delete operation.

   If you choose to delete the user, Hyperion Essbase updates the Users list box and the server security file with your changes. Hyperion Essbase automatically deletes users from all groups to which they belong.

➤ To delete a group:

1. Choose Security > Users/Groups.

   Hyperion Essbase displays the User/Group Security dialog box (see Figure 17-5).

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, Hyperion Essbase displays the following confirmation box:

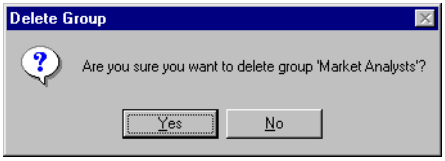


Figure 17-14: Delete Group Confirmation Box

4. Click Yes to delete the group, or click No to cancel the delete operation.

   If you choose to delete the group, Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Renaming Users and Groups

➤ To rename a user:

1. Choose Security > Users/Groups.

   Hyperion Essbase displays the User/Group Security dialog box (see Figure 17-5).

2. Select the name of the user in the Users list box.

3. Click Rename User.

   Hyperion Essbase displays the following dialog box:

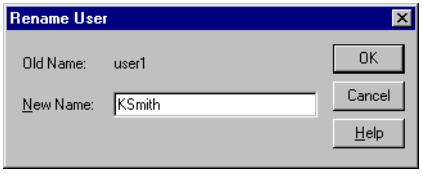


Figure 17-15: Rename User Dialog Box

4. Type the new name for the user in the New Name text box.

5. Click OK to rename the user.

   Hyperion 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.

   Hyperion Essbase displays the User/Group Security dialog box (see Figure 17-5).

2. Select the name of the group in the Groups list box.

3. Click Rename Group.

   Hyperion Essbase displays the following dialog box:

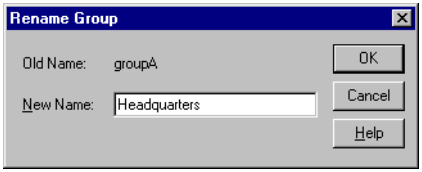


Figure 17-16: Rename Group Dialog Box

4. Type the new name for the group in the New Name text box.
5. Click OK to rename the group.

   Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, “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 Application 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.)

   Hyperion Essbase displays the following dialog box:

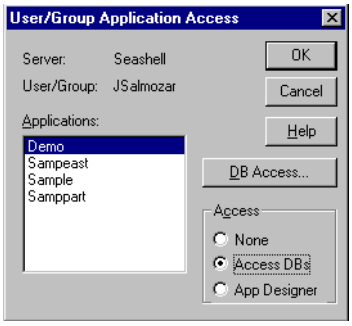


Figure 17-17: User/Group Application Access Dialog Box

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 17-24.

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 17-1).
- The database in question does not allow users sufficient access through its global settings (see "Minimum Database Access" on page 17-32).
- The users have no access granted to them through filters (see Chapter 18, "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 17-17). 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. Hyperion Essbase displays the User Database Access dialog box:

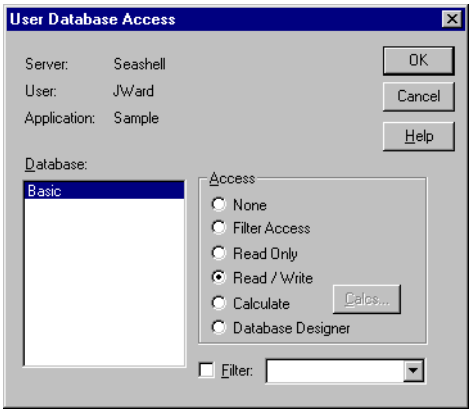


Figure 17-18: User Database Access Dialog Box

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 18, “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, Hyperion Essbase objects.

User Access Level	Privilege Description
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 18, “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 an Hyperion Essbase Spreadsheet Add-in. When you click Calcs, Hyperion Essbase displays the following dialog box:

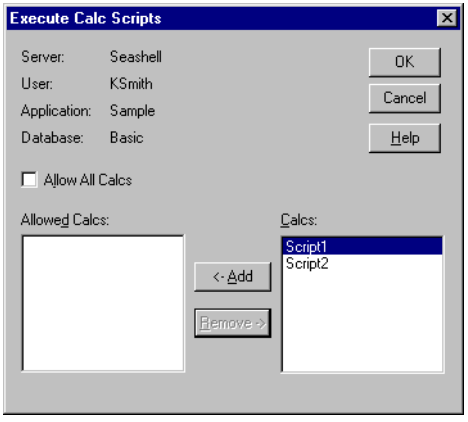


Figure 17-19: Execute Calc Scripts Dialog Box

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.

   Hyperion Essbase displays the following dialog box, showing all users and groups on the server:

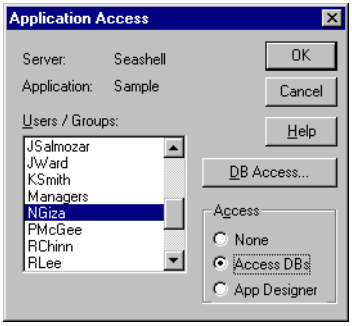


Figure 17-20: Application Access Dialog Box

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 17-28.

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. Hyperion Essbase displays the Database Access dialog box:

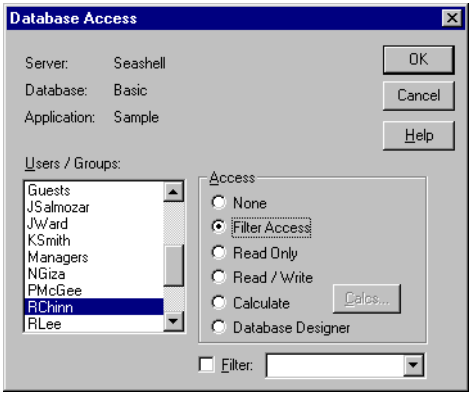


Figure 17-21: Database Access Dialog Box

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 does not show up in the spreadsheet or shows up 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 17-22.

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 Hyperion Essbase files, retrieve and update data values, and run report scripts. Write access does not permit calculations or outline modifications.
Calculate	Specifies Calculate and update access to any data value in the 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.)

Hyperion Essbase displays the following dialog box:

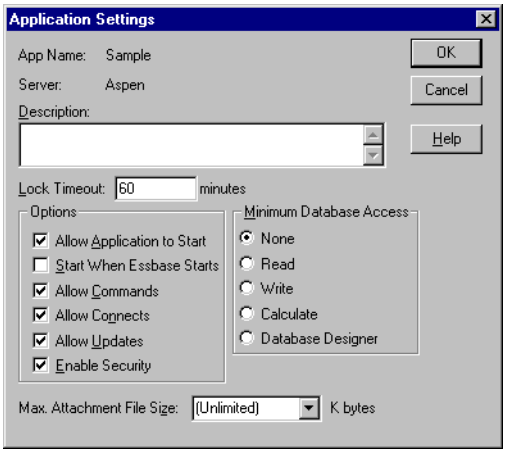


Figure 17-22: Application Settings Dialog Box

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 17-22), 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 18, “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.

CAUTION: *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 Hyperion Essbase server to improperly disconnect from the Hyperion Essbase client, and your application is no longer accessible, you will need to shut down and restart the application. See Chapter 45, "Running Hyperion 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 17-32). In addition, any database within an application can be defined with its own higher Global Access settings, which 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.)

   Hyperion Essbase displays the following dialog box:

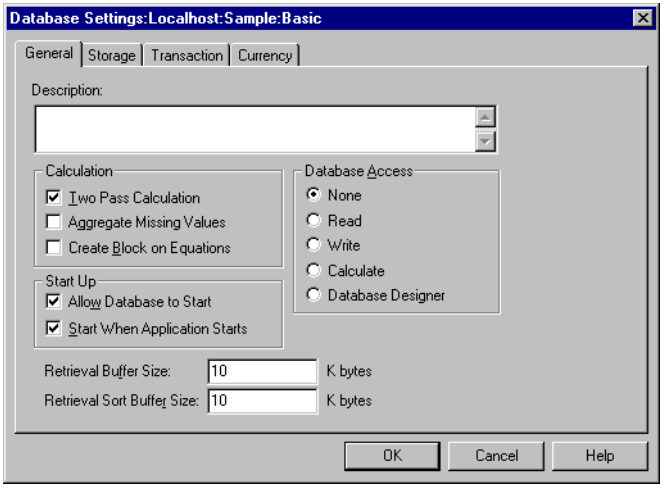


Figure 17-23: Database Settings Dialog Box

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 supervisor 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

Hyperion Essbase Spreadsheet Add-in users can interactively send data from a spreadsheet to the server. To maintain data integrity while providing multi-user concurrent access, Hyperion Essbase lets users lock data for the purpose of updating it. Users who want to update data must first lock the records to prevent other users from trying to change the same data.

The default maximum lock time is 3600 seconds, or 60 minutes. To prevent data from becoming inaccessible for long periods, Hyperion 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.

   Hyperion Essbase displays the Database Locks dialog box:

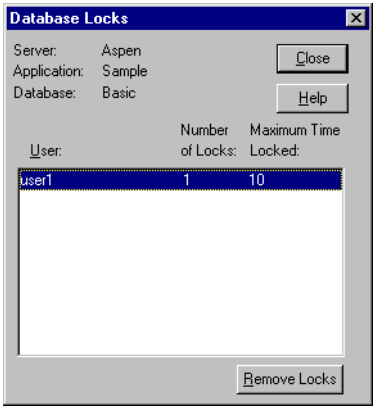


Figure 17-24: Database Locks Dialog Box

   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 the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Disconnecting Users

The security system lets you disconnect a user from the Hyperion 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.

   Hyperion Essbase displays the Connections dialog box:

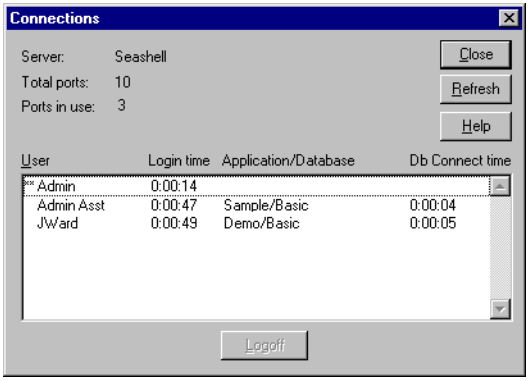


Figure 17-25: Connections Dialog Box

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 display 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 Hyperion 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.

   Hyperion Essbase displays the Server Settings dialog box.

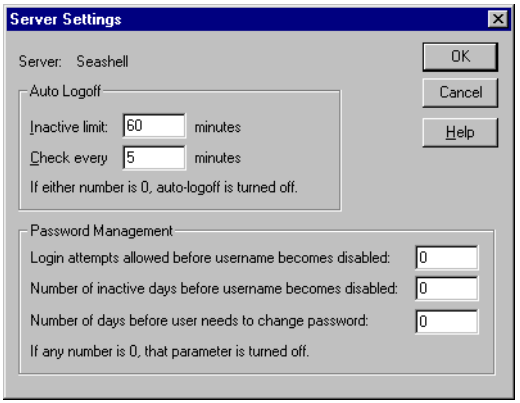


Figure 17-26: Server Settings Dialog Box

   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, Hyperion 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 the 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 17-39), 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 17-12.

1. To view or activate currently disabled user names, choose Security > Disabled Usernames.

   Hyperion Essbase displays the Disabled Usernames dialog box, which lists all disabled user names:

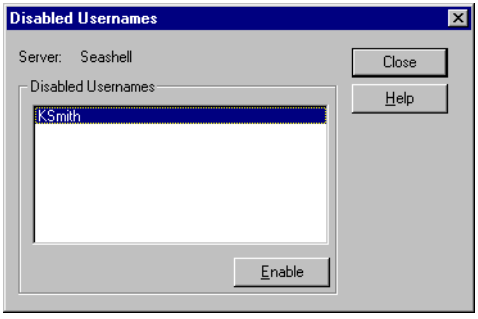


Figure 17-27: Disabled Usernames Dialog Box

2. To activate a user, select the user name from the list box and click Enable.

   Hyperion Essbase displays a confirmation box.

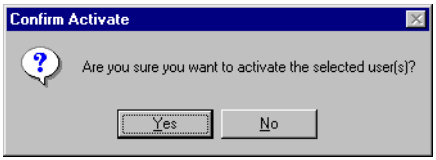


Figure 17-28: Confirm Activate Confirmation Box

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 18

Controlling Access to Database Cells

When the security levels defined for applications, databases, users, and groups are not enough, the Hyperion Essbase 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 applications or databases. For more information about user privileges, see Chapter 17, “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 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 Hyperion 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 17, “Managing Security at Global and User Levels.”

Calculation access is controlled by global access privileges or individually assigned user 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. Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, “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 “Mini-Tutorial” on page 18-6.

2. Choose Security > Filters.

   Hyperion Essbase displays the Filters dialog box.

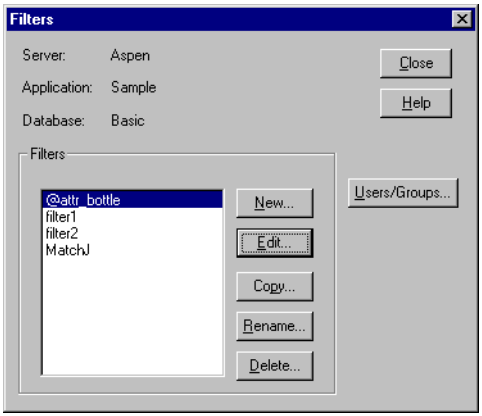


Figure 18-1: Filters Dialog Box

3. Click New.

   Hyperion Essbase displays the following confirmation box.

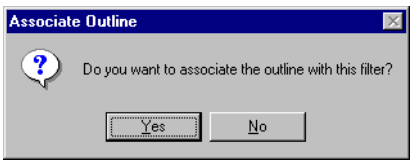


Figure 18-2: Associate Outline Confirmation Box

4. Click Yes to confirm that you want to associate the current outline with the filter. This will enable the member selection tool so that you don't have to type in all member specifications.

   Hyperion Essbase displays the Define Filter dialog box.

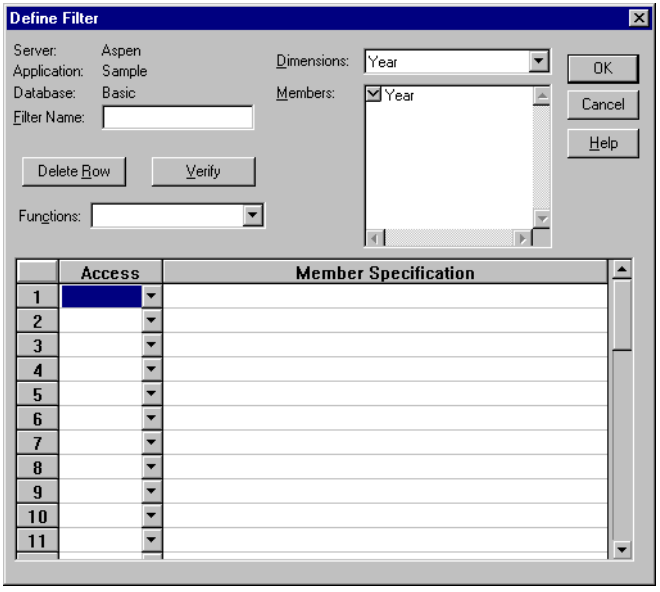


Figure 18-3: Define Filter Dialog Box

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 18-3.
   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 function to dimensions and members:
   a. Position the cursor in the Member Specification row and select the appropriate function from the Functions list box.
   b. Specify a member for the function by placing the cursor within the parentheses and selecting the member name from the outline in the Members list box.

      The member name should appear within the parentheses.
9. To delete a row, place the insertion point in the row and click Delete Row.
10. To verify that your syntax is correct for the entire filter sheet, click Verify.

    Note: For more information about functions and syntax, consult the online *Technical Reference* in the `DOCS` directory.

11. Click OK to save the filter.

Mini-Tutorial

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. Hyperion Essbase displays the Filters dialog box (Figure 18-1).

3. Click New.

   Hyperion Essbase displays the following confirmation box.

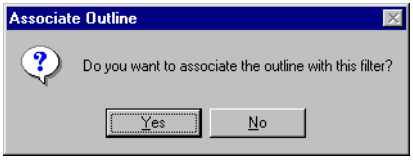


Figure 18-4: Associate Outline Confirmation Box

4. Click Yes to confirm that you want to associate the current outline with the filter.

   Hyperion Essbase displays the Define Filter dialog box (see Figure 18-3).

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 18-5: Sample Member Specifications in the Define Filter Dialog Box

See the instructions for creating a filter (see “Creating a New Filter” on page 18-3).

This filter defines the following access plan:

- No retrieval or update access to any data for members of the Total Expenses branch or the Ratios branch.
- Read-only access to all members of the Inventory branch below Actual.
- Read-only access to all members of the Inventory branch below Budget for the months of Jan through Jun.
- Write access to all members of the Inventory branch below Budget data for months of Jul through Dec.

Filtering Whole Members vs. Filtering Member Combinations

The following examples illustrate different ways to control access to database cells. Data can be protected by filtering entire members, or by filtering member combinations.

Filtering members separately, by defining access for each member in a separate row of the Define Filter dialog box, affects whole regions of data for those members. Filtering member combinations, using one row in the Define Filter dialog box, affects data at the member intersections.

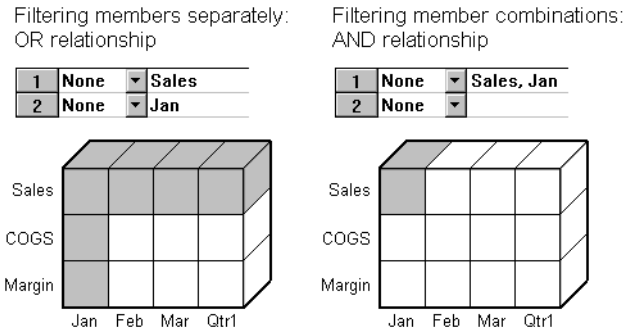


Figure 18-6: How Filters Affect Data: AND/OR Relationships

Filtering Members Separately

To filter all the data for one or more members, define access for each member on its own row in the Define Filter dialog box. Filter definitions on separate rows of a filter are treated with an OR relationship.

Example:

User KSmith is assigned the following filter preventing any access to the members Sales or Jan in the Sample Basic database:

	Access	Member Specification
1	None	Sales
2	None	Jan

Figure 18-7: Filter Blocking Access to Sales or Jan

The next time user KSmith connects to Sample Basic, she has no access to data values for the member Sales or for the member Jan. Her spreadsheet view of the profit margin for Qtr1 looks like the following:

	A	B	C	D	E
1				Product	Market
2	Sales	Jan	Scenario	#NoAccess	
3		Feb	Scenario	#NoAccess	
4		Mar	Scenario	#NoAccess	
5		Qtr1	Scenario	#NoAccess	
6	COGS	Jan	Scenario	#NoAccess	
7		Feb	Scenario	14307	
8		Mar	Scenario	14410	
9		Qtr1	Scenario	42877	
10	Margin	Jan	Scenario	#NoAccess	
11		Feb	Scenario	17762	
12		Mar	Scenario	17803	
13		Qtr1	Scenario	52943	

Figure 18-8: Results of Filter Blocking Access to Sales or Jan

All data for Sales is blocked from view, as well as all data for January, inside and outside of the Sales member. Data for COGS (Cost of Goods Sold), a sibling of Sales and a child of Margin, is available, with the exception of COGS for January.

Filtering Member Combinations

To filter data for member combinations, define the access for each member combination using a single row in the Define Filter dialog box. A filter definition using one row and a comma is treated as an AND relationship.

Example:

User RChinn is assigned the following filter which blocks only the intersection of the members Sales and Jan in the Sample Basic database:

	Access	Member Specification
1	None	Sales, Jan
2		

Figure 18-9: Filter Blocking Access to Sales for Jan

The next time user RChinn connects to Sample Basic, she has no access to the data value at the intersection of members Sales and Jan. Her spreadsheet view of the profit margin for Qtr1 looks like the following:

	A	B	C	D	E
1			Product	Market	Scenario
2	Sales	Jan	#NoAccess		
3		Feb	32069		
4		Mar	32213		
5		Qtr1	95820		
6	COGS	Jan	14160		
7		Feb	14307		
8		Mar	14410		
9		Qtr1	42877		
10	Margin	Jan	17378		
11		Feb	17762		
12		Mar	17803		
13		Qtr1	52943		
14					

Figure 18-10: Results of Filter Blocking Access to Sales, Jan

Sales data for January is blocked from view. However, Sales data for other months is available, and non-Sales data for January is available.

Filtering with Attribute Functions

You can use filters to restrict access to data for base members sharing a particular attribute. To filter data for members with particular attributes defined in an attribute dimension, use the attribute member in combination with the @ATTRIBUTE function or the @WITHATTR function.

Note: @ATTRIBUTE and @WITHATTR are member set functions. Most of the member set functions can be used in filter definitions. For more information about functions, see the online *Technical Reference* in the `DOCS` directory.

Example

User PJones is assigned the following filter, which blocks access to data for caffeine-free products. “Caffeinated_False” is a Boolean-type attribute member in Sample Basic, in the Pkg Type attribute dimension. This attribute is associated with members in the base dimension Product.

	Access	Member Specification
1	None	@ATTRIBUTE("Caffeinated_False")
2		

Figure 18-11: Filter Blocking Access to Members with Attribute “Caffeinated_False”

The next time user PJones connects to Sample Basic, he has no access to the data values for any base dimension members associated with Caffeinated_False. His spreadsheet view of first-quarter cola sales in California looks like the following:

	Sales	California	Qtr1	Actual
Cola	1998			
Diet Cola	367			
Caffeine Free Cola	#Miss			
Colas	2767			

Figure 18-12: Results of Filter Blocking Access to Caffeine-free Products

Sales data for Caffeine Free Cola is blocked from view. Note that Caffeine Free Cola is a base member, and Caffeinated_False is an associated member of the attribute dimension Caffeinated (not shown in the above spreadsheet view).

Editing a Filter

1. To edit an existing filter, choose Security > Filters. Hyperion Essbase displays the Filters dialog box (see Figure 18-1).
2. Select the filter you want to edit and click Edit. Hyperion Essbase displays the Associate Outline Confirmation box, as in Figure 18-2.
3. Click Yes to confirm that you want to associate the current outline with the filter. Hyperion Essbase displays the Define Filter dialog box, as in Figure 18-3.

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 18-3.
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 18-1).
2. Select the filter you want to copy and click Copy. Hyperion 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.

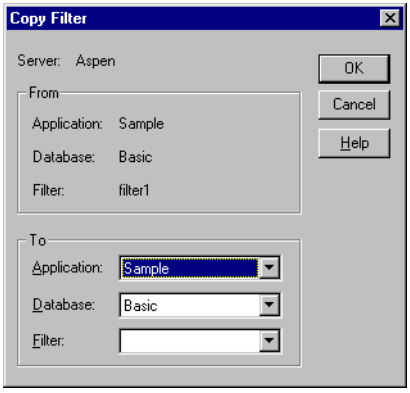


Figure 18-13: Copy Filter Dialog Box

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. Hyperion Essbase displays the following confirmation box:

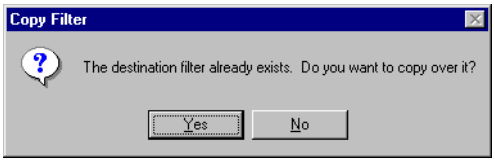


Figure 18-14: Copy Filter Confirmation Box

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 the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Renaming a Filter

1. Choose Security > Filters.

   Hyperion Essbase displays the Filters dialog box (see Figure 18-1).

2. Select the filter you want to rename and click Rename.

   Hyperion Essbase displays the Rename Filter dialog box.

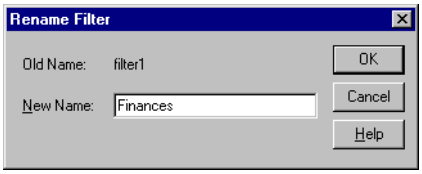


Figure 18-15: Rename Filter Dialog Box

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 the `DOCS` directory for information about this command. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Deleting a Filter

1. Choose Security > Filters.

   Hyperion Essbase displays the Filters dialog box (see Figure 18-1).

2. Select the filter you want to delete and click Delete. Hyperion Essbase displays the following confirmation box:

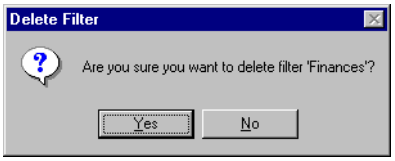


Figure 18-16: Delete Filter Confirmation Box

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 18-1.

2. Select the filter name you want to assign and click Users/Groups.

   Hyperion Essbase displays the Assign Filters dialog box:

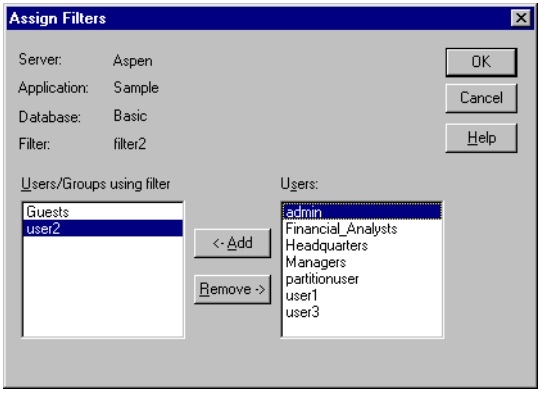


Figure 18-17: Assign Filters Dialog Box

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 18-18: 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 18-19: 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 18-20: 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 19, “Security Examples.”

Chapter 19

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 17, "Managing Security at Global and User Levels."

Security Problem 1

Three employees need to use Hyperion 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 17-6 in Chapter 17, "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 Hyperion 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 17-6 in Chapter 17, "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 17-22 in Chapter 17, "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 Hyperion 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 17-6 in Chapter 17, "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 17-22 in Chapter 17, "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 Hyperion 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 17-6 in Chapter 17, "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 17-22 in Chapter 17, “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 can allow the users to complete their updates or clear them directly with the Remove Locks option.

3. After confirming that no users have active locks, proceed to perform maintenance on the application.

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 20, “Introducing Data Loading,” introduces you to the parts of data loading, including external data sources, rules files, free-form data loading, and describes how Hyperion Essbase handles security and multi-user issues while loading data.
- Chapter 21, “Setting up a Rules File to Manipulate Records,” describes how to create a rules file for a particular data source, specify record manipulations, and validate and save the rules file using the Data Prep Editor.
- Chapter 22, “Manipulating Fields Using a Rules File,” describes how to manipulate fields using a rules file, including how to ignore fields, order fields, map fields to member names, and change data values using the Data Prep Editor.
- Chapter 23, “Performing 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 24, “Debugging and Optimizing Data Loads,” describes how to debug problems with data loads and how to optimize your data loads so that Hyperion Essbase can load the data more quickly.

Chapter

20

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 Hyperion 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 20-1, 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.

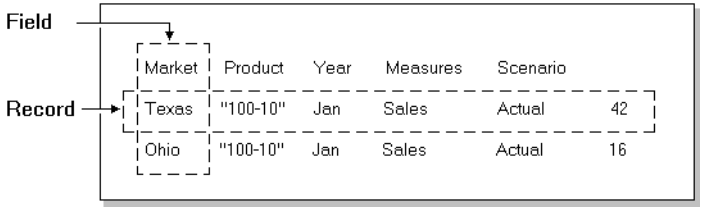


Figure 20-1: Records and Fields

As illustrated in Figure 20-2, 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.

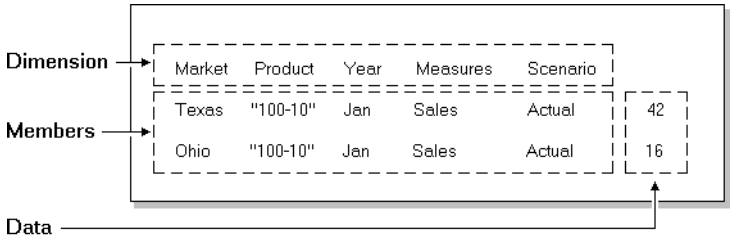


Figure 20-2: Kinds of Fields

How Does Hyperion Essbase Read a Data Source?

Hyperion Essbase reads data sources starting at the top and proceeding from left to right. To load a data value successfully, Hyperion Essbase must encounter one member from each dimension before encountering the data value. For example, in Figure 20-2, Hyperion Essbase loads the data value 42 into the database with the members Texas, 100-10, Jan, Sales, and Actual. If Hyperion 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 Hyperion 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 20-8.

- 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 20-18.

  Note: You must use a rules file to load SQL data and to build dimensions and members dynamically.

When Hyperion Essbase loads data from external sources:

1. Hyperion Essbase reads the external data source. You must format the external data source carefully.
2. If you are using a rules file, Hyperion Essbase transforms the data to match the Hyperion 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. Hyperion 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 20-2, for example, 42 is a data field. It is the dollar sales of 100-10 (Cola) sold in Texas in January.

Hyperion 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 22-15.

Valid Member Fields

A *member field* contains the name of a member or alias in a dimension. In Figure 20-2, 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 is a Dynamic Calc member, you cannot load data into it. Instead, load data into Qtr1, Qtr2, Qtr3, and Qtr4, which are not Dynamic Calc members.
A member name must be enclosed in quotes if it contains any of the following: • White space • Numeric characters (0–9) • Dashes (minus signs, hyphens) • Plus signs • & (ampersands)	For files that free form load into the Sample Basic database, the 100-10 product member name must be in quotes: "100-10" For rules on naming dimensions and members, see “Rules for Naming Dimensions and Members” on page 8-15.
If a member field maps to an alias, Hyperion Essbase uses the current alias table.	Default is the name of the default alias table.

Invalid Member or Data Fields

When Hyperion Essbase encounters an invalid member or data field, it stops the data load. Hyperion 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, Hyperion Essbase stops the data load when it encounters the 15- data value. Hyperion 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 20-3: Invalid Data Field

For information on continuing the load, see “Loading the Error Log File” on page 24-5.

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. Hyperion Essbase ignores excess delimiters in free-form data sources.

In Figure 20-4, for example, the fields are separated by spaces. Hyperion 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 20-4: File Delimiters

For more information, see “Setting File Delimiters” on page 21-10.

Ignored Characters

Some characters are in the data source for formatting reasons only. For that reason, Hyperion 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, Hyperion Essbase ignores the equal signs in Figure 20-5, but loads the other fields normally.

```
East Actual "100-10"
        Sales    Marketing
        =====    =========
Jan     10       8
Feb     21       16
```

Figure 20-5: Ignoring Formatting Characters During Loading

Introduction to Rules Files

Data load rules are a set of operations that Hyperion 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 Hyperion Essbase database during loading.

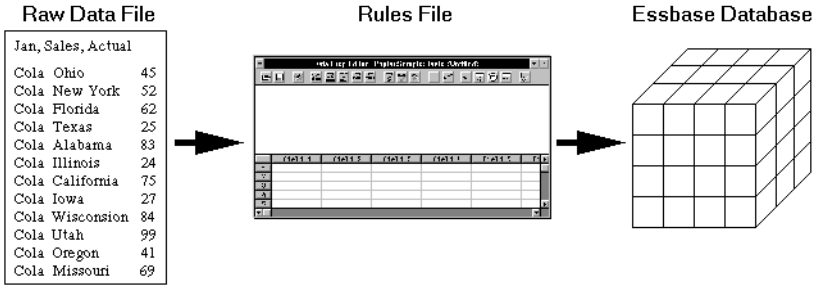


Figure 20-6: Loading Data Sources through Rules Files

Data load rules are stored in rules files. Hyperion Essbase loads the data in the data source into the database through the rules file without changing the data source. You can re-use a rules file with any data source that requires the same set of data loading rules.

You also use rules files in dimension build operations to add or change members and dimensions in outlines. For information about creating rules files and defining them for dimension build operations, see Chapter 13, “Introducing Dynamic Dimension Building” and Chapter 14, “Building Dimensions Using a Rules File.”

When to Use Data Load Rules

Use data load rules when the data load should:

- Ignore fields or strings in the data source.
- Change the order of fields by moving, joining, splitting, or creating them.
- Map the data in the data source to the database by changing strings.

- Change the data values in the data source by scaling data values or adding them to existing values in the data source.
- Set header records for missing values.
- Reject an invalid record and continue loading data.
- Add new dimensions and members to the database.
- Change existing dimensions and members in the database.

See Chapter 21, “Setting up a Rules File to Manipulate Records,” and Chapter 22, “Manipulating Fields Using a Rules File,” for information about manipulating fields and records. See Chapter 13, “Introducing Dynamic Dimension Building,” for information about changing or adding members and dimensions.

How to Create Data Load Rules

You create data load rules using the following process:

1. Select the data source in the Data Prep Editor. For example, you can select an SQL data source, a spreadsheet, or a text file. See “Selecting the Data Source” on page 21-1.
2. Set the file delimiter for your data source. For example, you can select tabs, commas, or spaces. See “Setting File Delimiters” on page 21-10.
3. Perform operations on records. For example, you can set up header records, and define select and reject operations. See “Setting up a Rules File to Manipulate Records” on page 21-1.
4. Perform operations on fields. For example, you can split them, join them, and create new ones. See “Manipulating Fields Using a Rules File” on page 22-1.
5. Map fields in the data source to dimensions and members in the database. See “Mapping Fields to Member Names” on page 22-12.
6. Save and validate the data load rules. For more information, see “Validating and Saving Data Load Rules” on page 21-21.

How Does Hyperion Essbase Execute Operations in a Rules File?

When Hyperion Essbase loads data using a rules file, it executes the operations in the rules file in the following order:

1. Hyperion Essbase sets all file delimiters, including fixed-width columns. For more information, see “Setting File Delimiters” on page 21-10.
2. Hyperion 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 22-5.

   If you’re not sure in what order the field operations were defined, select Options > Data File Properties and click the Field Edits tab. The Data File Properties dialog box appears, listing all the field operations.

   The rules file in Figure 20-7, for example, contains move, split, and join operations.

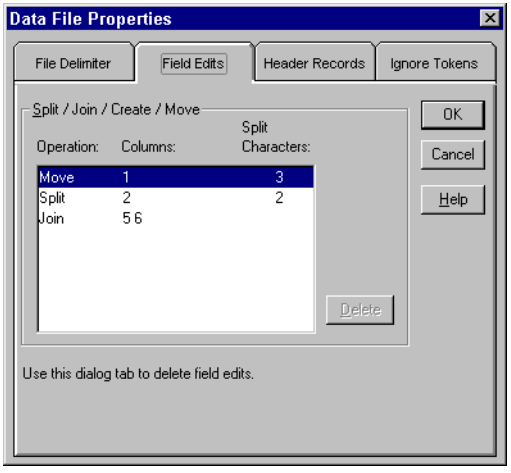


Figure 20-7: Field Operations

3. Hyperion Essbase applies all properties for each field, applying all of the properties to field1 before proceeding to field2. Hyperion Essbase applies field properties in the following order:

   a. Ignores fields set to ignore during data load.

   b. Ignores fields set to ignore during outline update.

   c. Flags the data field.

   d. Applies field names.

   e. Applies field generations.

   f. Performs all replaces in the order they are defined in the rules file. If you're not sure what order the replace operations are in, select Field > Properties and click the Global Properties tab. The Field Properties dialog box appears, listing all replace operations.

   g. Drops leading and trailing white spaces.

   h. Coverts spaces to underscores.

   i. Applies suffix and prefix operations.

   j. Scales data values.

   k. Converts text to lowercase.

   l. Converts text to uppercase.

   For more information, see Chapter 22, "Manipulating Fields Using a Rules File."

4. If you choose to skip lines, Hyperion Essbase skips the number of lines that you specified, otherwise Hyperion Essbase proceeds to the first record. For more information, see "Defining Header Information in the Rules File" on page 21-11.

5. Hyperion Essbase performs selection or rejection criteria in the order that they are defined in the rules file. Hyperion Essbase loads or doesn't load individual records in the data source based on these criteria specified in the rules file.

   If you're not sure in what order the selection or rejection criteria are defined, select Record > Select or Record > Reject. The Select Record or Reject Record dialog box displays, listing all the selection or rejection operations.

   The rules file in Figure 20-8, for example, contains a selection criterion.

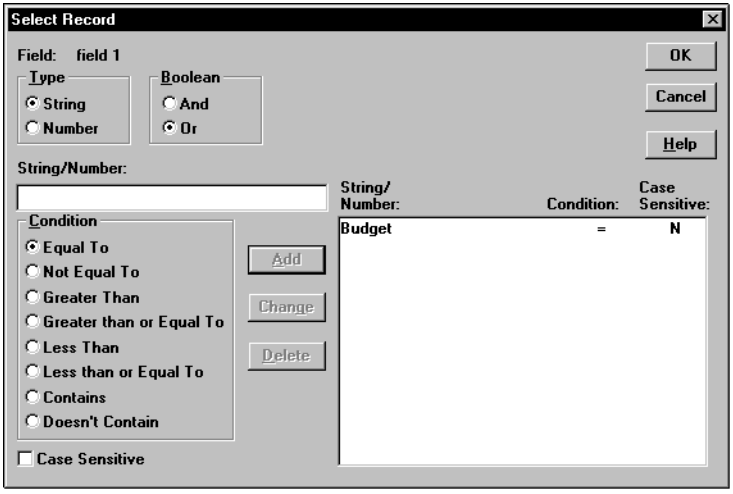


Figure 20-8: Selecting Records

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

Hyperion 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 21-11.

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 20-9 to the Sample Basic database, Hyperion 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 20-9: Valid Missing Members

Hyperion Essbase stops the data load if no prior records contain a value for the missing member field. If you tried to load Figure 20-10 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 20-9) is not specified.

```
Jan, Sales, Actual
           Cola          25
           "Root Beer"   50
           "Diet Cola"   19
```

Figure 20-10: Invalid Missing Members

For information on restarting the load, see "Loading the Error Log File" on page 24-5.

Rules for Member Fields

After Hyperion 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 20-11: 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 20-12, 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 20-12: Quoted Member Names

Unknown Member Names

If a member field contains an unknown member name, Hyperion 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, Hyperion Essbase continues to the next record. If there are no prior records, the data load stops.

For example, when you load Figure 20-13 into the Sample Basic database, Hyperion Essbase rejects the record containing Ginger Ale because it is not a valid member name. Hyperion 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 20-13: 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 13, “Introducing Dynamic Dimension Building.”

For information on restarting the load, see “Loading the Error Log File” on page 24-5.

Rules for Data Fields

After Hyperion Essbase identifies all dimensions and maps the member fields into the appropriate members in the outline, it loads the data fields to the Hyperion Essbase database. The data source or rules file must contain enough information for Hyperion Essbase to determine where to put the data. To Hyperion Essbase, data are the numbers stored for each intersection in the database. In Figure 20-2, 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 20-14 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 20-14: Setting Data Fields

To identify a column as a data field, see “Defining a Column as a Data Field” on page 22-21.

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 Hyperion Essbase knows where to put the data.

```
             Jan, Actual
Cola         East    Sales   100
"Root Beer"  East    Sales   200
```

Figure 20-15: 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 20-14, 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`), Hyperion Essbase does not change the existing data value in the database. Hyperion 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 22-15.

Rules for Extra Fields

If the data source contains fields that you don't want to load into the database, you can tell Hyperion Essbase to ignore those fields. For example, the Sample Basic database has five standard dimensions into which you would load data: Year, Product, Market, Measures, and Scenario. If the data source had an extra field, such as Salesperson, that isn't a member of any dimension, tell Hyperion 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 22-6.

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 20-16, 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 20-16: Missing Fields

Figure 20-17 is valid because `#MI` was inserted to replace the missing field.

```
Actual Ohio Sales Cola
Jan     Feb    Mar    Apr
10      15     20     #MI
```

Figure 20-17: 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 20-18 into the Sample Basic database using a rules file, it would fail. Hyperion Essbase reads the extra comma between East and Cola in the first record as an extra field. Hyperion Essbase then puts Cola into Field 3. In the next record, however, Cola is in Field 2. Hyperion 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 20-18: 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 20-8.

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 22-15.

Use the LOADDATA command in ESSCMD to load data free form. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, “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 20-19, 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 20-19: Multiple Ranges of Member Names

In Figure 20-19, Sales is defined for the first two columns and Profit for the last two.

Ranges Set Automatically

When Hyperion 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 Hyperion Essbase encounters another member name from the same dimension, at which point Hyperion Essbase replaces the range with the new member or new member range.

Figure 20-20, for example, contains a range of Jan to Feb in the Year dimension. It remains in effect until Hyperion Essbase encounters another member name, such as Mar. If Hyperion 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 20-20: Ranges of Member Names

Data Values Out of Range

When Hyperion 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. Hyperion Essbase loads any data fields read before the invalid field into the database, resulting in a partial load of the data.

Figure 20-21, for example, contains more data fields than the defined range of members. The data load stops when it reaches the `10` data field. Hyperion 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 20-21: Extra Data Values

For information on restarting the load, see “Loading the Error Log File” on page 24-5.

Duplicate Members in a Range

If the same member appears more than once in a range, Hyperion Essbase ignores the duplicate members.

The file in Figure 20-22 contains duplicate members.

```
Cola East
        Actual    Budget    Actual    Budget
        Sales     Sales     COGS      COGS
Jan     108       110       49        50
Feb     102       120       57        60
```

Figure 20-22: Duplicate Members in a Range

Hyperion Essbase ignores the duplicate members. The members that Hyperion 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 20-23: Ignored Duplicate Members

Because Hyperion 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 20-24: How Hyperion Essbase Interprets the File in Figure 20-22

How Hyperion Essbase Reads Multiple Ranges

As Hyperion Essbase scans a file, it processes the most recently encountered range first when identifying a range of data values. In Figure 20-24, 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, Hyperion Essbase encounters the Actual and Budget range first and the Sales and COGS range last. Because the Sales and COGS range is encountered last, Hyperion 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. Hyperion Essbase supports loading data from both types of columns.

Symmetric Columns

Symmetric columns have the same number of members under them. In Figure 20-25, for example, each dimension column has one column of members under it. For example, Product has 100-10 under it.

```
Product     Measures   Market    Year    Scenario    *data*
"100-10"    Sales      Texas     Jan     Actual      112
"100-10"    Sales      Ohio      Jan     Actual      145
```

Figure 20-25: 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 20-26: Groups of Symmetric Columns

Asymmetric Columns

Columns can also be asymmetric. In Figure 20-27, 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 20-27: 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 20-28, 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 20-28: Invalid Asymmetric Columns

This file in Figure 20-29 is valid because the Jan label is now over both Actual and Budget. It is clear to Hyperion 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 20-29: Valid Asymmetric Columns

Security and Multi-User Considerations

Hyperion 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, Hyperion Essbase protects your data using the security system described in Chapter 17, “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. Hyperion Essbase uses a block locking scheme for handling multi-user issues. When you load data, Hyperion 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 Hyperion Essbase waits for a locked block to be released, and other transaction handling parameters. See Chapter 42, “Ensuring Data Integrity,” for more information.

    For information on how to see which user has a lock on a particular block, see Chapter 17, “Managing Security at Global and User Levels.”
  - Updates the block. Hyperion 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 42, “Ensuring Data Integrity.” for more information.

Chapter 21

Setting up a Rules File to Manipulate Records

This chapter describe how to create a data load or dimension build rules file that performs operations on records using Hyperion Essbase Application Manager. For information about performing operations on fields within a record, see Chapter 22, "Manipulating Fields Using a Rules File." For information about loading data using rules files, including prerequisites, see Chapter 23, "Performing a Data Load."

This chapter contains the following sections:

Selecting the Data Source

This section describes how to select your data source, including:

Connecting to the Server

Before you can create data load rules for a data source, you may want to connect to the server. You are not required to connect to the server before defining data load rules, because you can create the rules file on your client machine.

Viewing the Application and Database

Before you create data load rules for a data source, you may want to view the application and database. You are not required to view the application or database before defining data load rules. To view rules files, you must open Data Prep Editor which is available through the Application Desktop window. The Application Desktop window appears after you connect to the server.

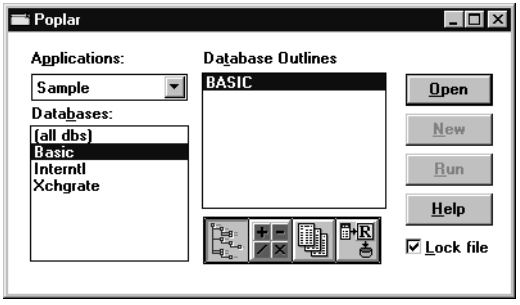


Figure 21-1: Application Desktop Window

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.

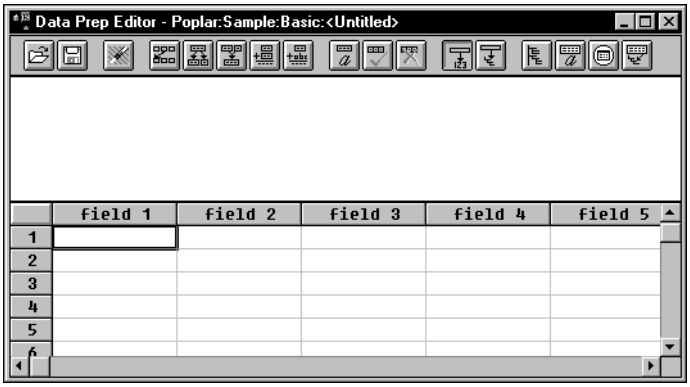


Figure 21-2: Data Prep Editor

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.

- Select View > Data Load or click the Data Load button,  , to view data load fields. Fields set to be ignored during the data load turn gray.
- Select View > Dimension Building Fields or click the Dimension Build button,  , to view dimension build fields. Fields set to be ignored during the dimension build turn gray. See Chapter 13, “Introducing Dynamic Dimension Building,” for more information on dimension building.

Customizing the Data Prep Editor

You can customize your view of the Data Prep Editor:

- To hide the gridlines, select View > Gridlines. The check mark disappears next to the Gridlines command in the View menu and the gridlines disappear from the Data Prep Editor. To show the gridlines, select View > Gridlines.
- To hide the raw data, select View > Raw Data. The check mark disappears next to the Raw Data command in the View menu and the raw data window disappears from the Data Prep Editor. To show the raw data, select View > Raw Data.
- To hide the toolbar, select View > Toolbar. The check mark disappears next to the Toolbar command in the View menu and the toolbar disappears from the Data Prep Editor. To show the toolbar, select View > Toolbar.

Opening a Data Source

After you open the Data Prep Editor, you can open data sources, such as text files, spreadsheet files, and SQL data sources. This section describes how to open the following kinds of data sources:

- “Opening a Text File” on page 21-5
- “Opening a Spreadsheet File” on page 21-7
- “Opening an SQL Data Source” on page 21-8

Opening a Text File

After you open the Data Prep Editor, you can open text files in it.

➤ To open a text file:

1. Select File > Open Data File to view a list of text files. The Open Server Data File Object dialog box contains values for the last data source you opened for this rules file. In this example, the last file opened was the Act1 file in the Sample Basic database.

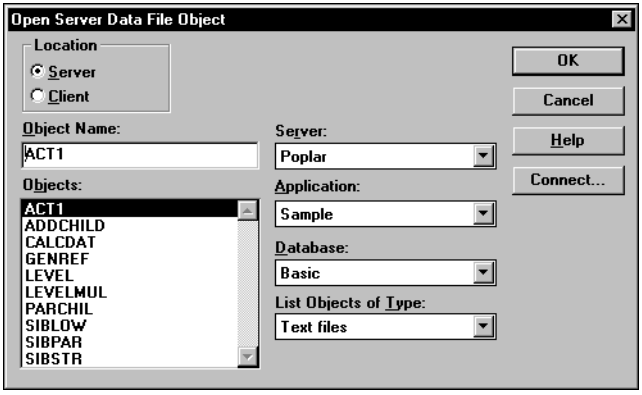


Figure 21-3: Open Server Data File Object Dialog Box

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. Select File > Open Data File to open the Open Server Data File Object dialog box.

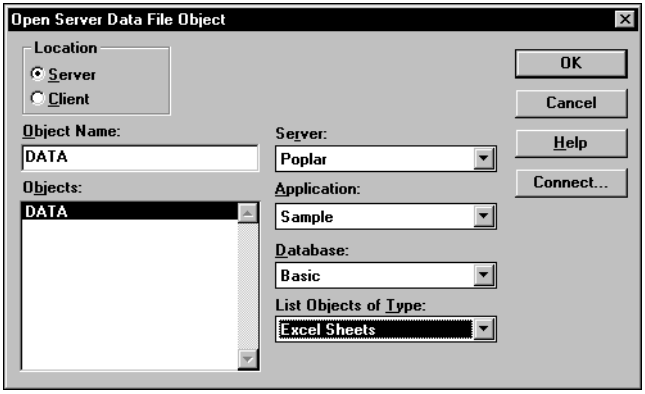


Figure 21-4: Open Server Data File Object Dialog Box: Spreadsheet Files

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 *Hyperion Essbase SQL Interface Guide* provides information on supported environments, installation, and connection to supported data sources. Contact your Hyperion 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 Hyperion Essbase dimensions, you don't have to perform any field mapping.

➤ To open an SQL data source:

1. Select File > Open SQL to open the Select Server, Application and Database dialog box. If you are connected to a server, the dialog box contains the values for that server.

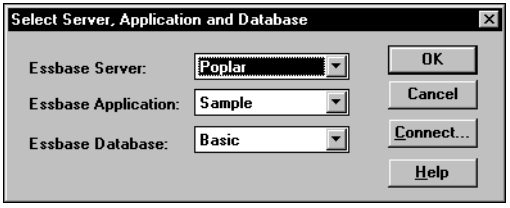


Figure 21-5: Select Server, Application and Database Dialog Box

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:

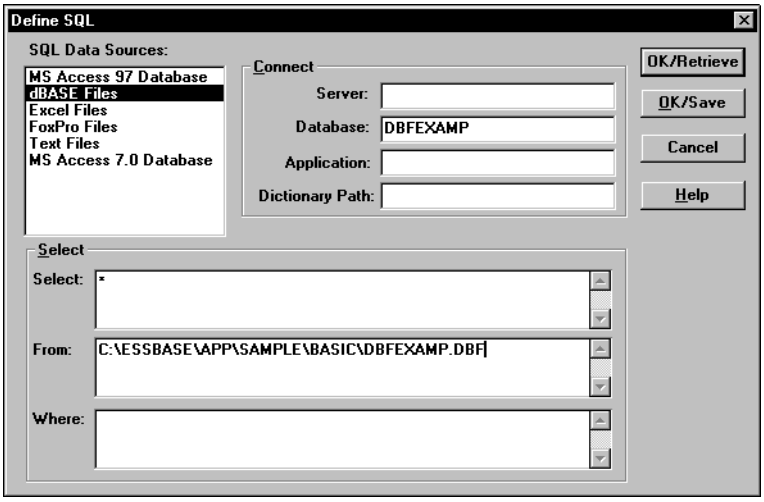


Figure 21-6: Define SQL Dialog Box

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. Select Options > Data File Properties or click the Data File Properties button,  , to open the Data File Properties dialog box. Click the File Delimiter tab.

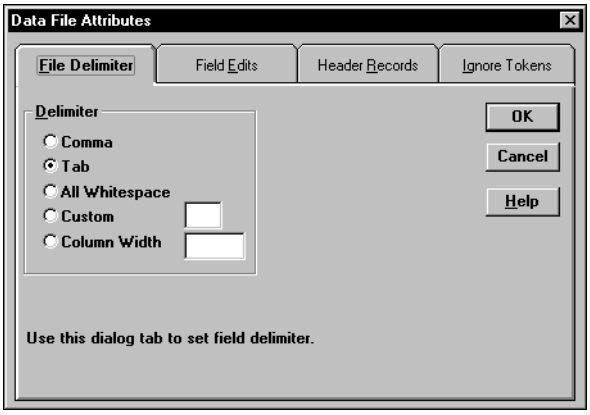


Figure 21-7: File Delimiter Page

2. Select the type of delimiter to use in your file:
   - Comma
   - Tab
   - All Whitespace

- Custom—in the text box, enter a single character as the custom delimiter.
- Column Width—in the text box, enter the width of the column. The column width must be a five digit or smaller number. Use column width for data sources with fixed-width columns.

3. Click OK.

Using Header Information

Data sources can contain data records and header records. *Data records* contain data: member fields and data fields. *Header records* describe the contents of the data source and how to load data values in the data source to the database.

Rules files contain records that translate the data in the data source to map it to the database. As part of that information, rules files can also contain header records.

You can create header records using the following methods:

- In the rules file using the Data Prep Editor.
- In the data source using a text editor or spreadsheet; then set them as header records in the rules file.

Defining Header Information in the Rules File

You can define header information in the rules file. These headers are only used during data loading or dimension building and do not change the data source. Header information in a rules file is not used if there is also a dynamic reference in the rules file pointing to a header record in the data source.

1. Select Options > Data Load Settings or click the Global Data Load Properties button,  , to open the Data Load Settings dialog box. Click the Header Definition tab.

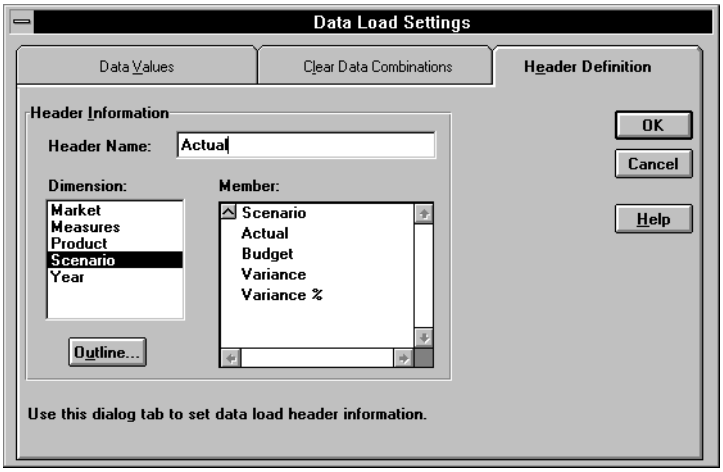


Figure 21-8: Header Definition Page

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 dynamic references. *Dynamic references* are set in the rules file and tell Hyperion Essbase to read the header information as a header record and not a data record. When setting dynamic references, you can also specify which type of header information is in which header record.

Header information defined in the data source takes precedence over header information defined in the rules file.

➤ To define header information in the data source:

1. Add the header information to the data source using a text editor or spreadsheet. Member combinations must be separated by a comma.

2. Select Options > Data File Properties or click the Data File Properties button,  , to open the Data File Properties dialog box. Click the Header Records tab.

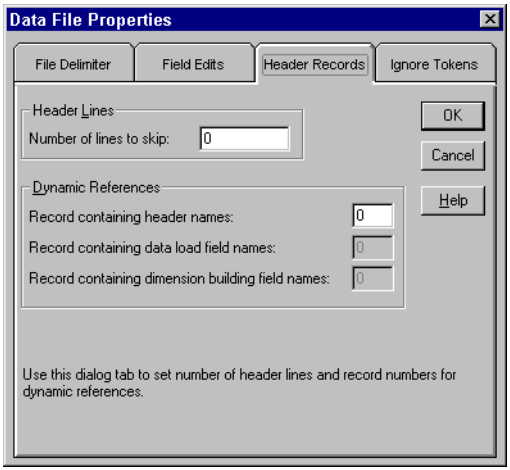


Figure 21-9: Header Records Page

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 Hyperion Essbase does not process the records in those lines as data records. Hyperion Essbase still processes those records as header records. You should tell Hyperion 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 13, "Introducing Dynamic Dimension Building," for more information.
7. Click OK.

Selecting Records

You can specify which records Hyperion 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 Hyperion Essbase loads the record. If a field or fields in the record does not meet the selection criteria, Hyperion Essbase doesn't load the record. For example, to load only the Budget data from a data source, you could create a selection criterion to load only records where the first field was Budget.

➤ To define the selection criteria:

1. Select the field to which to apply the criteria. For example, field 1.
2. Select Record > Select or click the Record Selection button,  , in the Data Prep Editor toolbar to open the Select Record dialog box.

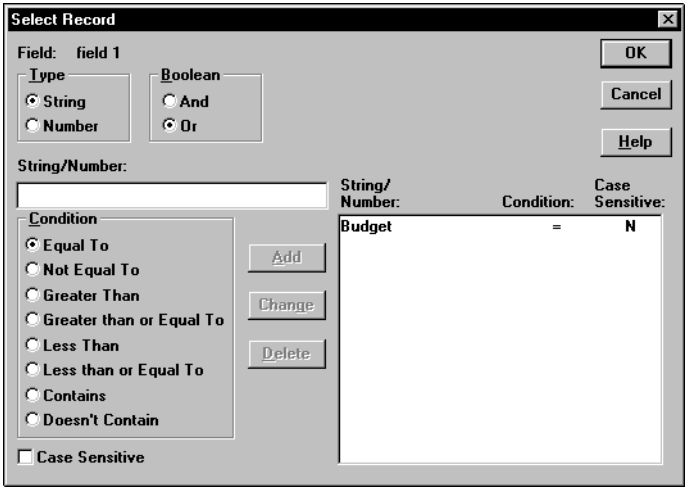


Figure 21-10: Select Record Dialog Box

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 selecting Case Sensitive.
4. Enter the string or number upon which the criterion is based in the String/Number text box; for example, Budget.
5. Select how to evaluate the field by clicking one of the condition options in the Condition box; for example, Equal To.
6. Add the selection criterion to the list by clicking Add. To create multiple selection criteria for a single field, repeat this process for each selection criterion. To change or delete a criterion from the list, select the criterion to change or delete and click Change or Delete.
7. If you define multiple selection criteria on a single field, you can specify how Hyperion 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 Hyperion Essbase combines the criteria. See "Defining Multiple Select and Reject Criteria" on page 21-18.

8. Click OK.

Rejecting Records

You can specify which records Hyperion 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 Hyperion Essbase to reject the record. If a field in the record does not meet the rejection criteria, Hyperion Essbase loads the record. For example, to reject the Actual data from a data source and load only the Budget data, you could set a rejection criterion to reject records where the first field was Actual.

➤ To define rejection criteria:

1. Select the field to which to apply the criterion.

2. Select Record > Reject or click the Record Rejection button,  , to open the Reject Record dialog box.

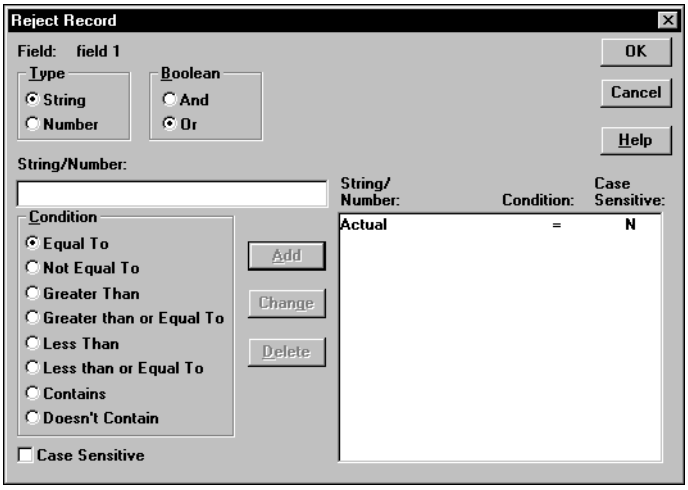


Figure 21-11: Reject Record Dialog Box

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 selecting the Case Sensitive box.
4. Enter the string or number upon which the criterion is based in the String/Number text box; for example, Actual.
5. Select how to evaluate the field by clicking one of the condition options in the Condition box; for example, Equal To.
6. To create multiple rejection criteria for a single field, add the first one by clicking Add and repeat this process for each rejection criterion. To change or delete a criterion from the list, select the criterion to change or delete and click Change or Delete.

7. If you define multiple rejection criteria on a single field, you can specify how Hyperion 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 Hyperion Essbase should combine the criteria. See "Defining Multiple Select and Reject Criteria" on page 21-18.

8. Click OK.

Defining Multiple Select and Reject Criteria

When you define select and reject criteria on multiple fields, you can specify how Hyperion Essbase combines the rules across fields, that is, whether the criteria are connected logically with AND or OR. If you select And from the Boolean group, the fields must match all the selection or rejection criteria. If you select Or from the Boolean group, the fields must only match one of the selection or rejection criteria. Setting the global Boolean applies it to all select or reject operations in the rules file, for both data load and dimension-building fields.

Note: If your selection and rejection criteria apply to the same record (that is, you try to select and reject the same record), the record is rejected.

For example, to load only the New York, Actual data in a data source containing data for other markets and scenarios, create select criteria to select records that contain New York and Actual in the specified fields.

1. Select Options > Data Load Settings or click the Global Data Load Properties button,  , to open the Data Load Settings dialog box. Click the Data Values tab.

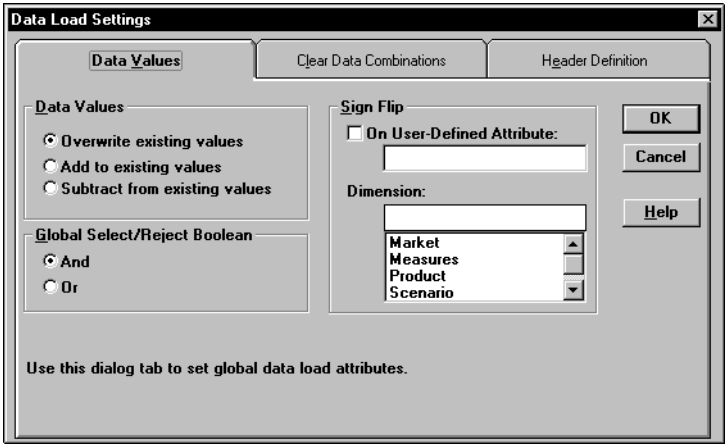


Figure 21-12: Data Values Page: Global Select/Reject Boolean

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 Hyperion Essbase displays in the Data Prep Editor and the first record to display:

1. Select Record > Record View Count to open the Record View Count dialog box.

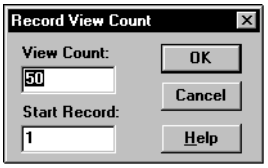


Figure 21-13: Record View Count Dialog Box

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. Hyperion 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, Hyperion Essbase does not display records 1 through 4.

   Note: Hyperion 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 21-14, for example, displays records 5 through 8.

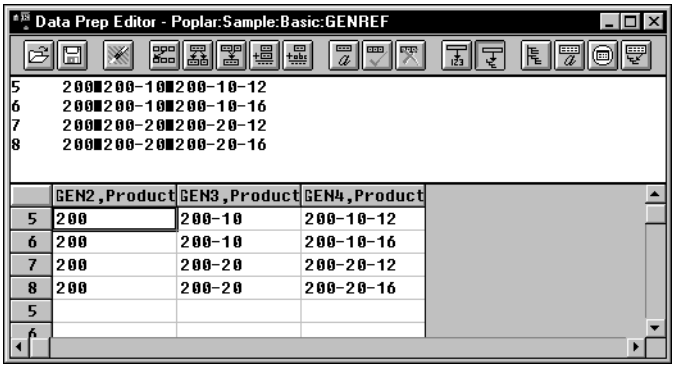


Figure 21-14: Displaying Record 5 through 8

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 21-22.

Rules files are validated to make sure the member and dimension mapping defined in the rules file maps to the outline. See Chapter 22, “Manipulating Fields Using a Rules File,” to learn how to map fields to members. Validation cannot ensure that the data source will load properly.

Associating a Rules File with an Outline

➤ To associate a rules file with an outline:

1. Select Options > Associate Outline or click the Outline button,  , to open the Associate Server Outline Object dialog box.

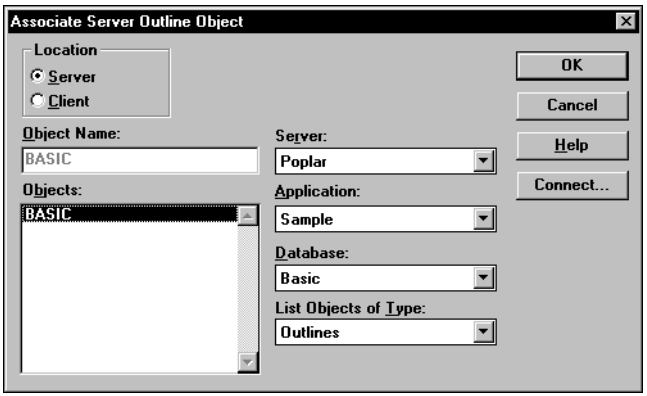


Figure 21-15: Associate Server Outline Object Dialog Box

2. Make sure the values for Object Name, Server, Application, and Database are correct. By default, Hyperion 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. Select Options > Validate or click the Validate Rules button,  , to validate the rules file against the outline. When Hyperion Essbase finishes the validation, the Validate Rules dialog box appears. It contains information about the validation process, including which fields did not map to the outline.

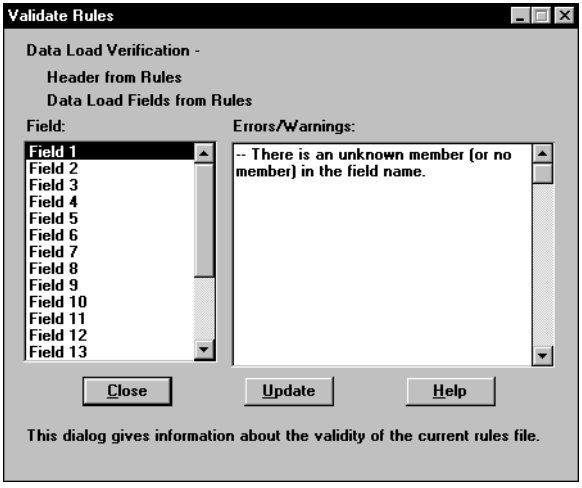


Figure 21-16: Validate Rules Dialog Box

2. If the rules file validates with no problems, you can use it to load data. See Chapter 20, "Introducing Data Loading," for information on loading data.
3. If the rules file is not correct, you must fix it. Go to the field specified in the Validate Rules dialog that did not pass validation. In Figure 21-16, 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 24, "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. Select File > Save or click the Save button,  . If you haven't saved the rules file before, the Save Server Object dialog box appears.

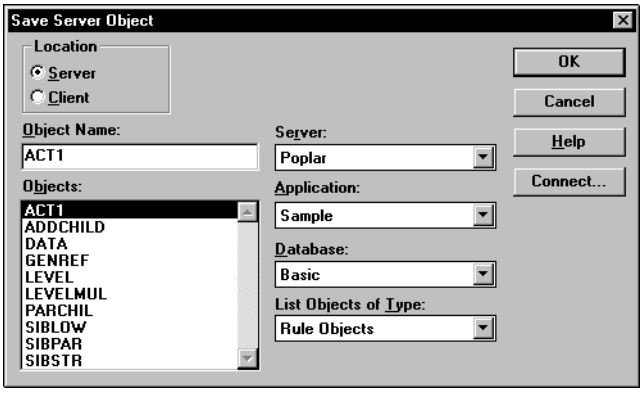


Figure 21-17: Save Server Object Dialog Box

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, the rules file name you specify must be eight or fewer alphanumeric characters. Hyperion Essbase automatically adds an extension of `.RUL`. For example, you can name the rules file `ACT1.RUL` by entering `ACT1` in the Object Name text box.
4. Click OK.

Saving Under a Different Name

To save a rules file under a different name, select File > Save As to open the Save Server Object dialog box. Follow the same steps as for saving a rules file, but enter a different name.

Chapter 22

Manipulating Fields Using a Rules File

This chapter describes how to manipulate fields during a data load or dimension build using a rules file using 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 21, "Setting up a Rules File to Manipulate Records."

For more information about loading data using rules files, including prerequisites, see Chapter 23, "Performing a Data Load."

This chapter contains the following sections:

Selecting Multiple Fields

You can select multiple fields and then set the properties for them. Hyperion Essbase grays out any menu items and controls you cannot use when more than one field is selected.

➤ To select continuous fields:

1. Click the first field.
2. Shift-click the last field.

To select discontinuous fields, use one of the following methods:

- Select the fields in the first region, hold down Ctrl and drag the mouse along the fields in the second region.
- Click the first field, then hold down Ctrl and click the other fields.

Ignoring Fields

You can ignore all the fields in a column in your data source that do not map to the database. The fields still exist in the data source, but they are not loaded into the Hyperion Essbase database. For example, you could have a column containing comment fields and you could choose to ignore the fields in that column for each record in the data source.

You can also ignore individual fields in your data source that match a string called a *token*. When you ignore fields based on string values, these fields are ignored everywhere they appear in the data source, not just in a single column.

Ignoring All Fields in a Column

You can ignore an entire column, that is, ignore the field in a specified column for each record.

➤ To ignore all fields in a column:

1. Select the columns to ignore.
2. Choose Field > Properties or click the Field Properties button,  , to open the Field Properties dialog box. Click the Global Properties tab.

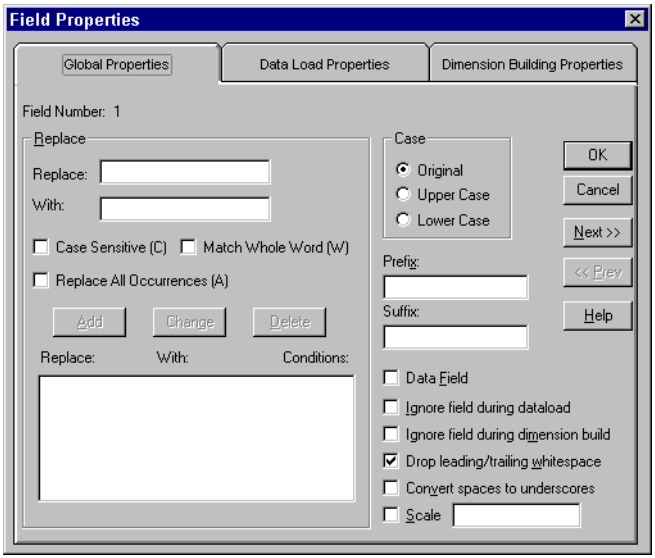


Figure 22-1: Global Properties Page: Ignore Check Boxes

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 Properties to bring up the Data File Properties dialog box. Click the Ignore Tokens tab. Token is another word for string.

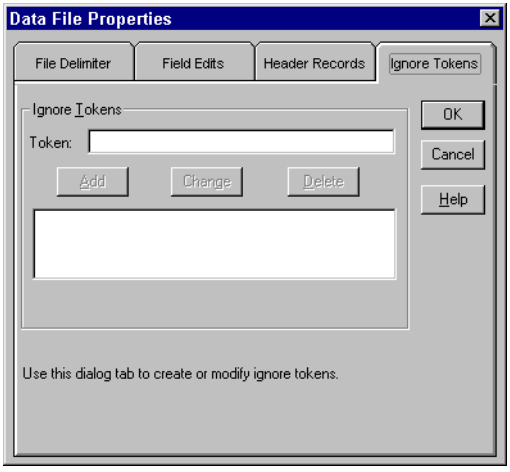


Figure 22-2: Ignore Tokens Page

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:

Note: Whenever you want to undo a single operation, choose Edit > Undo. To undo multiple field operations, see "Undoing Field Ordering" on page 22-11.

Moving Fields

➤ To move one or more fields:

1. Select the field to move. If you do not select a field, the currently active field is selected by default.
2. Choose Field > Move or click the Move button,  , to open the Move Field dialog box.

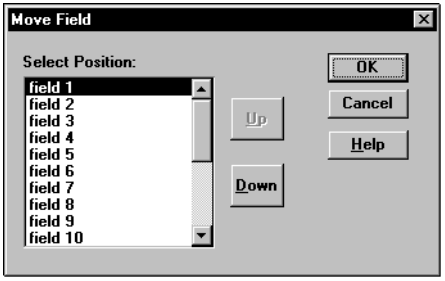


Figure 22-3: Move Field Dialog Box

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 22-11.

Joining Fields

You can join multiple fields into one field. For example, if you receive a data source with separate fields for the product number (100) and product family (-10), you must join those fields (100-10) to load them into the Sample Basic database.

1. Move the fields to join into the order in which you want to join them. If you do not know how to move fields, see "Moving Fields" on page 22-5.
2. Select the fields to join, for example 100 and -10. If you do not know how to select multiple fields, see "Selecting Multiple Fields" on page 22-2.
3. Choose Field > Join or click the Join button,  , to open the Join Fields dialog box.

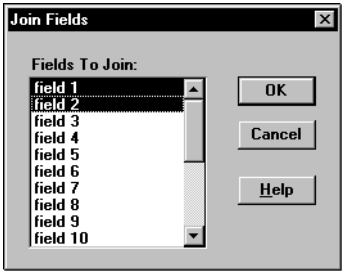


Figure 22-4: Join Fields Dialog Box

4. If you did not select the fields to join before opening the Join Fields dialog box, select the fields in the Fields to Join list box. The order in which fields are joined is determined by the order in which they appear in the list box.
5. Click OK. The selected fields merge into one. The new field is named after the first field in the join. The original fields are part of the joined field.

Note: To undo a join, see "Undoing Field Ordering" on page 22-11.

Creating a New Field While Leaving Existing Fields Intact

You can create a copy of a field or you can join two or more fields by putting the joined fields into a brand new field. This leaves the existing fields intact.

Creating a New Field By Joining Fields

You may need to concatenate fields from your source data to create the member you want to define in your rules file.

For example, if you receive a data source with separate fields for the product number (100) and product family (-10), you must join those fields (100-10) to load them into the Sample Basic database. But suppose that you want to leave the 100 and -10 fields in the data source after the join, that is, the data source would contain three fields: 100, -10, and 100-10. To do this, create the new field using a join.

1. Select the fields to join, for example, 100 and -10. If you do not know how to select multiple fields, see “Selecting Multiple Fields” on page 22-2.

2. Choose Field > Create Using Join or click the Create Using Join button,  , to open the Create Field Using Join dialog box.

Figure 22-5: Create Field Using Join Dialog Box

3. If you did not select the fields to join, select the fields in the Create Field Using Join dialog box.

4. Click OK.

   The new field displays to the left of the first field containing joined information. For example, in Figure 22-6, a new 100-10 field displays to the left of the existing 100 and -10 fields.

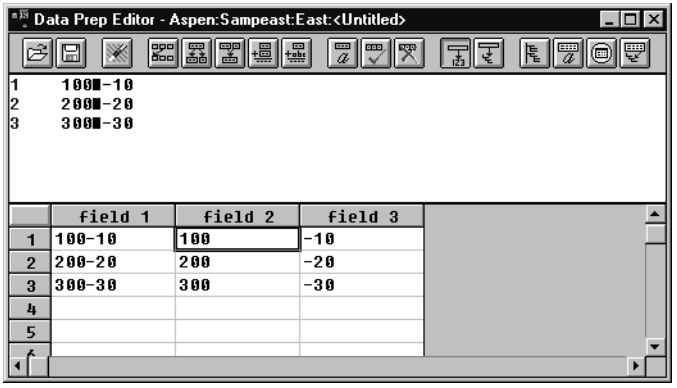


Figure 22-6: New 100-10 Field

Copying Fields

You may need to create a new field in the rules file that is a copy of an existing one; for example, when you define a multilevel attribute dimension and associate attributes to members of a base dimension during the same dimension build.

1. Select the field to copy, for example, 12.
2. Choose Field > Create Using Join or click the Create Using Join button,  , to open the Create Field Using Join dialog box.
3. Make sure the only field selected in the Create Field Using Join dialog box is the field you want to duplicate.
4. Click OK.

   The new field displays to the left of the field you selected to copy. You may need to move the field to another location. See “Moving Fields” on page 22-5.

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 22-3. 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.

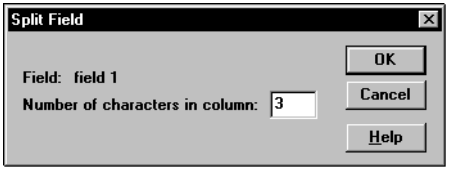


Figure 22-7: Split Field Dialog Box

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 displays to the left of the original field.

Note: To undo a split, see “Undoing Field Ordering” on page 22-11.

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.

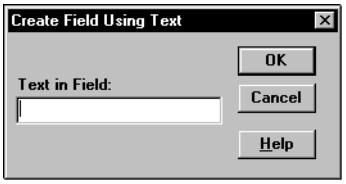


Figure 22-8: Create Field Using Text Dialog Box

3. Enter the text to put into the new field, for example, a hyphen (-).
4. Click OK. The new field displays to the left of the selected field.

Note: To undo a field you created using text, see “Undoing Field Ordering” on page 22-11.

Undoing Field Ordering

You can undo the last field operation you performed such as move, join, split, or create using text, using the Edit > Undo command. You can also undo field operations even if you have performed other actions. Undoing field operations is sequential; you must undo them from the last operation to the first.

1. Select the field or fields. If you do not know how to select multiple fields, see "Selecting Multiple Fields" on page 22-2.
2. Choose Options > Data File Properties or click the Data File Properties button,  , to open the Data File Properties dialog box. Click the Field Edits tab.

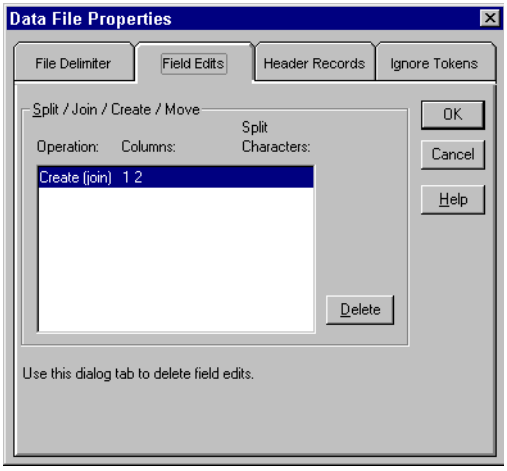


Figure 22-9: Field Edits Page

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 22-12.
- Replace strings, or translate, the fields in the data source to match members in the database. See “Replacing Text Strings” on page 22-14.

Naming Fields

Use a rules file to name data source fields to match Hyperion 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 Hyperion Essbase dimensions, you do not have to perform any field mapping.

➤ To name a field:

1. Select the field to name in the Data Prep Editor.

2. Choose Field > Properties or click the Field Properties button,  , to open the Field Properties dialog box. Click the Data Load Properties tab.

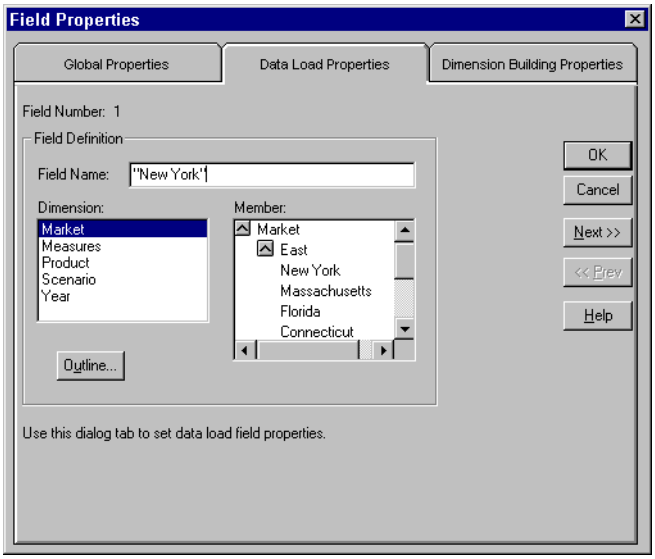


Figure 22-10: Data Load Properties Page

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 quotation marks around the member name. If you click the member name in the Member list box, Hyperion Essbase automatically puts quotation marks around member names with spaces in them.

4. To name the next field in the Data Prep Editor, click the Next button. To change the previous field, click the Prev button. This saves all changes before going to the next field.

5. When you are finished, click OK.

Replacing Text Strings

Use a rules file to replace text strings so that the fields map to Hyperion Essbase member names during a data load. The data source is not changed. For example, if the data source abbreviates New York to NY, you could have the rules file replace each NY with New York while loading the data.

1. Select the field or fields containing the text string you want to change.

2. Choose Field > Properties or click the Field Properties button,  , to open the Field Properties dialog box. Click the Global Properties tab.

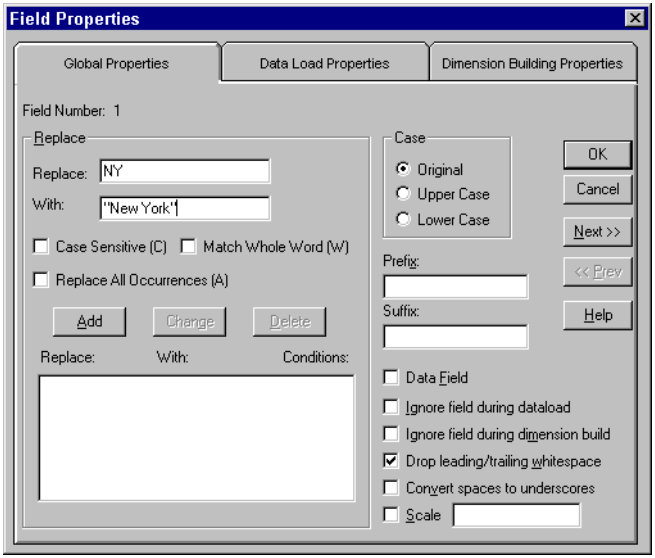


Figure 22-11: Global Properties Page: Replace Box

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 22-15.

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, Hyperion Essbase only changes the first occurrence.
6. Click Add to add the replacement operation to the list. To change an existing operation in the list, select the operation and click Change. To delete an operation from the list, select the operation and click Delete.
7. To move to the next column, click the Next button. To move to the previous column, click the Prev button.

   Note: The Next and Prev buttons only work if a single field is selected.
8. Click OK.

Replacing an Empty Field with Text

You may want to replace empty fields in a column with text. If, for example, empty fields in the column represent default values, you could insert the default values to replace the empty ones or insert `#MI` to represent missing values.

Replacing an empty field with text requires several steps. To replace an empty field with text:

1. Select the column containing the empty fields you want to replace.
2. Choose Field > Create Using Text or click the Create Using Text button,  .
3. Enter the text to put in the new field. Enter a dummy string that is not in the selected column, such as “temp.” Click OK.
4. Join the new field with the column containing the fields to replace. Select both columns, choose Field > Join, and click OK. The previously blank fields now contain the dummy string you entered in step 3; for example, temp.

5. Select the column containing the dummy string and choose Field > Properties or click the Field Properties button,  . Click the Global Properties tab.
6. Enter the text string you want to replace in the Replace text box. This should be the dummy string you entered in step 3; for example, temp. Now enter the text to replace it with in the With text box. This should be the final value you want to put in the fields, for example, default. Select the Match Whole Word.
7. Click Add and then click OK.
8. Now replace the extra dummy strings in the column with nothing. Choose Field > Properties or click the Field Properties button,  . Click the Global Properties tab. Enter the dummy string in the Replace text box; for example, temp. Enter nothing in the With text box. Make sure the Match Whole Word option is not checked. Click Add and then click OK.

Changing the Case of Fields

Use a rules file to change the case of a field so the field maps to Hyperion Essbase member names during a data load. The data source is not changed. For example, if the data source capitalizes a field that is in lower case in the database, you could change the field to lower case; for example, from JAN to jan.

1. Select the field or fields containing the text string you want to change.

2. Choose Field > Properties or click the Field Properties button,  , to open the Field Properties dialog box. Click the Global Properties tab.

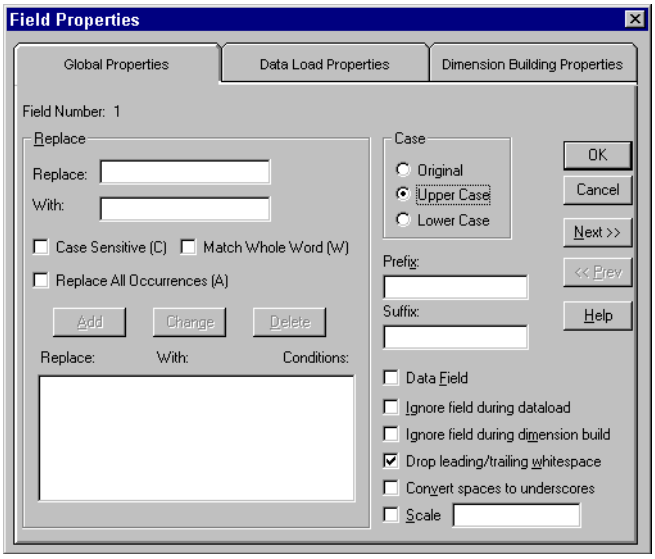


Figure 22-12: Global Properties Page: Case Box

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, Hyperion Essbase drops leading and trailing white space.

➤ To drop leading or trailing white space:

1. Select the field or fields. If you do not know how to select multiple fields, see "Selecting Multiple Fields" on page 22-2.

2. Choose Field > Properties or click the Field Properties button,  , to open the Field Properties dialog box. Click the Global Properties tab.

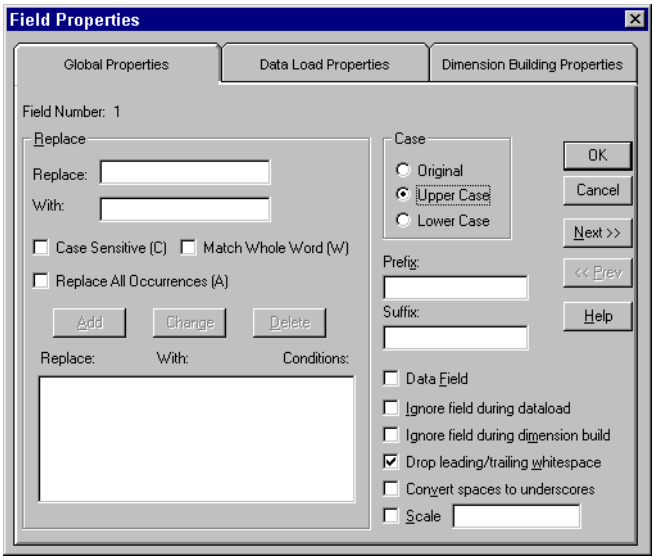


Figure 22-13: Global Properties Page: Drop Leading/Trailing Whitespace Check Box

3. By default, the "Drop leading/trailing whitespace" box is checked. If it is not already checked, check it.

4. Click OK.

Converting Spaces to Underscores

You can convert spaces in fields in the data source to underscores to make them match member names in the database.

➤ To convert spaces to underscores:

1. Select the field or fields. If you do not know how to select multiple fields, see “Selecting Multiple Fields” on page 22-2.

2. Choose Field > Properties or click the Field Properties button,  , to open the Field Properties dialog box. Click the Global Properties tab.

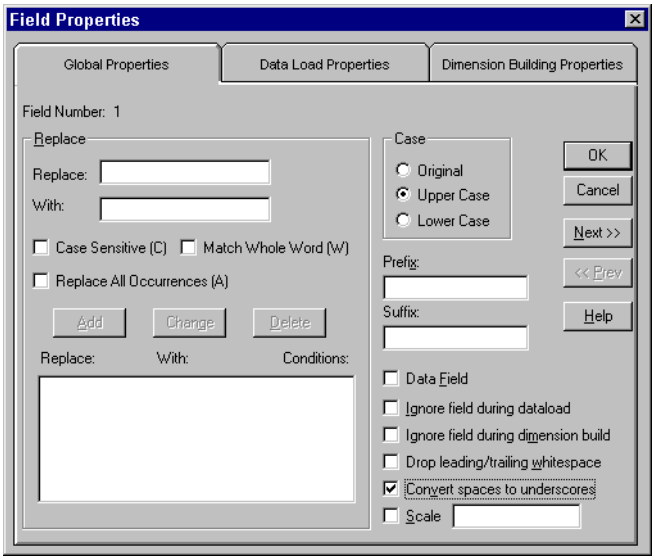


Figure 22-14: Global Properties Page: Convert Spaces to Underscores Check Box

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 Hyperion Essbase member names during the data load.

1. Select the field or fields. If you do not know how to select multiple fields, see "Selecting Multiple Fields" on page 22-2.

2. Choose Field > Properties or click the Field Properties button,  , to open the Field Properties dialog box. Click the Global Properties tab.

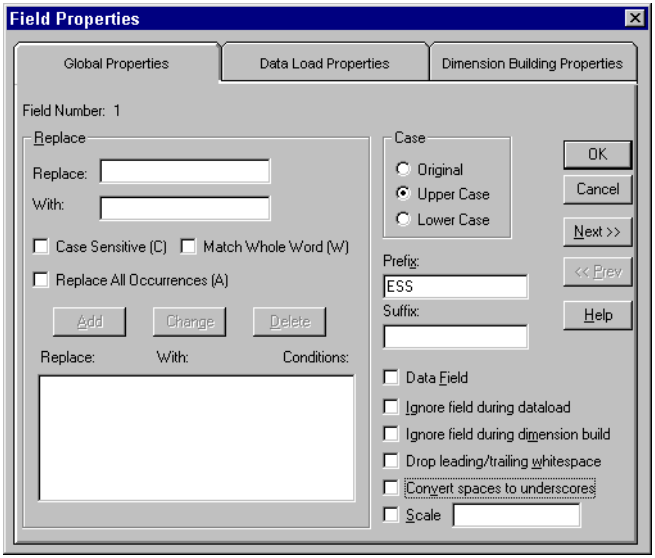


Figure 22-15: Global Properties Page: Prefix and Suffix Text Boxes

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 > Properties or click the Field Properties button,  , to open the Field Properties dialog box. Click the Global Properties tab.

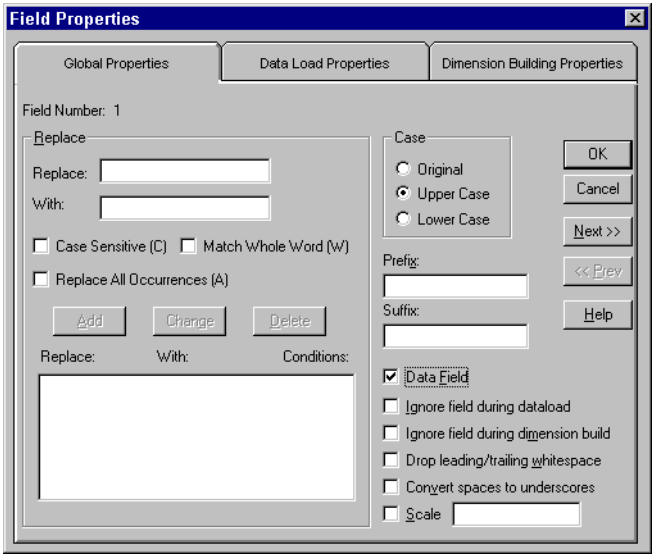


Figure 22-16: Global Properties Page: Data Field Check Box

3. Check Data Field.
4. Click OK.

Changing Data Values

By default, Hyperion Essbase overwrites the existing values in the database, but the following sections describe:

- “Adding to and Subtracting from Existing Values” on page 22-22
- “Clearing Existing Data Values” on page 22-23
- “Scaling Data Values” on page 22-25

Adding to and Subtracting from Existing Values

You can add or subtract the values in incoming records to existing values in an Hyperion Essbase database. For example, if you load weekly values, you can add them to create monthly values in the database.

➤ To add or subtract existing values:

1. Select the field to add to or subtract from.
2. Choose Options > Data Load Settings or click the Global Data Load Properties button,  , to open the Data Load Settings dialog box. Click the Data Values tab.

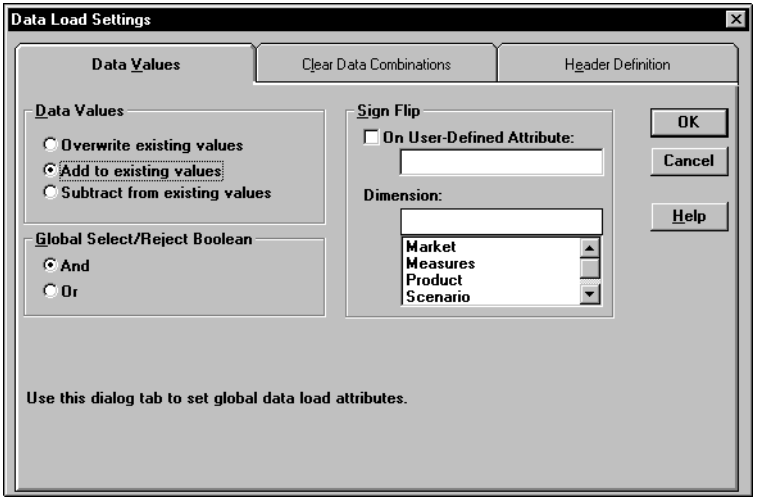


Figure 22-17: Data Values Page: Data Values Box

3. Choose “Add to existing values” to add the values or the “Subtract from existing values” to subtract the values.

CAUTION: Using this option makes it more difficult to recover if the database crashes while loading data, although Hyperion Essbase lists the number of the last row committed in the application event log file. For more information, see “Application Event Log File” on page 46-28.

To solve this problem, as a Database Transaction setting, set the Commit Row value as 0. This causes Hyperion Essbase to view the entire load as a single transaction and commit the data only when the load is complete. For more information, see “Specifying Isolation Level” on page 41-15.

4. Click OK.

Clearing Existing Data Values

You can clear existing data from the database before loading new values. By default, Hyperion 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, Hyperion Essbase adds or subtracts the new values with the existing ones.

Before adding or subtracting new values, you need to make sure the existing values are correct. If you are loading the first set of values into the database, you must make sure there is no existing value.

For example, let us assume that the Sales figures for January are calculated by adding together the values for each week in January. That means:

```
January 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, Hyperion 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 are not part of the data load. For example, if a data source contained data for January, February and March and you only wanted to load the March data, you could clear the January and February data.

Note: If you are using transparent partitions, you can clear the values using just the same steps as for clearing data in a local database.

➤ To clear existing values:

1. Choose Options > Data Load Settings or click the Global Data Load Properties button,  , to bring up the Data Load Settings dialog box. Click the Clear Data Combinations tab.

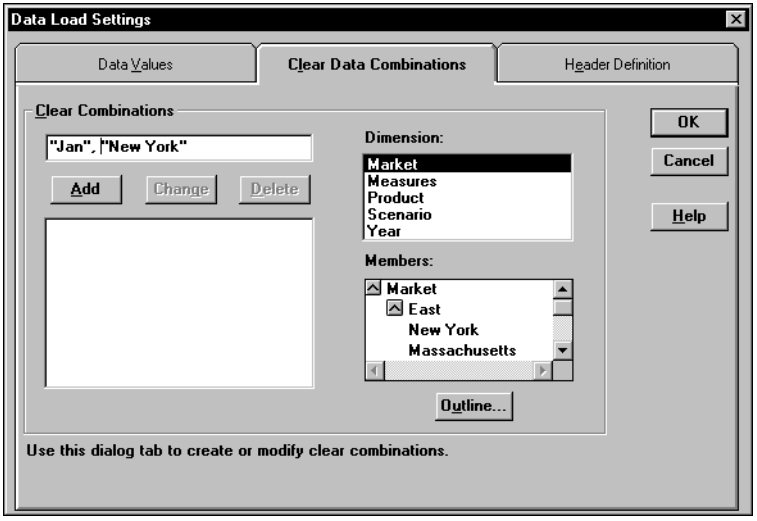


Figure 22-18: Clear Data Combinations Page

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 21-22 for more information on associating a rules file with an outline.

   You can enter Hyperion 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 Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

   Note: You must separate member combinations with a comma.

3. Click Add to add the member combination to the list. To change a member combination that is in the list, select the item in the list and click Change. It appears in the Clear Combinations text box. To delete member combinations from the list, select them and click Delete.

Scaling Data Values

You can scale data values if the values in the data source are not in the same scale as the values in the database. For example, the data source could track Sales in hundreds while the database tracks them in thousands. In this case, you would want to multiply the incoming values by 10.

1. Select the field to scale.

2. Choose Field > Properties or click the Field Properties button,  , to open the Field Properties dialog box. Click the Global Properties tab.

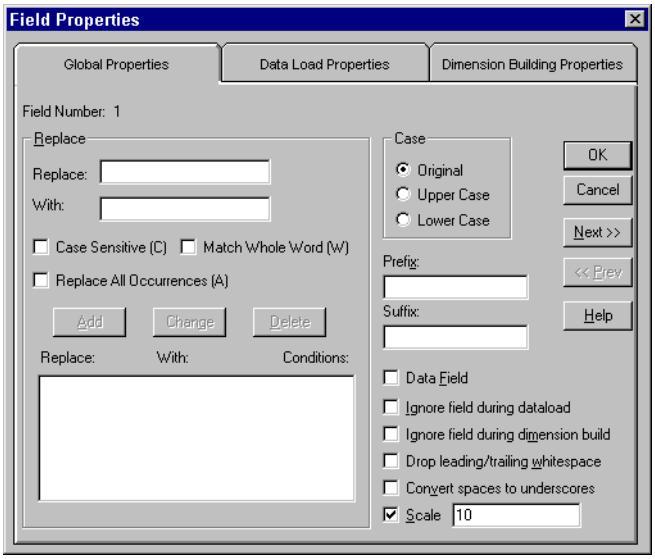


Figure 22-19: Global Properties Page: Scale Check Box

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 UDAs (user-defined attributes) in the outline. When loading data into the Accounts dimension, for example, you could specify that any record whose Accounts member had a UDA of Expense should change from a plus sign to a minus sign. You set UDAs in the Outline Editor. See Chapter 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 Properties button,  , to open the Data Load Settings dialog box. Click the Data Values tab.

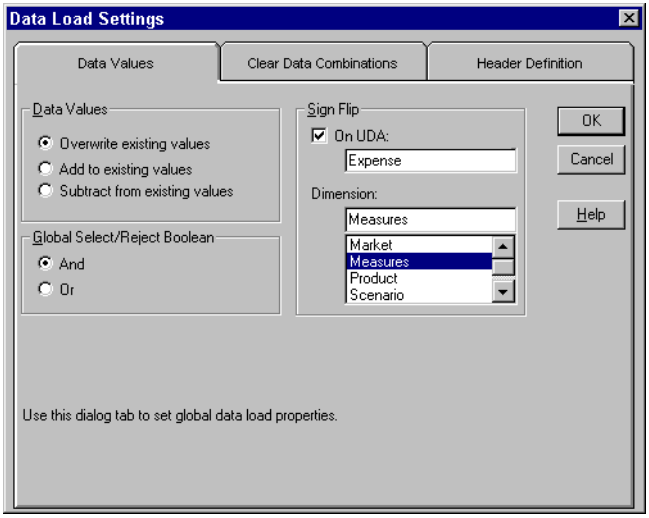


Figure 22-20: Data Values Page

2. Check On UDA under Sign Flip.
3. Enter the UDA, for example, Expense. Use the Outline Editor to get a list of UDAs. See Chapter 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 21-22 for more information.
5. Click OK.

Chapter 23

Performing a Data Load

This chapter describes how to load data from one or more external data sources to your Hyperion Essbase OLAP Server using the Hyperion Essbase Application Manager. It shows you how to use free-form or rules file data sources to load data or build dimensions dynamically.

You can load data without updating the outline, you can update the outline without loading data, or you can do both operations simultaneously.

This chapter contains the following sections:

Use the LOADDATA or UPDATEFILE commands in ESSCMD to load data without a rules file. See the online *Technical Reference* in the `DOCS` directory for information about these commands. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Prerequisites for Loading Data and Building Dimensions

To start loading data or building dimensions, you must have:

- A Hyperion Essbase database into which to load the data or build the outline.
- A connection to your server.
- Valid data sources:
  - Microsoft Excel files with the `.XLS` extension, Version 4.0 and higher. You must load Microsoft Excel files Version 5.0 and higher as client objects or files in the file system.
  - Lotus 1-2-3 files with the `.WKS`, `.WK1`, `.WK3`, or `.WK4` extension
  - Spreadsheet audit log files
  - ASCII text files (flat files) from ASCII backups or external sources
  - Hyperion Essbase export files. For information about reloading exported data, see "Reloading Exported Data" on page 48-4.
  - SQL data sources
- If you are not using a rules file for loading data, a data source correctly formatted for free-form data loading. See "Rules for Free-Form Data Sources" on page 20-18.
- If you are using a rules file for loading data, a rules file validated for data load. For information about defining data load rules files, see Chapter 20, "Introducing Data Loading."
- If you are performing a dimension build, a rules file validated for dimension build. For more information about defining dimension build rules, see Chapter 13, "Introducing Dynamic Dimension Building."

Note: You must use a rules file to load SQL data or to build dimensions and members dynamically.

Choosing the Data Sources Using the Hyperion Essbase Application Manager

You can select data sources using the Hyperion Essbase Application Manager or Windows. For a list of valid data sources, see “Prerequisites for Loading Data and Building Dimensions” on page 23-2.

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 the `DOCS` directory for information about this command. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

➤ To select a data source using Hyperion Essbase Application Manager:

1. Make sure you are connected to a server:
2. Click the Application Desktop window.

3. Select Database > Load Data to specify how to load data or build dimensions. The Data Load dialog box displays.

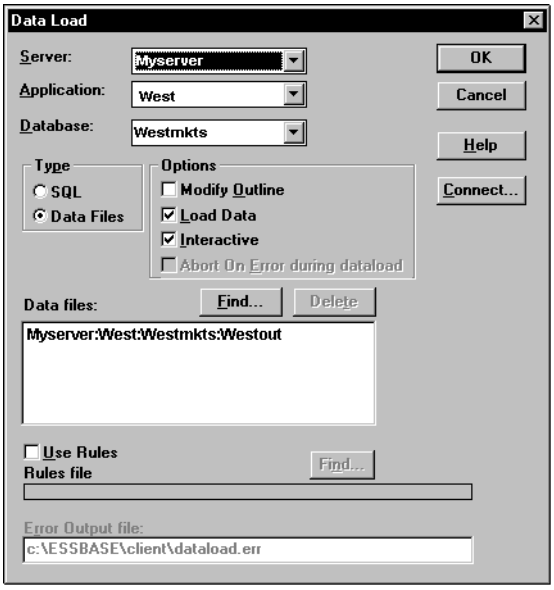


Figure 23-1: Data Load Dialog Box

4. Click Connect to open the Hyperion Essbase System Login dialog box. Enter the correct values and click OK. Now you are ready to start:
   - "Choosing SQL Data Sources" on page 23-4
   - "Choosing Text or Spreadsheet Files" on page 23-5

Choosing SQL Data Sources

➤ To load SQL data into an Hyperion Essbase database:

1. Click the Application Desktop window.
2. Select the application and database to load the data into or build dimensions for.
3. Select Database > Load Data. The Data Load dialog box displays.

4. To access SQL data, you must first connect to the SQL data source. Select the SQL option. The Data Load dialog box changes to look like this:

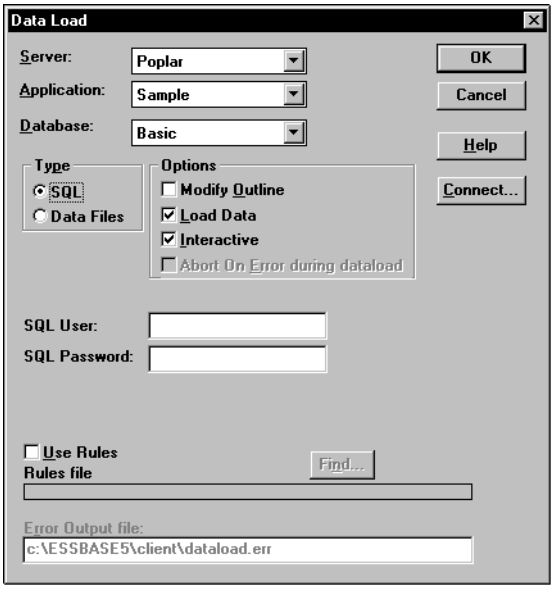


Figure 23-2: SQL Data Load Dialog Box

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 23-9 to finish, because you must use a rules file to load SQL data sources.

Choosing Text or Spreadsheet Files

➤ To select text or spreadsheet files:

1. Click the Application Desktop window.
2. Select the application and database to load the data into or build dimensions for.
3. Select Database > Load Data. The Data Load dialog box displays.
4. Click the Data Files option button if it is not already selected.

5. Click Find to select a text or spreadsheet file to load. The Open Server Data File Object dialog box displays:

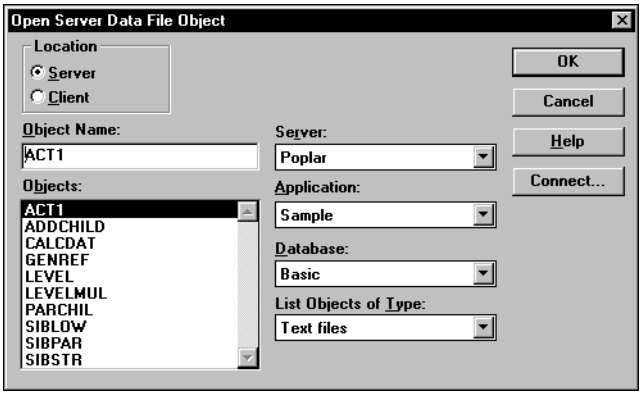


Figure 23-3: Open Server Data File Object Dialog Box

6. Make sure the appropriate Hyperion 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 23-3, for example, you could select ACT1.

   If you select Client, the file may reside in either the application or database directory under `\ESSBASE\CLIENT` or on the drives accessible from the client file system. Click File System to select a file from a standard Open Client Data Files dialog box. Select the file to open, for example, `ASYMM.XLS` in the `\ESSBASE\CLIENT\SAMPLE` directory.

To select multiple files, hold down the Ctrl key and click the files.

Note: ESSBASE is the default directory specified during installation. You may have specified a different default directory.

Load Microsoft Excel files Version 5.0 and higher as client objects or files in the file system, not as server objects.

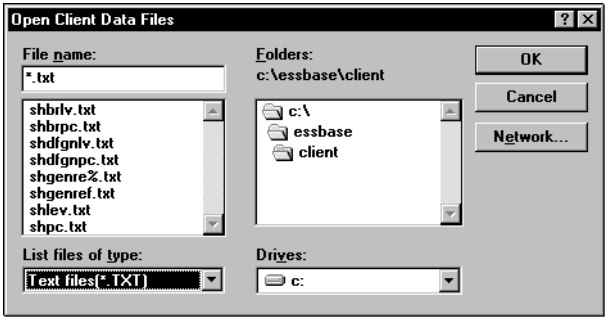


Figure 23-4: Open Client Data Files Dialog Box

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 select the data sources using Hyperion Essbase Application Manager, the Windows File Manager or the Windows Explorer. For a list of valid data sources, see "Prerequisites for Loading Data and Building Dimensions" on page 23-2.

Make sure you are connected to the server before you select the data sources.

➤ To select a list of files:

1. Open the Windows File Manager or Explorer. Arrange your windows so that either the Hyperion Essbase Application Manager or its icon is visible.

2. Locate and select the desired data sources.

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 displays. This is the dialog box where you specify how to load data or build dimensions.

   Note: If the data source contains blank fields for data values, replace them with `#MI` or `#MISSING`. Otherwise, the data will not load correctly. To replace a blank field with `#MI` or `#MISSING`, see "Replacing an Empty Field with Text" on page 22-15.

Specifying How to Load Data or Build Dimensions

After you select the data sources, specify how Hyperion Essbase loads those data sources and whether to build dimensions dynamically using the Data Load dialog box. If you have not chosen your data sources yet, see "Choosing the Data Sources Using the Hyperion Essbase Application Manager" on page 23-3 or "Choosing the Data Sources Using Windows" on page 23-7.

You can set the following options:

- Modify Outline.
- Load Data

If you are loading data without a rules file, skip to "Setting the Error Log File" on page 23-12.

Using a Rules File with the Data Source

Rules files perform operations on the data as it is loaded, such as moving fields or building new dimensions.

➤ To build dimensions or load SQL data, you must use a rules file.

1. Select your data sources. If you have not chosen your data sources yet, see "Choosing the Data Sources Using the Hyperion Essbase Application Manager" on page 23-3 or "Choosing the Data Sources Using Windows" on page 23-7.
2. From the Data Load dialog box, select Use Rules.
3. Click the Find button to open the Open Server Rules Object dialog box.

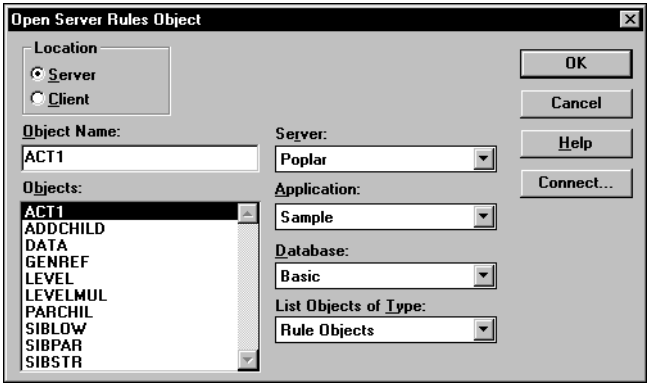


Figure 23-5: Open Server Rules Object Dialog Box

4. Make sure the appropriate Hyperion 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, select "Abort on error during data load."

Building Dimensions Dynamically by Modifying the Outline

You can modify the outline using a data source and rules file. This lets you change or add new dimensions and members to the database based on data in your data source instead of by using the Outline Editor. You must use a rules file to change the outline.

CAUTION: Modifying the outline restructures your database.

To change the outline, select the following options in the Data Load dialog box:

- Modify Outline.

  Hyperion 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 13, “Introducing Dynamic Dimension Building.”

  Note: If Modify Outline is not selected, Hyperion Essbase rejects any records containing new members during the data load.

- Interactive.

  Each time a data source fails, Hyperion Essbase tells you which data source failed to load and asks you if you want to continue reading the remaining data sources.

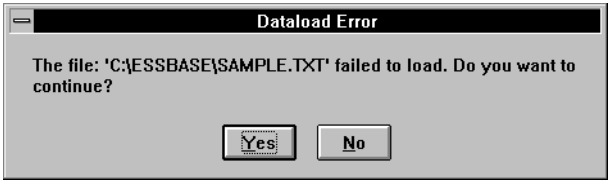


Figure 23-6: Dataload Error Dialog Box

To continue loading the remaining data sources, click Yes. To stop, click No. Any data sources loaded before you stop are in the database.

Check the error log to determine why Hyperion Essbase did not load the data source or perform the dimension build operation. See “Finishing the Data Load or Dimension Build” on page 23-13 if you do not know how to do this.

Updating the Database Outline in Batch Mode

After you create a dimension build rules file, you may want to automate the process of updating dimensions. You can modify the outline, load data, and calculate databases using a batch job. See Chapter 44, “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.

CAUTION: If the Server Output File text box is blank, Hyperion Essbase does not capture errors.

For more information on errors during data loading or dimension building, see “Finishing the Data Load or Dimension Build” on page 23-13.

Now you can start loading data or building dimensions.

Starting the Data Load or Dimension Build

After you have set the data load options in the Data Load dialog box, you can start loading the data sources or building dimensions dynamically.

- For data loads, make sure Load Data is selected.
- For dimension builds, make sure Modify Outline is selected.

Click OK.

To speed up or optimize a data load, see the “Optimizing Data Loads” on page 24-7.

Finishing the Data Load or Dimension Build

When the data load or dimension build finishes, Hyperion 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 display.

Complete Load

In a complete load, Hyperion Essbase had no problems loading every specified record in each data source. When data sources load completely, Hyperion Essbase lists the data sources successfully loaded in the following dialog box.
In Figure 23-7, for example, the `Calcdat` file loaded successfully.

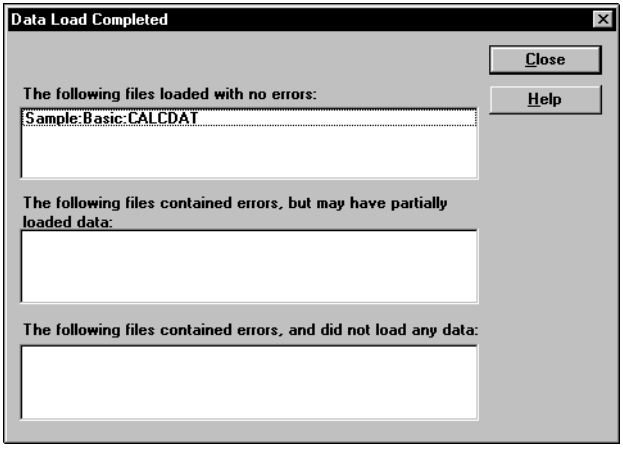


Figure 23-7: Data Load Completed Dialog Box

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. Hyperion Essbase lists all free-form data loads that fail here.

In Figure 23-8, for example, the Act1 text file did not load successfully.

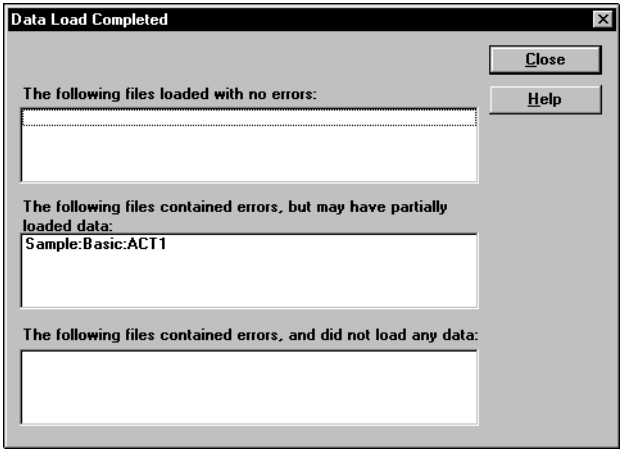


Figure 23-8: Partial Data Load

To fix the data source

1. Open the error log file. It is 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 23-12.

   Note: Hyperion 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 did not load. Figure 23-9 is the error log that Hyperion 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 23-9: Error Log for Partial Data Load

3. After you determine what did not load, open the data sources or records that did not load and fix them. To fix the data source in Figure 23-9, 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 22-3.

   To view a specific record in a data source from the Data Prep Editor, see “Setting the Records Displayed” on page 21-20.

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 24-5 for more information.

No Load

When no data sources or records were loaded into the database, the following dialog box displays:

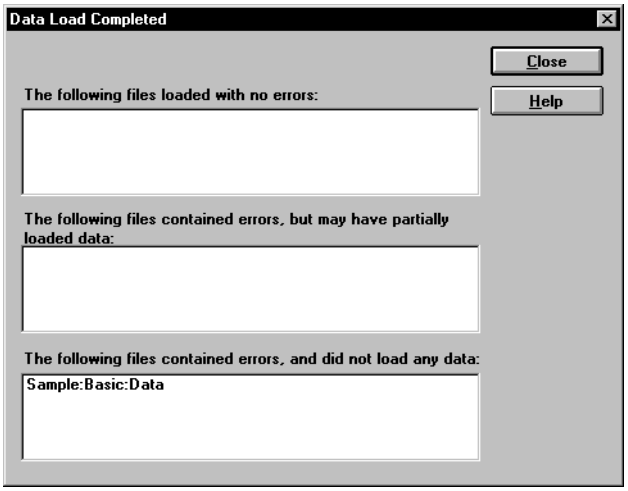


Figure 23-10: No Data Loaded

➤ To fix the data sources:

1. Open the error log file for the data load. If you did not set an error log, set one and restart the load. See “Setting the Error Log File” on page 23-12.
2. Browse through the error log file. It contains a list of each of the data sources and records that did not load. In Figure 23-10, for example, `Data` did not load.
3. After you determine what did not load, open the data sources or records that did not load and fix them. To jump directly to a record in a data source, see “Setting the Records Displayed” on page 21-20.
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, do not load data directly into a parent.
- If you must load data into the parent member, make sure Hyperion Essbase knows not to aggregate `#MISSING` values from children into their parents. Set this at the database level through Hyperion Essbase Application Manager or use the SETDBSTATEITEM command in ESSCMD. You can also use the SET AGGMISSG command in calc scripts. See the online *Technical Reference* in the `DOCS` directory for more information.

  This only works if the children's values are empty (`#MISSING`). If the children have data values, those values will still overwrite the values of the parent. See "Calculating #MISSING Values" on page 33-35 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 22-2.

4. Define a rejection criterion to reject all records except those you want to load. For example, reject all records where the ignored column is less than 250 and greater than 500. See "Rejecting Records" on page 21-16.

   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 the `DOCS` directory for more information.

Loading Data Using a Spreadsheet

If you load data using a spreadsheet, see the following documents:

- *Hyperion Essbase Spreadsheet Add-in User's Guide* for your spreadsheet.
- If you want to use VBA functions, you must also use the Hyperion Essbase Visual Basic API. See ESBIMPORT( ) in the online *API Reference* in the `DOCS` directory. See the Spreadsheet Add-in Online Help for information about using VBA with the Visual Basic API.

Chapter 24

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 data loads. This chapter contains the following sections:

Debugging a Data Load

If you try to load a data source into Hyperion Essbase OLAP Server, but it does not load correctly, check the following:

- Are you connected to the appropriate application and database?
- Did you try to load the correct data source?

If the answer to those questions is yes, then there is probably something wrong. When you have trouble loading a data source, look at the error log file generated for that data load. It lists the errors that occurred when Hyperion 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 23-12 for information on setting up an error log. By default, Hyperion Essbase creates an error log when you load data using a rules file.
- Are you sure that the data source and Hyperion Essbase server are available? See the "Verifying that the Server Is Available" on page 24-4 and "Verifying that the Data Source Is Available" on page 24-4 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 24-5.

If the error log file exists but is empty, Hyperion Essbase does not think that an error occurred during loading. Check the following:

- Does the rules file contain selection/rejection criteria that rejected every record in the data source? See “Selecting Records” on page 21-15 if you do not know how to set up selection/rejection criteria.
- Is the rules file correct? Does the rules file validate properly? See “Problems Validating a Rules File” on page 24-6.

Not All Errors Are in the Log File

When Hyperion Essbase cannot load a record, it writes the record to the error log file, `DATALOAD.ERR`, on the client. There is a limit to the number of records that an error log can contain. The default limit is 1000 records, but you can set the limit to be lower than 1000 by setting DATAERRORLIMIT in the `ESSBASE.CFG` file. See the online *Technical Reference* in the `DOCS` directory for more information.

When Hyperion Essbase writes the maximum allowed number of records in the error log file, it does not log any other errors it encounters. The data load, however, continues. Any subsequent errors are lost.

Data Loaded Incorrectly

If the data source loads correctly, but the data in the database is wrong, check the following:

- Are you sure that you loaded the correct data source? If so, check the data source again to make sure it contains the correct values.
- Are there any blank fields in the data source? You must insert `#MI` or `#MISSING` into a data field that has no value. Otherwise, the data source may not load correctly. To replace a blank field with `#MI` or `#MISSING`, see “Replacing an Empty Field with Text” on page 22-15.

- 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 13-32.
- Did you add incoming data to existing data instead of replacing it using a rules file? See "Changing Data Values" on page 22-22.
- Have you selected or rejected any records that you did not intend to select or reject using a rules file? See Chapter 21, "Setting up a Rules File to Manipulate Records."
- If the sign is reversed (for example, a minus sign instead of a plus sign), did you perform any sign flips on UDAs (user-defined attributes) in your rules file using a rules file? See "Flipping Field Signs" on page 22-27.
- Did you clear data combinations that you did not intend to clear using a rules file? See "Clearing Existing Data Values" on page 22-23.
- Did you scale the incoming values incorrectly using a rules file? See "Scaling Data Values" on page 22-25.
- Are all member and alias names less than 79 characters long?

  Note: You can check data by exporting it, running a report on it, or by using a spreadsheet. To do exports and reports, see Chapter 36, "Developing Report Scripts" and Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD." To use a spreadsheet, see the *Hyperion Essbase Spreadsheet Add-in User's Guide* for your particular spreadsheet.

After you fix the problem with the database or the rules file, you can load just the records that failed by loading the error log. See "Loading the Error Log File" on page 24-5.

Verifying that the Server Is Available

Try to access the server without using Hyperion Essbase to help identify if the problem is with Hyperion Essbase and not with your server or network. Check the following:

- Is the server machine running? Try to connect to it without using Hyperion Essbase. If you cannot, check with your system administrator.
- Is the Hyperion Essbase server running? Check with your Hyperion Essbase administrator.
- Can the client machine connect to the server machine? Try to connect to the server machine from the client machine without using Hyperion Essbase.

Verifying that the Data Source Is Available

If Hyperion 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. Hyperion 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 Hyperion 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. Hyperion 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 42-2.

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`. Hyperion 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 is called `DIMBUILD.ERR`.

2. Load the `DATALOAD.TXT` file using the same rules file you used for the original data sources. If you do not know how to load a data source, see Chapter 20, "Introducing Data Loading."

Recovering from a Server Crash

If the server crashes while you are loading data, Hyperion Essbase sends you a time-out error. If you are overwriting the values in the database with the data source, reload the data sources after the server is running again.

If the Isolation Level transaction setting is Committed, you must re-start the data load from the beginning. If the Isolation Level is Uncommitted, and you are adding to or subtracting from the existing values in the database when the server crashes, do the following:

1. Determine how much data Hyperion 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 42-2.

Problems Validating a Rules File

If you cannot 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 are not also in the data records.
- You did not 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 Hyperion 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 22-4 for information on how to do this.
3. Define a rejection criterion that rejects the end of file marker. See "Rejecting Records" on page 21-16 for information on how to do this.

Optimizing Data Loads

Loading large data sources into an Hyperion 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 Hyperion Essbase Load Data?

When Hyperion 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, Hyperion Essbase does not 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 is 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 24-1, for illustration purposes, displays its fields such that you can easily see the sparse, non-attribute dimensions on the left and dense dimensions on the right, based on the structure of the Sample Basic database.

Note: Because you do not load data into attribute dimensions, even though they are set as sparse, they are not relevant to this discussion.

Sparse, non-attribute dimensions:

- Scenario
- Product
- Market

Dense dimensions:

- Measures
- Year

```
Scenario   Product   Market   Measures   Year
Actual     Cola      Ohio     Sales      Jan    25
```

Figure 24-1: 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 24-2, for example, sorts the records to put like records together. The values for the Measures dense dimension change first. Hyperion 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 24-2: Sorted Records

Figure 24-3, on the other hand, does not sort its records and changes its sparse dimensions before the dense ones. It loads more slowly than Figure 24-2, because Hyperion 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 24-3: Unsorted Records

If you are using a data source that loads more than one cell per record, use the same idea as Figure 24-2, 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 20-8. 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 Hyperion Essbase stores sparse and dense dimensions. All data is stored in blocks. A block contains cells for all possible dense dimension intersections. Hyperion 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, Hyperion Essbase uses the index to find the block where the data is stored and loads that block into its cache. Hyperion 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 do not 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 Hyperion Essbase accesses data using the index. Hyperion Essbase uses the index cache size to determine how much of the index can be paged into memory. Hyperion 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 the index cache size is large enough to hold the index in memory, then this method does not speed data loading.

For more information about setting the index cache size, see "Using Hyperion Essbase Application Manager for Database Settings" on page 41-5. For more information about choosing the size of the index cache, see Chapter 15, "Estimating Disk and Memory Requirements for a 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 Hyperion Essbase must read before loading data values. Figure 24-4 shows a file that does not use ranges and Figure 24-5 shows the same file optimized to use ranges. Figure 24-4 contains 32 fields that Hyperion 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 24-4: Data Source Without Ranges

Figure 24-5 contains only 22 fields that Hyperion Essbase must read in order to load the data values properly. In ranges, the last range cycles the fastest. In Figure 24-5, 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 Hyperion Essbase

assigns the first value, 4, to Jan, New York, Cola. Hyperion Essbase assigns the second value, 3, to Jan, New York, Diet Cola. The third value, 8, is assigned to Jan, Ohio, Cola. Hyperion 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 24-5: 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 Hyperion Essbase to read it. Therefore, as long as a data source is complete, the smaller it is, the faster Hyperion 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 Hyperion Essbase to read it. Therefore, as long as a data source is complete, the smaller it is, the faster Hyperion Essbase can read and load it.

Part V

Calculating 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 Hyperion Essbase calculates dimensions and members, calculate data values dynamically, calculate time series data, create calculation scripts, and optimize calculations. Part V contains the following chapters:

- Chapter 25, "Introduction to Database Calculations," introduces you to the basic concepts behind database calculations.
- Chapter 26, "Developing Formulas," introduces you to formulas and describes how to create formulas on members using the Formula Editor.
- Chapter 27, "Examples of Formulas," contains detailed examples of formulas.
- Chapter 28, "Defining the Calculation Order," describes how to set the calculation order of the members in a database.
- Chapter 29, "Dynamically Calculating Data Values," describes how to set Hyperion Essbase to calculate the values for dimensions and members when they are requested by users, instead of in advance.
- Chapter 30, "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 31, "Developing Calc Scripts," introduces you to calc scripts and describes how to create calc scripts using the Calc Script Editor.
- Chapter 32, "Examples of Calc Scripts," contains detailed examples of calc scripts.

- Chapter 33, “Optimizing Calculations,” describes how to make your calc scripts execute more quickly.
- Chapter 34, “Using Intelligent Calculation to Optimize Calculation,” describes how to use intelligent calculation to make your calc scripts execute more quickly.

Chapter 25

Introduction to Database Calculations

This chapter describes how to calculate a database. It also explains the concept of calculating a multidimensional database.

This chapter includes the following sections:

Understanding Database Calculations

A database contains two types of values. It contains the values that you enter, which are called *input data*, and the values that have been calculated from the input data.

Consider the following examples:

- You enter regional sales figures for a variety of products. You calculate the total sales for each product.
- You enter the budget and actual values for the Cost of Goods Sold for several products in several regions. You calculate the variance between budget and actual values for each product in each region.
- Your database contains regional sales figures and prices for all your products. You calculate what happens to your total profit if you increase the price of one product in one region by 5%.

Hyperion Essbase offers two ways that you can calculate a database:

- Outline calculation
- Calc script calculation

Which way you choose depends on the type of calculation that you want to do.

Outline Calculation

Outline calculation is the simplest method of calculation. Hyperion Essbase bases the calculation of the database on the relationships between members in the database outline and on any formulas that have been associated with members in the outline.

For example, Figure 25-1 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 to calculate the value for East. The values for East, West, South, and Central are added to calculate the total value for Market.

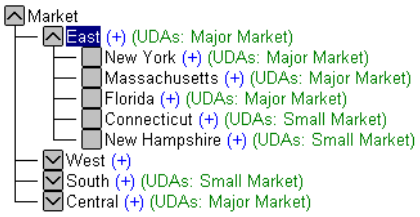


Figure 25-1: Relationship Between Members of the Market Dimension

Figure 25-2 shows the Scenario dimension from the Sample Basic database. The Variance and Variance % members are calculated using the formulas attached to them.

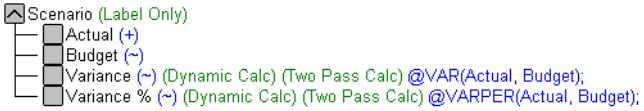


Figure 25-2: Calculation of Variance and Variance %

For more information on creating database outlines, see Chapter 8, “Creating and Changing Database Outlines.”

When you design an overall database calculation, it may be more efficient to calculate some member combinations when you retrieve the data, instead of pre-calculating the member combinations during the regular database calculation. You can use dynamic calculations to calculate data at retrieval time. For more information, see Chapter 29, “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 a database. For example, you can calculate part of a 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%.

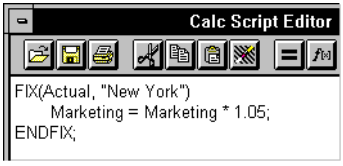


Figure 25-3: Calc Script Editor

For more information on calc scripts, see Chapter 31, “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:

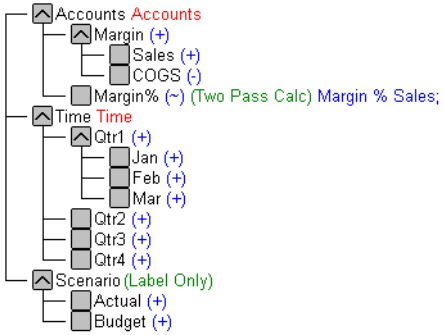


Figure 25-4: Calculating a Multidimensional Database

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. The example displays only the members in Qtr1: Jan, Feb, and Mar.

The Scenario dimension has two child members: Budget for budget values and Actual for actual values.

An intersection of members (one member on each dimension) represents a data value. Our example has three dimensions; therefore, the dimensions and data values in the database can be represented as a cube:

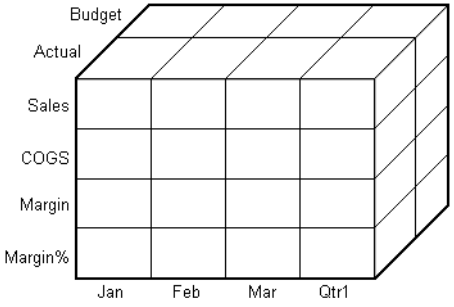


Figure 25-5: Three-Dimensional Database

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:

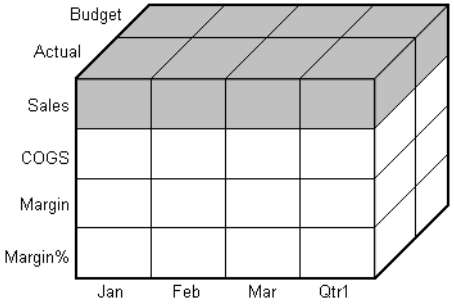


Figure 25-6: Sales, Actual, Budget Slice of the Database

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

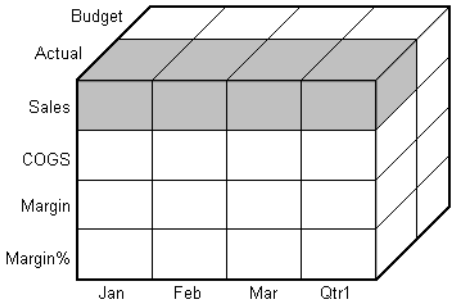


Figure 25-7: Actual, Sales Slice of the Database

To refer to a specific data value in a multidimensional database, you need to specify its member on each dimension. A data value is stored in a single cell in the database. In Figure 25-8, the cell containing the data value for Sales, Jan, Actual is shaded.

In Hyperion Essbase, member combinations are denoted by a cross-dimensional operator. The symbol for the cross-dimensional operator is ->. So Sales, Jan, Actual is written Sales->Jan->Actual.

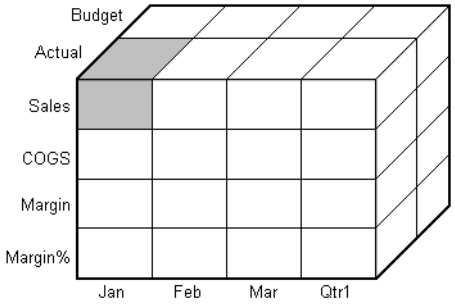


Figure 25-8: Sales, Jan, Actual Slice of the Database

When Hyperion Essbase calculates the formula “Margin% = Margin % Sales,” it takes each Margin value and calculates it as a percentage of its corresponding Sales value.

Hyperion Essbase cycles through the database and calculates Margin% as follows:

1. Margin->Jan->Actual as a percentage of Sales->Jan->Actual. The result is placed in Margin%->Jan->Actual.
2. Margin->Feb->Actual as a percentage of Sales->Feb->Actual. The result is placed in Margin%->Feb->Actual.
3. Margin->Mar->Actual as a percentage of Sales->Mar->Actual. The result is placed in Margin%->Mar->Actual.
4. Margin->Qtr1->Actual as a percentage of Sales->Qtr1->Actual. The result is placed in Margin%->Qtr1->Actual.
5. Margin->Jan->Budget as a percentage of Sales->Jan->Budget. The result is placed in Margin%->Jan->Budget.
6. Hyperion Essbase continues cycling through the database until it has calculated Margin% for every combination of members in the database.

For more information on database calculation order, see Chapter 28, “Defining the Calculation Order.”

Setting the Default Calculation

By default, the default calculation for a database is a CALC ALL of the database outline. CALC ALL consolidates all dimensions and members and calculates all formulas in the outline.

However, you can specify any calc script as the default database calculation. Thus, you can assign a frequently-used script to the database rather than loading the script each time you want to perform its calculation. Also, if you want a 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 in Hyperion Essbase Application Manager:

1. Select Database > Set Default.

   Hyperion Essbase displays the Set Default Calc dialog box.

2. Select Use Calc Script Object.
3. From the list of available calc scripts, select a calc script.
4. Click OK.

You can use the SETDEFAULTCALCFILE command in ESSCMD to perform this task. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, “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 the `DOCS` directory for information about this command. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Calculating a Database

You can calculate a database using any one of these tools:

- Hyperion Essbase Application Manager
- Hyperion Essbase Spreadsheet Add-in
- ESSCMD

This section includes information on calculating a database using Hyperion Essbase Application Manager and ESSCMD. For information on calculating a database from Hyperion Essbase Spreadsheet Add-in, see the *Hyperion Essbase Spreadsheet Add-in User’s Guide*.

Running a Calculation

With Hyperion Essbase Application Manager, you can run the default outline calculation from the desktop.

➤ To calculate a database from Hyperion Essbase Application Manager:

1. In the server dialog box (in this case, Aspen), select the appropriate application and database.

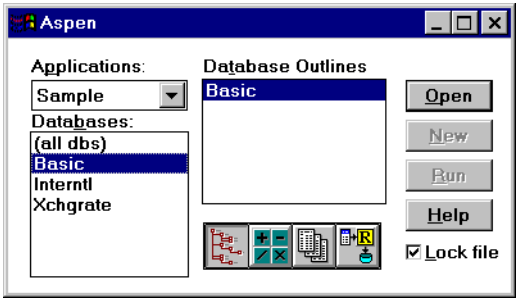


Figure 25-9: Server Dialog Box

2. From the Hyperion Essbase Application Manager menu, select Database > Calculate.

Hyperion Essbase displays the Calculate Database dialog box.

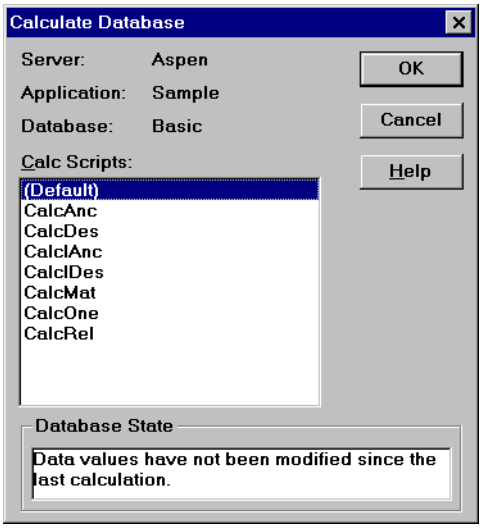


Figure 25-10: Calculate Database Dialog Box

Under Database State, Hyperion Essbase indicates the current calculation state of the database:

Calculation State	Description
Calculation in progress.	Hyperion Essbase is currently calculating the database.
Data values have been modified since the last calculation.	Data values have been changed in the database since the database was last calculated. The last calculation may have been an entire calculation of the database or a calculation of a subset of the database.
Data values have not been modified since the last calculation.	No data values have been changed in the database since the database was last calculated. The last calculation may have been an entire calculation of the database or a calculation of a subset of the database.

3. Select "(Default)" from the list to run the default outline calculation. Or, select a calc script to run the calc script.

   The list includes only calc scripts to which you have security access. For information on Hyperion Essbase security privileges, see Chapter 17, "Managing Security at Global and User Levels."

4. Click OK.

   Hyperion Essbase calculates the database.

   Note: Hyperion Essbase displays a message while it is calculating the database. For lengthy calculations, you may want Hyperion Essbase to perform the calculations in the background. Use the Windows Alt + Tab key combination to switch to another Windows application. While Hyperion Essbase is calculating the database, other Hyperion Essbase 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 the `DOCS` directory for information about these commands. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Canceling a Calculation

➤ To stop a calculation before Hyperion Essbase completes it, click the Cancel button in the Calculate Database dialog box.

When you cancel a calculation, Hyperion Essbase does one of the following:

- Reverts all values to their previous state
- Retains any values calculated before the cancellation

How Hyperion Essbase handles the cancellation depends on your Hyperion Essbase Kernel Isolation Level settings. For more information on these settings, see Chapter 42, "Ensuring Data Integrity."

Considering Security

In order to calculate a database, you must have calculate privileges for the database outline.

If you have calculate privileges, you can calculate any value in the database. With calculate privileges, you can calculate a value even if a security filter denies you read and update privileges. Careful consideration should be given to providing users with calculate privileges.

For more information on providing users with calculate privileges and on security filters, see Chapter 17, “Managing Security at Global and User Levels.”

Chapter 26

Developing Formulas

This chapter explains how to develop and use formulas to calculate a database. It provides detailed examples of formulas, which you may want to adapt for your own use. For more examples, see Chapter 27, “Examples of Formulas.”

This chapter includes the following sections:

Using Formulas

Formulas calculate relationships between members in a database outline. You can use formulas in two ways:

- Apply them to members in the database outline.
- Place them in a calc script.

In most cases you can optimize calculation performance by applying formulas to members in the database outline. However, if you need to control database calculations more carefully, you can use a calc script and include formulas. For more information, see Chapter 31, “Developing Calc Scripts.”

Your use of formulas can have significant implications for calculation performance. Consider the information in Chapter 33, “Optimizing Calculations” before designing and creating formulas.

The following figure shows the Measures dimension from the Sample Basic database. The Margin %, Profit %, and Profit per Ounce members are calculated using the formulas applied to them.

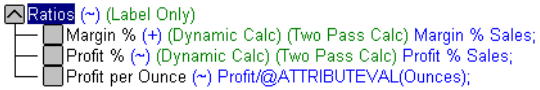


Figure 26-1: Calculation of Margin %, Profit %, and Profit per Ounce

Hyperion Essbase provides a comprehensive set of operators and functions, which you can use to construct formula calculations on a 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 formulas:

Operator	Description
Mathematical	Perform common arithmetic operations. For example, you can add, subtract, multiply, or divide values. For a complete list of the mathematical operators, see the online *Technical Reference* in the DOCS directory.
Conditional	Control the flow of formula executions based on the results of conditional tests. For example, you can use an IF statement to test for a specified condition. For a list of the conditional operators, see the online *Technical Reference* in the DOCS directory. For more information on writing conditional formulas, see "Specifying Conditions" on page 26-29.
Cross-dimensional	Point to the data values of specific member combinations. For example, point to the sales value for a specific product in a specific region. For more information, see "Using the Cross-Dimensional Operator (->)" on page 26-46.

See "Inserting Text and Operators in a Formula" on page 26-15 for information on how to add operators to formulas.

Functions

Functions are predefined routines that perform specialized calculations and return sets of members or data values. The following table shows the types of functions you can use in formulas:

Operator	Description
Boolean	Provide a conditional test by returning either a TRUE (1) or FALSE (0) value. For example, you can use the @ISMBR function to determine whether the current member is one that you specify. For more information, see Chapter 27, "Examples of Formulas."
Mathematical	Perform specialized mathematical calculations. For example, you can use the @AVG function to return the average value of a list of members. For more information, see Chapter 27, "Examples of Formulas."
Relationship	Look up data values within a database during a calculation. For example, you can use the @ANCESTVAL function to return the ancestor values of a specified member combination. For more information, see Chapter 27, "Examples of Formulas."
Range	Declare a range of members as an argument to another function or command. For example, you can use the @SUMRANGE function to return the sum of all members that lie within a specified range. For more information, see Chapter 27, "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 27, "Examples of Formulas."
Member Set	Generate a list of members that is based on a specified member. For example, you can use the @ICHILDREN function to return a specified member and its children. For more information, see the online *Technical Reference* in the `DOCS` directory.

Operator	Description
Allocation	Allocate values that are input at a parent level across child members. You can allocate values within the same dimension or across multiple dimensions. For example, you can use the @ALLOCATE function to allocate sales values that are input at a parent level to the parent's children; each child's allocation is determined by its share of the previous year's sales.
Forecasting	Manipulate data for the purposes of smoothing or interpolating data, or calculating future values. For example, you can use the @TREND function to calculate future values that are based on curve-fitting to historical values.
Statistical	Calculate advanced statistics. For example, you can use the @RANK function to calculate the rank of a specified member or a specified value in a data set.
Date and Time	Use date and time characteristics in calculation formulas. For example, you can use the @TODATE function to convert date strings to numbers that can be used in calculation formulas.

For a complete list of operators, functions, and syntax, see the online *Technical Reference* in the `DOCS` directory.

Calculating Formulas

For formulas applied to members in a database outline, Hyperion Essbase calculates formulas when you do the following:

- Run a default (CALC ALL) calculation of a database.
- 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 itself. For more information, see Chapter 31, "Developing Calc Scripts."

For a formula in a calc script, Hyperion Essbase calculates the formula when it occurs in the calc script.

If a formula is associated with a dynamically-calculated member, Hyperion Essbase calculates the formula when the user requests the data values. In a calc script, you cannot calculate a dynamically-calculated member or make a dynamically-calculated member the target of a formula calculation. For more information, see Chapter 29, “Dynamically Calculating Data Values.”

Using dynamically-calculated members in a formula on a database outline or in a calc script can significantly affect calculation performance. Performance is affected because Hyperion Essbase has to interrupt the regular calculation to perform the dynamic calculation.

You cannot use substitution variables in formulas that you apply to the database outline. For more information, see “Using Substitution Variables” on page 26-48.

Guidelines for 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 a member name in double quotation marks ("") if the member name meets any of the following conditions:

  – Contains spaces

    For example,

    ```
    "Opening Inventory" = "Ending Inventory" - Sales + Additions;
    ```

  – Is the same as an operator or function name. See the online *Technical Reference* in the `DOCS` directory for a list of operators and functions.

  – Includes any non-alphanumeric character; for example, hyphens (-), asterisks (*), and slashes (/).

  – Is all numeric or starts with one or more numerals; for example, “100” or “10Prod”

  For a complete list of member names that must be enclosed in quotation marks, see “Rules for Naming Dimensions and Members” on page 8-15.

- End each IF statement in a formula with an ENDIF statement.

  For example, the following formula contains a simple IF... ENDIF statement. You can apply this formula to the Commission member in a database outline:

  ```
  IF(Sales < 100)
     Commission = 0;
  ENDIF;
  ```

  If you are using an IF statement nested within another IF statement, end each IF with an ENDIF. For example:

  ```
  "Opening Inventory"
  (IF (@ISMBR(Budget))
     IF (@ISMBR(Jan))
     "Opening Inventory" = Jan;
     ELSE
     "Opening Inventory" = @PRIOR("Ending Inventory");
     ENDIF;
  ENDIF;)
  ```

- You do not need to end ELSE or ELSEIF statements with ENDIFs. For example:

  ```
  IF (@ISMBR(@DESCENDANTS(West)) OR @ISMBR(@DESCENDANTS(East)
     Marketing = Marketing * 1.5;
  ELSEIF(@ISMBR(@DESCENDANTS(South)))
     Marketing = Marketing * .9;
  ELSE Marketing = Marketing * 1.1;
  ENDIF;
  ```

  Note: If you use ELSE IF (with a space in between) rather than ELSEIF (one word) in a formula, you must supply an ENDIF for the IF statement.

- Although ending ENDIF statements with a semicolon (;) is not required, it is good practice to follow each ENDIF statement in a formula with a semicolon.

When writing formulas, you can check the syntax using the Formula Editor syntax checker. For more information, see “Checking Syntax” on page 26-25.

For detailed information on syntax for Hyperion Essbase functions and commands, see the online *Technical Reference* in the `DOCS` directory.

Creating Formulas

This section provides a step-by-step example of creating and saving a simple formula in an outline. For an example of creating a formula in a calc script, see Chapter 31, “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 the Hyperion Essbase installation. If you do not have Sample Basic installed, contact your Hyperion Essbase administrator.

This example shows you how to create a formula on the Variance member of the Scenario dimension. This formula calculates the variance between Budget values and Actual values.

➤ To create the example formula:

1. Start Hyperion Essbase Application Manager, and connect to your Hyperion Essbase server.
2. Select the Sample application and the Basic database, and click Open to open the Sample Basic outline.

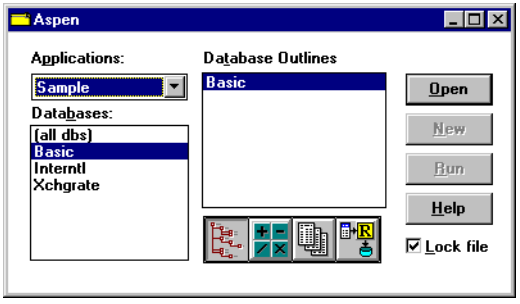


Figure 26-2: Application Desktop Window

If another user has Sample Basic open and locked, you can clear “Lock file” in the bottom right-hand corner of the application desktop window. However, if you clear “Lock file,” you cannot save your work.

3. Double-click the Scenario dimension to display its members.

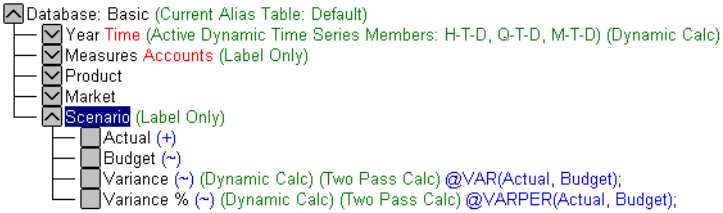


Figure 26-3: Scenario Dimension in Sample Basic Outline

4. Select the Variance member in the outline, and click the  button.

   Hyperion Essbase displays the formula in Formula Editor.

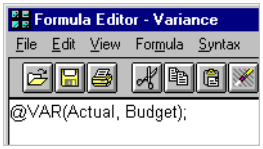


Figure 26-4: Formula Editor Showing Variance Formula

5. To re-create the formula, select the existing formula and select Edit > Delete in Formula Editor.

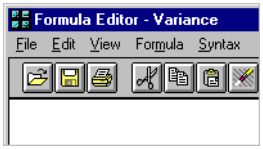


Figure 26-5: Formula Editor With Variance Formula Deleted

6. In the Dimensions list, select Scenario.

   Hyperion Essbase displays Scenario in the Members list.

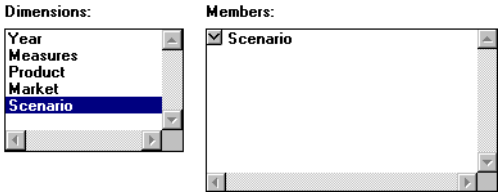


Figure 26-6: Formula Editor Dimensions and Members Lists With Scenario Selected

7. In the Members list, double-click the button next to Scenario to display the members under Scenario.

8. Select Formula > Paste Function or click the button.

   Hyperion Essbase displays the Function Templates dialog box.

9. In the Categories list, select Math.

10. In the Templates list, select @VAR.

    Hyperion Essbase displays the function and the default arguments below the Categories list.

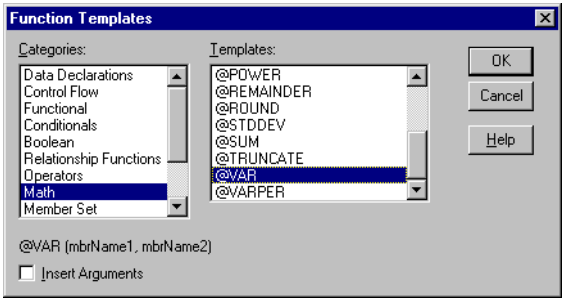


Figure 26-7: Function Templates Dialog Box

11. Check Insert Arguments to insert default, temporary arguments in Formula Editor.
12. Click OK.

    Hyperion Essbase inserts `@VAR(mbrName1, mbrName2)` at the cursor position.

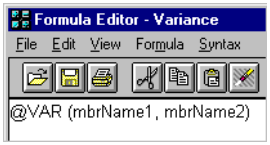


Figure 26-8: Formula Editor With Variance Formula Added

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 Formula Editor.
16. Click the  button to save the changes to the outline.

You have recreated the formula on Variance in Sample Basic.

Building Formulas in Formula Editor

You use Formula Editor in Hyperion Essbase Application Manager to create formulas. You can type the formulas directly into the formula text area, or you can use the Formula Editor user interface features to create the formula.

Formulas are ASCII text. If required, you can create a formula in the text editor of your choice and paste it into Formula Editor.

Opening Formula Editor

Open Formula Editor to create new formulas or open existing ones.

➤ To open Formula Editor from Hyperion Essbase Application Manager:

1. In Outline Editor, highlight the member whose formula you want to create or edit.

2. Select Edit > Formula or click the Formula Editor button,  .

   Hyperion Essbase opens Formula Editor for the selected member. If the member already has a formula, the formula is displayed in Formula Editor. The following figure shows Formula Editor for the Variance member in the Sample Basic database.

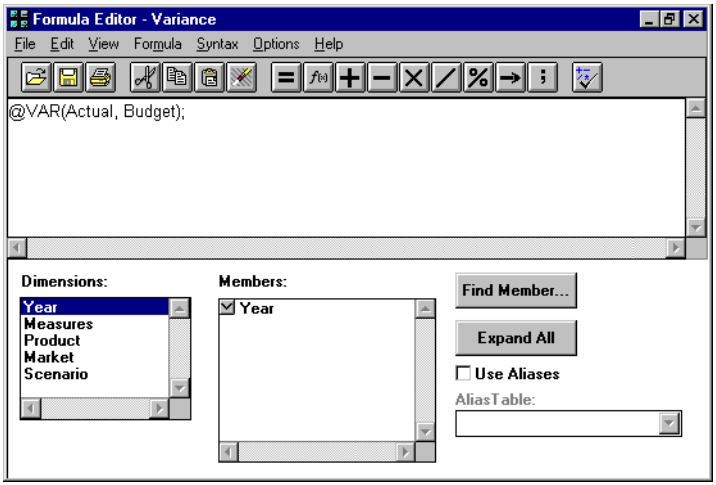


Figure 26-9: Formula Editor Window

Displaying a Formula

Open the database outline to display members and their associated formulas in Outline Editor. You can also highlight the member for which you want to see a formula and click the = button to open Formula Editor.

You can use the GETMBRCALC command in ESSCMD to display member formulas. See the online *Technical Reference* in the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

Adding a Formula

You can use Hyperion Essbase Application Manager to add a formula to a database outline.

➤ To add a formula:

1. Highlight the member for which you want to add a formula.
2. Click the = button to open Formula Editor.
3. Type or insert the formula in the Formula Editor window. See "Inserting Text and Operators in a Formula" on page 26-15.

Changing a Formula

To change an existing formula, open it in Formula Editor.

➤ To change a formula:

1. Highlight the member that has the formula you want to edit.
2. Click the = button to open Formula Editor.
3. Make the required changes to the formula.

Saving a Formula

You can save formulas to the database outline.

➤ To save a formula after you have created or opened it:

1. In Formula Editor, select File > Save or click the  button to save the changes in Formula Editor.
2. Close Formula Editor.
3. Click the  button in Outline Editor to save the changes in the database outline.

   Hyperion Essbase displays the formula beside the member in the database outline.

Printing a Formula

You can print the contents of a formula from Formula Editor.

➤ To print a formula:

In Formula Editor, select File > Print, or click the  button.

Deleting a Formula

You can delete a formula that has been saved to the database outline.

➤ To delete a formula:

1. In Outline Editor, select the member with the formula that you want to delete.
2. To open Formula Editor, click the  button.

   Hyperion Essbase displays the formula in the Formula Editor window.
3. Select the text of the formula.
4. Select Edit > Delete to delete the text of the formula.

5. Click the  button to save the changes in Formula Editor.

6. Close Formula Editor and click the  button to save the changes in the database outline.

   Hyperion Essbase no longer displays the formula beside the member in the database outline.

➤ To undo the last action:

In Formula Editor, select Edit > Undo, or click the  button.

Inserting Text and Operators in a Formula

You can type text and operators directly into the Formula Editor text area, or you can use the toolbar buttons to add the text and operators. You can also copy, cut, and search for text in Formula Editor.

➤ To type text in Formula Editor:

1. In Formula Editor, click in the formula text area below the toolbar.
2. Type the appropriate text.

   Text is displayed at the cursor position as you type.

Figure 26-10: Adding a Formula in Formula Editor

➤ To insert an equal sign (=) in a formula:

1. In Formula Editor, place the cursor where you want to insert the equal sign (=).
2. Type = or click the  button.

➤ To insert a mathematical operator (+, -, X, /, %) in a formula:

1. In Formula Editor, place the cursor where you want to insert the mathematical operator.
2. Type the operator or click one of the following toolbar buttons:

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.

➤ To insert the cross-dimensional operator (->) in a formula:

1. In Formula Editor, place the cursor where you want to insert the cross-dimensional operator.
2. Type a **–** (hyphen) followed by a **>** (greater than symbol), or click the  button.

For more information on the cross-dimensional operator, see “Using the Cross-Dimensional Operator (->)” on page 26-46.

➤ To insert the semicolon formula end-of-line character (;) in a formula:

1. In Formula Editor, place the cursor at the end of the formula.
2. Type a **;** (semicolon) or click the  button.

➤ To insert a function or operator in a formula:

1. In Formula Editor, place the cursor where you want to insert the function.
2. Select Formula > Paste Function, or click the  button.

   Hyperion Essbase displays the Function Templates dialog box.
3. In the Categories list, select a function category. For example, to insert the @VAR function, select Math.
4. In the Templates list, select the required function or operator. For example, scroll down the list and select @VAR.

   Hyperion Essbase displays the function or operator and the default arguments below the Categories list.

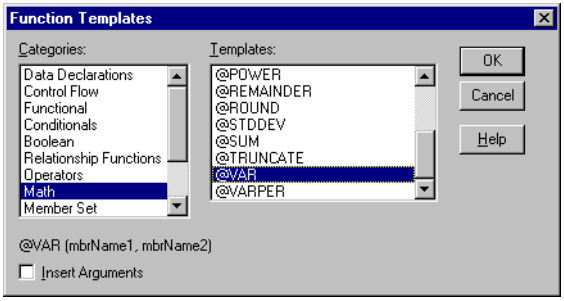


Figure 26-11: Function Templates Dialog Box With Math Category Selected

5. If required, select Insert Arguments to insert default, temporary arguments in the function.

6. Click OK.

   Hyperion Essbase inserts @VAR ( ) at the cursor position.

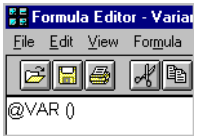


Figure 26-12: Formula Editor Showing @VAR Formula

If you checked Insert Arguments, Hyperion Essbase inserts @VAR and default, temporary arguments. You can then type over the default arguments with the correct arguments.

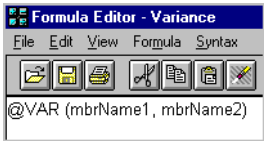


Figure 26-13: Formula Editor Showing @VAR with Arguments

➤ To cut text in Formula Editor:

Select the text that you want to cut and do one of the following:

- Select Edit > Cut.
- Click the  button.
- Press Ctrl + X.

➤ To copy text in Formula Editor:

Select the text that you want to copy and do one of the following:

- Select Edit > Copy.
- Click the  button.
- Press Ctrl + C.

➤ To paste text in Formula Editor:

Select the text that you want to paste and do one of the following:

- Select Edit > Paste.
- Click the  button.
- Press Ctrl + V.

➤ To find and replace text in Formula Editor:

1. In Formula Editor, select Edit > Find.
2. Hyperion Essbase displays the Find dialog box.

Figure 26-14: Formula Editor Find Dialog Box

3. In the Find what text box, type the characters that you want to search for.
4. Click the Find Next button.

➤ To do a case-sensitive search in Formula Editor:

1. In the Find dialog box, select Match case.

   For example, to search for Margin but not “margin”, type **Margin** in the Find what text box, and select Match case.
2. Click Find Next.

Inserting Members in a Formula

You can insert dimension and member names in Formula Editor instead of typing them.

➤ To insert a dimension name in a formula:

1. In Formula Editor, place the cursor where you want to insert the dimension name.
2. In the Dimensions list, select the dimension that you want to insert in the formula.

   The dimension name displays in the Members list. If a button displays to the left of the dimension name, then the dimension has children. The following shows the Scenario dimension in the Sample Basic database.

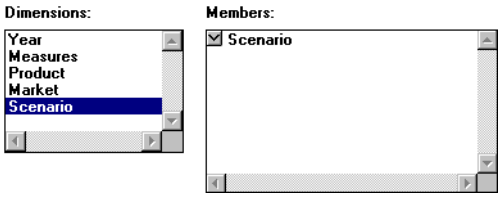


Figure 26-15: Formula Editor Dimensions and Members Lists

3. To insert a dimension name in the formula, click the dimension name in the Members list.

   Hyperion Essbase inserts the dimension name at the cursor position. To insert a member name (a member name other than the dimension name), expand the member branch and select the member you want to insert.

➤ To expand a member branch to display a member's children:

In the Members list, double-click the button next to the member name to display the member's children.

The button changes to a button.

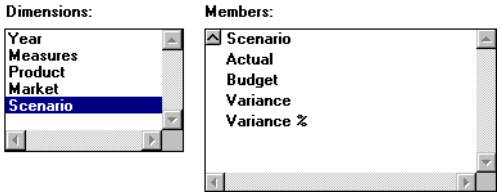


Figure 26-16: Formula Editor Dimensions and Members Lists, Expanding the Scenario Member

Double-click the button to collapse the member branch.

➤ To collapse a member branch:

In the Members list, double-click the button to collapse the member branch.

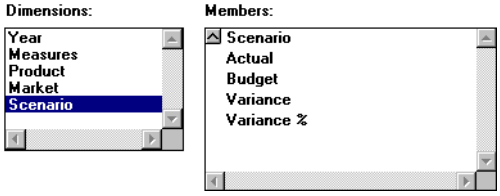


Figure 26-17: Formula Editor Dimensions and Members Lists, Expanded Scenario Dimension

The button changes to a button. Hyperion Essbase does not display the member's children:

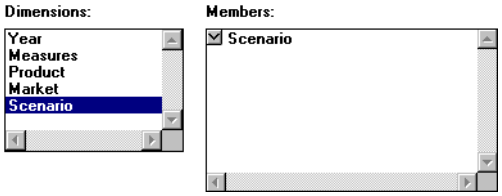


Figure 26-18: Formula Editor Dimensions and Members Lists, Collapsing the Scenario Dimension

Searching for Members

➤ To search for a specific member in Formula Editor:

1. In the Dimensions list, select the dimension that you want to search for a member.

   For example, select the Measures dimension from the Sample Basic database.

2. Click Find Member.

   Hyperion Essbase displays the Find dialog box.

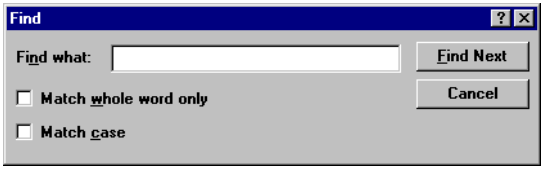


Figure 26-19: Formula Editor Find Dialog Box

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, type **market**.

   a. To search for whole words only, select Match whole word only.

      For example, to search for Margin, but not Margin %, type **margin** in the Find what text box, and select Match whole word only.

   b. To search for case-sensitive characters, select Match case.

      For example to search for Margin, but not margin, type **Margin** in the Find what text box, and select Match case.

4. Click Find Next.

   Hyperion Essbase finds and selects the Marketing member.

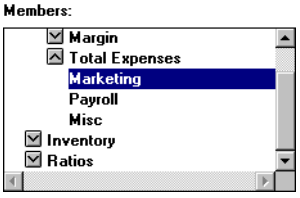


Figure 26-20: Formula Editor Members List With Marketing Selected

➤ To expand a dimension to display all members in Formula Editor:

1. In the Dimensions list, select the dimension for which you want to display all members.

   For example, select the Product dimension in the Sample Basic database.

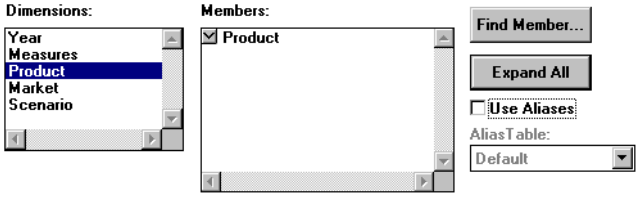


Figure 26-21: Formula Editor Dimensions and Members List With Product Selected

2. Click Expand All.

   In the Members list, Hyperion Essbase displays all members in the dimension.

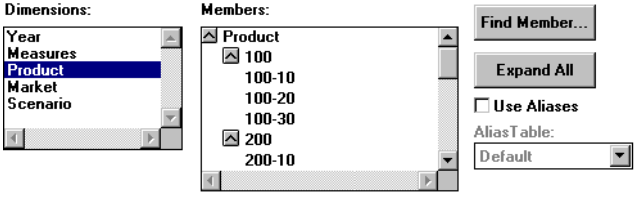


Figure 26-22: Formula Editor Dimensions and Members List Showing the Children of Product

➤ To display and insert alias names in Formula Editor:

Check Use Aliases.

Hyperion Essbase displays the alias names for the members. The following example shows the Product dimension from the Sample Basic database.

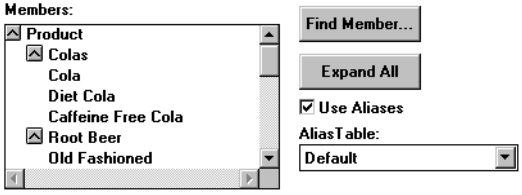


Figure 26-23: Formula Editor Dimensions and Members List with Alias Names

To select a different alias table, from the Alias Table list box, select a table.

When you select a member from the Members list, Hyperion Essbase inserts the alias name at the cursor position. If required, Hyperion Essbase automatically encloses the alias name in double quotation marks ("").

Checking Syntax

Hyperion Essbase includes a formula syntax checker that tells you about any syntax errors in formulas. For example, Hyperion Essbase tells you if you have mistyped a function name.

The syntax checker cannot tell you about semantic errors in a formula. Semantic errors occur when a formula does not work as you expect. To find semantic errors, run the calculation and check the results to ensure that they are as you expect.

➤ To check the syntax of a formula in Formula Editor:

Select Syntax > Check, or click the  button.

Hyperion Essbase displays the syntax checker results at the bottom of the Formula Editor window. If Hyperion Essbase finds no syntax errors, it displays the following message:

Figure 26-24: Formula Editor Syntax Checker, No Errors Message

If Hyperion Essbase finds one or more syntax errors, it displays the number of the line that includes the error and a brief description of the error. For example, if you do not include a semicolon end-of-line character at the end of a formula, Hyperion Essbase displays a message similar to the following:

Figure 26-25: Formula Editor Syntax Checker, Syntax Error Message

➤ To step through errors in Formula Editor:

Select Syntax > Next Error or Syntax > Previous Error.

When you reach the first or last error, Hyperion Essbase displays the message:

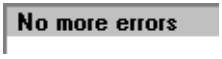


Figure 26-26: Formula Editor Syntax Checker, No More Errors Message

Hyperion Essbase retains the list of error messages until you check the syntax again.

If a formula passes validation in Formula Editor or Outline Editor, but the server detects semantic errors when the outline is saved:

- The incorrect formula is saved as part of the outline, even though it contains errors.
- The server writes a message in the application log file that indicates what the error is and displays the incorrect formula.
- The server writes an error message to the comment field of the member associated with the incorrect formula. The message indicates that the incorrect formula was not loaded. You can view this comment in Outline Editor by closing and reopening the outline.
- If you do not correct the member formula, and a calculation that includes that member is run, the formula is ignored during the calculation.

After you have corrected the formula and saved the outline, the message in the member comment is deleted. You can view the updated comment when you reopen the outline.

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 cross-dimensional operators in formulas. For more examples of formulas, see Chapter 27, “Examples of Formulas.”

Before writing formulas, review the guidelines in “Guidelines for Formula Syntax” on page 26-6.

Writing Basic Equations

You can apply a mathematical operation to a formula to create a basic equation. For example, you can apply the following formula to the Margin member in Sample Basic.

```
Sales - COGS;
```

In a calc script, you define basic equations as follows:

Member = *mathematical_operation*;

where *Member* is a member name from the database outline and *mathematical_operation* is any valid mathematical operation. For example:

```
Margin = Sales - COGS;
```

Whether the example equation is in the database outline or in a calc script, Hyperion Essbase cycles through the database subtracting the values in COGS from the values in Sales and placing the results in Margin.

As another example, you can apply the following formula to a Markup member:

```
(Retail - Cost) % Retail;
```

In a calc script, this would be:

```
Markup = (Retail - Cost) % Retail;
```

In this example, Hyperion Essbase cycles through the database subtracting the values in Cost from the values in Retail, calculating the resulting values as a percentage of the values in Retail, and placing the result in Markup.

For more information on the nature of multidimensional calculations, see Chapter 25, “Introduction to Database Calculations.”

Specifying Conditions

You can define formulas that use a conditional test or a series of conditional tests to control the flow of calculation.

The IF and ENDIF commands define a *conditional block*. The formulas between the IF and the ENDIF commands are executed only if the test returns TRUE (1). You can use the ELSE and ELSEIF commands to specify alternative actions if the test returns FALSE (0). The formulas following each ELSE command are executed only if the previous test returns FALSE (0). Conditions following each ELSEIF command are tested only if the previous IF command returns FALSE (0).

For more information on the syntax of the IF and ENDIF commands, see "Guidelines for Formula Syntax" on page 26-6.

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 command, you can use functions that return TRUE or FALSE (1 or 0, respectively) based on the result of a conditional test. These functions are known as *Boolean functions*.

You use Boolean functions to determine which formula to use. The decision is based on the characteristics of the current member combination. For example, you might want to restrict a certain calculation to the members in the Product dimension that contain input data. In this case, you preface the calculation with an IF test based on @ISLEV(Product,0).

If one of the function parameters is a cross-dimensional member, such as @ISMBR(Sales->Budget), all of the parts of the cross-dimensional member must match the properties of the current cell to return a value of TRUE (1).

You can use the following Boolean functions to specify conditions.

To determine if...	Use the function...
The current member has a specified accounts tag (for example, an Expense tag)	@ISACCTYPE
The current member is an ancestor of the specified member	@ISANCEST
The current member is an ancestor of the specified member, or the specified member itself	@ISIANCEST
The current member is a child of the specified member	@ISCHILD

To determine if...	Use the function...
The current member is a child of the specified member, or the specified member itself	@ISICHILD
The current member is a descendant of the specified member	@ISDESC
The current member is a descendant of the specified member, or the specified member itself	@ISIDESC
The current member of the specified dimension is in the generation specified	@ISGEN
The current member of the specified dimension is in the level specified	@ISLEV
The current member matches any of the specified members	@ISMBR
The current member is the parent of the specified member	@ISPARENT
The current member is the parent of the specified member, or the specified member itself	@ISIPARENT
The current member (of the same dimension as the specified member) is in the same generation as the specified member	@ISSAMEGEN
The current member (of the same dimension as the specified member) is in the same level as the specified member	@ISSAMELEV
The current member is a sibling of the specified member	@ISSIBLING
The current member is a sibling of the specified member, or the specified member itself	@ISISIBLING
A specified user-defined attribute (UDA) exists for the current member of the specified dimension	@ISUDA

When you place formulas on the database outline, you can use only the IF, ELSE, ELSEIF, and ENDIF commands and Boolean functions to control the flow of the calculations. You can use additional control commands in a calc script.

For more information on calc scripts, see Chapter 31, “Developing Calc Scripts.” For more information on Hyperion Essbase functions and calculation commands, see the online *Technical Reference* in the `DOCS` directory.

Examples of Specifying Conditions

You can apply the following formula to a Commission member in the database outline. In the first example, the formula calculates commission at 1% of sales if the sales are greater than 500000:

```
IF(Sales > 500000)
Commission = Sales * .01;
ENDIF;
```

If you place the formula in a calc script, you need to associate the formula with the Commission member as follows:

```
Commission(IF(Sales > 500000)
Commission = Sales * .01;
ENDIF;)
```

Hyperion 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, Hyperion Essbase multiplies the value in Sales by 0.01 and places the result in Commission.

In the next example, the formula tests the ancestry of the current member and then applies the appropriate Payroll calculation formula.

```
IF(@ISIDESC(East) OR @ISIDESC(West))
Payroll = Sales * .15;
ELSEIF(@ISIDESC(Central))
Payroll = Sales * .11;
ELSE
Payroll = Sales * .10;
ENDIF;
```

If you place the formula in a calc script, you need to associate the formula with the Payroll member as follows:

```
Payroll(IF(@ISIDESC(East) OR @ISIDESC(West))
Payroll = Sales * .15;
ELSEIF(@ISIDESC(Central))
Payroll = Sales * .11;
ELSE
Payroll = Sales * .10;
ENDIF;)
```

Hyperion 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, Hyperion 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, Hyperion 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, Hyperion 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 25, “Introduction to Database Calculations.” For more information on the @ISIDESC function, see the online *Technical Reference* in the `DOCS` directory.

Using Interdependent Values

Hyperion Essbase optimizes calculation performance by calculating formulas for a range of members in the same dimension at the same time. However, some formulas require values from members of the same dimension, and Hyperion Essbase may not yet have calculated the required values.

A good example is that of cash flow, in which the opening inventory is dependent on the ending inventory from the previous month.

In Sample Basic, the Opening Inventory and Ending Inventory values need to be calculated on a month-by-month basis.

	Jan	Feb	Mar
Opening Inventory	100	120	110
Sales	50	70	100
Addition	70	60	150
Ending Inventory	120	110	160

Assuming that the Opening Inventory value for January is loaded into the database, the required calculation is:

1. January Ending = January Opening – Sales + Additions
2. February Opening = January Ending
3. February Ending = February Opening – Sales + Additions
4. March Opening = February Ending
5. March Ending = March Opening – Sales + Additions

You can calculate the required results by applying interdependent, multiple equations to a single member in the database outline.

The following formula, applied to the Opening Inventory member in the database outline, calculates the correct values:

```
IF(NOT @ISMBR (Jan))
    "Opening Inventory" = @PRIOR("Ending Inventory");
ENDIF;
"Ending Inventory" = "Opening Inventory" - Sales + Additions;
```

If you place the formula in a calc script, you need to associate the formula with the Opening Inventory member as follows:

```
"Opening Inventory" (IF(NOT @ISMBR (Jan))
    "Opening Inventory" = @PRIOR("Ending Inventory");
ENDIF;
"Ending Inventory" = "Opening Inventory" - Sales + Additions;)
```

Hyperion Essbase cycles through the months, performing the following calculations:

1. The IF statement and @ISMBR function check that the current member on the Year dimension is not Jan. This step is necessary because the Opening Inventory value for Jan is an input value.
2. If the current month is not Jan, the @PRIOR function obtains the value for the previous month's Ending Inventory. This value is then allocated to the current month's Opening Inventory.
3. The Ending Inventory is calculated for the current month.

Note: To calculate the correct results, it is necessary to place the above formula on a single member, Opening Inventory. If you place the formulas for Opening Inventory and Ending Inventory on their separate members, Hyperion 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 you may need to specify a range of members. For example, the **@**ISMBR function tests to see if a member that is currently being calculated matches any of a list or range of specified members. You can specify members using the following syntax:

To specify...	Use...
A single member	The member name. For example: Mar97
A list of members	A comma-delimited (,) list of member names. For example: Mar97, Apr97, May97
A range of all members at the same level, between and including the two defining members	The two defining member names separated by a colon (:). For example: Jan97:Dec97
A range of all members in the same generation, between and including the two defining members	The two defining member names separated by two colons (::). For example: Q197::Q491
A function-generated list of members or a range of members	See “Generating Member Lists” on page 26-36.
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 of members or a range of members in a function that requires either, Hyperion Essbase uses the level 0 members of the dimension tagged as time. If no dimension is tagged as time, Hyperion Essbase displays an error message.

Generating Member Lists

You can generate member lists that are based on a specified member by using the following member set functions.

To generate a list of members including...	Use the function...
All ancestors of the specified member, including ancestors of the specified member as a shared member. This function does not include the specified member.	@ALLANCESTORS
All ancestors of the specified member, including ancestors of the specified member as a shared member. This function includes the specified member.	@IALLANCESTORS
The ancestor of the specified member at the specified generation or level.	@ANCEST
All ancestors of the specified member (optionally up to the specified generation or level) but not the specified member.	@ANCESTORS
All ancestors of the specified member (optionally up to the specified generation or level) including the specified member.	@IANCESTORS
All children of the specified member, but not including the specified member.	@CHILDREN
All children of the specified member, including the specified member.	@ICHILDREN
The current member being calculated for the specified dimension.	@CURRMBR
A range of members that is based on the relative position of the member combination Hyperion Essbase is currently calculating.	@CURRMBRRANGE
All descendants of the specified member (optionally up to the specified generation or level) but not the specified member.	@DESCENDANTS
All descendants of the specified member (optionally up to the specified generation or level) including the specified member.	@IDESCENDANTS

To generate a list of members including...	Use the function...
All members of the specified generation in the specified dimension.	@GENMBRS
All members of the specified level in the specified dimension.	@LEVMBRS
All siblings of the specified member, but not the specified member.	@SIBLINGS
All siblings of the specified member, including the specified member.	@ISIBLINGS
All siblings that precede the specified member in the database outline, but not the specified member.	@LSIBLINGS
All siblings that follow the specified member in the database outline, but not the specified member.	@RSIBLINGS
All siblings that precede the specified member in the database outline, including the specified member.	@ILSIBLINGS
All siblings that follow the specified member in the database outline, including the specified member.	@IRSIBLINGS
Separate lists of members to be processed by functions that require multiple list arguments.	@LIST
A merged list of two member lists to be processed by another function.	@MERGE
A member list that crosses the specified member from one dimension with the specified member range from another dimension.	@RANGE
A list of members from which some members have been removed.	@REMOVE
All members that match the specified wildcard selection.	@MATCH
The parent of the current member being calculated in the specified dimension.	@PARENT
All members of the specified generation or level that are above or below the specified member.	@RELATIVE
All members that have a common user-defined attribute (UDA) defined on the Hyperion Essbase server.	@UDA

To generate a list of members including...	Use the function...
All base members that are associated with the specified attribute member.	@ATTRIBUTE
All base members that are associated with attributes that satisfy the specified conditions.	@WITHATTR

For more information on Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

Performing Mathematical Operations

You can perform many mathematical operations in formulas by using the following mathematical functions.

To calculate...	Use the function...
The absolute value of an expression	@ABS
The average value of the values in the specified member list	@AVG
The factorial of an expression	@FACTORIAL
The next lowest integer value of a member or expression	@INT
The maximum value among the expressions in the specified member list	@MAX
The minimum value among the expressions in the specified member list	@MIN
The modulus produced by the division of two specified members	@MOD
The value of the specified member raised to the specified power	@POWER
The remainder value of an expression	@REMAINDER
A member or expression rounded to the specified number of decimal places	@ROUND
The summation of values of all specified members	@SUM

To calculate...	Use the function...
The truncated value of an expression	@TRUNCATE
The variance (difference) between two specified members. See “Calculating a Variance or Percentage Variance Between Actual and Budget Values” on page 26-39.	@VAR
The percentage variance (difference) between two specified members. See “Calculating a Variance or Percentage Variance Between Actual and Budget Values” on page 26-39.	@VARPER

For more information on Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

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 Hyperion Essbase to show a positive variance if the actual values are lower than the budget values. For example, you want Hyperion Essbase to show a positive variance if actual costs are lower than budgeted costs.

- Non-expense items

  You want Hyperion Essbase to show a negative variance if the actual values are lower than the budget values. For example, you want Hyperion Essbase to show a negative variance if actual sales are lower than budgeted sales.

By default, Hyperion Essbase assumes that members are non-expense items and calculates the variance accordingly.

➤ To tell Hyperion Essbase that a member is an expense item:

1. In Outline Editor, select the member. The member must be on the dimension tagged as accounts. See Chapter 30, “Calculating Time Series Data.”

2. Click the  button.

   Hyperion Essbase tags the member as an expense item. When you use the @VAR or @VARPER functions, Hyperion Essbase shows a positive variance if the actual values are lower than the budget values.

   For example, in Sample Basic, the children of Total Expenses are expense items. The Variance and Variance % members of the Scenario dimension calculate the variance between the Actual and Budget values.

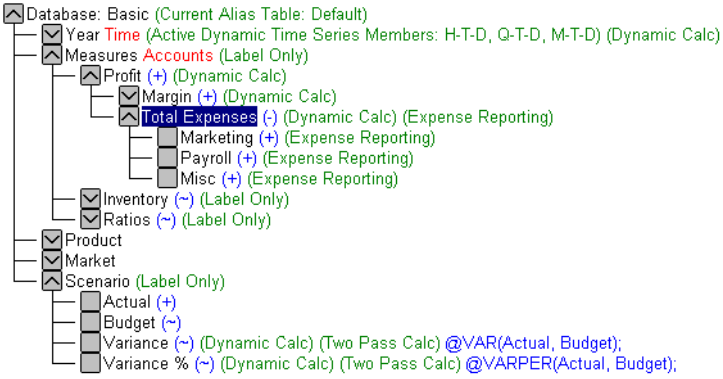


Figure 26-27: Sample Basic Showing Expense Items

Calculating Statistics

You can use the following statistical functions to calculate advanced statistics in Hyperion Essbase.

To calculate...	Use the function...
The correlation coefficient between two parallel data sets	@CORRELATION
The number of values in the specified data set	@COUNT
The median, or middle number, in the specified data set	@MEDIAN
The mode, or the most frequently occurring value, in the specified data set	@MODE
The rank of the specified member or value in the specified data set	@RANK
The standard deviation, based upon a sample, of the specified members	@STDEV
The standard deviation, based upon the entire population, of the specified members	@STDEVP
The standard deviation, crossed with a range of members, of the specified members	@STDEVRANGE
The variance, based upon a sample, of the specified data set	@VARIANCE
The variance, based upon the entire population, of the specified data set	@VARIANCEP

For more information on Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

Allocating and Forecasting Values

You can allocate values that are input at the parent level across child members in the same dimension or in different dimensions by using the following allocation functions.

To allocate...	Use the function...
Values from a member, cross-dimensional member, or value across a member list within the same dimension. The allocation is based on a variety of specified criteria.	@ALLOCATE
Values from a member, cross-dimensional member, or value across multiple dimensions. The allocation is based on a variety of specified criteria.	@MDALLOCATE

Note: For examples of calc scripts using the @ALLOCATE and @MDALLOCATE functions, see “Allocating Values Within or Across Dimensions” on page 32-9 and the online *Technical Reference* in the `DOCS` directory.

You can manipulate data for the purposes of smoothing data, interpolating data, or calculating future values by using the following forecasting functions.

To do the following...	Use the function...
Apply a moving average to a data set and replace each term in the list with a trailing average. This function modifies the data set for smoothing purposes.	@MOVAVG
Apply a moving maximum to a data set and replace each term in the list with a trailing maximum. This function modifies the data set for smoothing purposes.	@MOVMAX
Apply a moving median to a data set and replace each term in the list with a trailing median. This function modifies the data set for smoothing purposes.	@MOVMED
Apply a moving minimum to a data set and replace each term in the list with a trailing minimum. This function modifies the data set for smoothing purposes.	@MOVMIN

To do the following...	Use the function...
Apply a smoothing spline to a set of data points. A spline is a mathematical curve that is used to smooth or interpolate data.	@SPLINE
Calculate future values and base the calculation on curve-fitting to historical values.	@TREND

For more information on Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

Using Range Functions

You can execute a function for a range of members by using the following range functions.

To calculate...	Use the function...
The average value of a member across a range of members	@AVGRANGE
The maximum value of a member across a range of members	@MAXRANGE
The minimum value of a member across a range of members	@MINRANGE
The summation of values of all specified members across a range of members	@SUMRANGE

For more information on Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

Using the Current Member Combination to Look Up Values

You can use the member combination that Hyperion Essbase is currently calculating to look up specific values. These functions are referred to as *relationship* functions.

To look up...	Use the function...
The numeric value associated with a level 0 attribute member from a numeric, Boolean, or date attribute dimension, for the current member combination	@ATTRIBUTEVAL
The generation number of the current member combination for the specified dimension	@CURGEN
The level number of the current member combination for the specified dimension	@CURLEV
The generation number of the specified member	@GEN
The level number of the specified member	@LEV
The ancestor values of the specified member combination	@ANCESTVAL
The ancestor values of the specified member combination across multiple dimensions	@MDANCESTVAL
The shared ancestor values of the specified member combination	@SANCESTVAL
The next or *n*th member in a range of members, retaining all other members identical to the current member and in the specified dimension	@SHIFT
The next or *n*th member in a range of members, retaining all other members identical to the current member across multiple dimensions	@MDSHIFT
The next or *n*th member in a range of members	@NEXT
The previous or *n*th previous member in a range of members	@PRIOR
The parent values of the specified member combination	@PARENTVAL

To look up...	Use the function...
The parent values of the specified member combination across multiple dimensions	@MDPARENTVAL
The shared parent values of the specified member combination	@SPARENTVAL
A data value from another database to be used for calculation of a value from the current database	@XREF

For more information on Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

Calculating Financial Functions

You can include financial calculations in formulas by using the following functions.

To calculate...	Use the function...
An accumulation of values up to the specified member	@ACCUM
The proceeds of a compound interest calculation	@COMPOUND
A series of values that represent the compound growth of the specified member across a range of members	@COMPOUNDGROWTH
Depreciation for a specific period, calculated using the declining balance method.	@DECLINE
A value discounted by the specified rate, from the first period of the range to the period in which the amount to discount is found	@DISCOUNT
A series of values that represents the linear growth of the specified value	@GROWTH
The simple interest for a specified member at a specified rate	@INTEREST
The internal rate of return on a cash flow	@IRR

To calculate...	Use the function...
The Net Present Value of an investment (based on a series of payments and incomes)	@NPV
The period-to-date values of members in the dimension tagged as time	@PTD
The amount per period that an asset in the current period may be depreciated (calculated across a range of periods). The depreciation method used is straight-line depreciation.	@SLN
The amount per period that an asset in the current period may be depreciated (calculated across a range of periods). The depreciation method used is sum of the year's digits.	@SYD

For more information on Hyperion Essbase functions, see the online *Technical Reference* in the `DOCS` directory.

Using Date and Time Functions

You can use dates with other functions by using the following date function.

To...	Use the function...
Convert date strings to numbers that can be used in calculation formulas	@TODATE

Using the Cross-Dimensional Operator (->)

The cross-dimensional operator points to data values of specific member combinations.

You create the cross-dimensional operator using a hyphen (-) and a greater than symbol (>). Do not leave spaces in between the cross-dimensional operator and the member names.

For example, in this simplified illustration, the shaded data value is Sales->Jan->Actual.

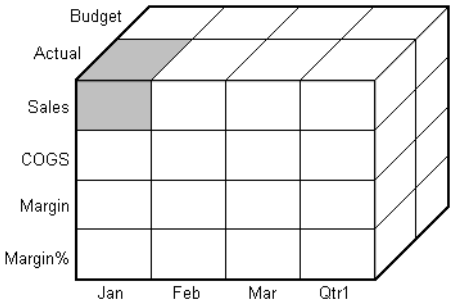


Figure 26-28: Defining a Single Data Value by Using the Cross-Dimensional Operator

The following example illustrates how to use the cross-dimensional operator. This example allocates miscellaneous expenses to each product in each market.

The value of Misc_Expenses for all products in all markets is known. The formula allocates a percentage of the total Misc_Expenses value to each Product->Market combination. The allocation is based on the value of Sales for each product in each market.

```
Misc_Expenses = Misc_Expenses->Market->Product * (Sales /
(Sales->Market->Product));
```

Hyperion Essbase cycles through the database, performing the following calculation:

1. Hyperion Essbase divides the Sales value for the current member combination by the total Sales value for all markets and all products (Sales->Market->Product).
2. It multiplies the value calculated in step 1 by the Misc_Expenses value for all markets and all products (Misc_Expenses->Market->Product).
3. It allocates the result to Misc_Expenses for the current member combination.

Consider carefully how you use the cross-dimensional operator, as it can have significant performance implications. For detailed information, see Chapter 33, “Optimizing 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, Hyperion Essbase replaces the substitution variable with the value you have assigned to it. You can create and assign values to substitution variables using Hyperion Essbase Application Manager or ESSCMD.

You can set substitution variables at the server, application, and database levels. Hyperion 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.

Hyperion 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 can use the following **@**ISMBR function as part of a conditional test in a calc script:

```
@ISMBR(&UpToCurr)
```

Before Hyperion Essbase runs the calc script, it replaces the substitution variable, as follows:

```
@ISMBR(Jan:Jun)
```

Working with Formulas in Partitions

A Hyperion Essbase partition can span multiple servers, processors, or computers. For more information on partitioning, see Chapter 6, “Designing Partitioned Applications,” and Chapter 16, “Building and Maintaining Partitions.”

You can use formulas in partitioning, just as you use formulas on your local database. However, if a formula you use in one database references a value from another database, Hyperion Essbase has to retrieve the data from the other database when calculating the formula. In this case, you need to ensure that the referenced values are up-to-date and to consider carefully the performance impact on the overall database calculation. For more information, see the information on writing calc scripts for partitions in “Writing Calc Scripts for Partitions” on page 31-57.

With transparent partitions, you need to consider carefully how you use formulas on the data target. For more information, see “Transparent Partitions and Member Formulas” on page 6-22 and “Performance Considerations for Transparent Partitions” on page 6-20.

Chapter 27

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 32, “Examples of Calc Scripts.”

This chapter includes the following sections:

Calculating Period-to-Date Values

You can use the @PTD function to calculate period-to-date values. You can also use Dynamic Time Series members to calculate period-to-date values. For detailed information, see Chapter 30, “Calculating Time Series Data.”

For example, the following figure shows the Inventory branch of the Measures dimension from the Sample Basic database.

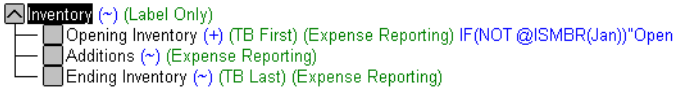


Figure 27-1: Inventory Branch from Sample Basic Outline

To calculate period-to-date values for the year and for the current quarter, you add two members to the Year dimension, QTD for quarter-to-date and YTD for year-to-date:

Figure 27-2: Calculating Period-to-Date Values

Assuming that 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 as appropriate. However, Opening Inventory has a time balance tag, First, and Ending Inventory has a time balance tag, Last. Hyperion Essbase takes these values and treats them accordingly. For more information on time balance tags and other accounts tags, see Chapter 30, “Calculating Time Series Data.”

The following table provides an example of the calculation results for the members in the Inventory branch and for the Sales member:

Measures->Time	Jan	Feb	Mar	Apr	May	QTD	YTD
Opening Inventory	100	110	120	110	140	**110**	**100**
Additions	110	120	100	160	180	**340**	**670**
Sales	100	110	110	130	190	**320**	**640**
Ending Inventory	110	120	110	140	130	**130**	**130**

The values for Sales and Additions have been summed.

Opening Inventory has a First tag. For QTD, Hyperion Essbase takes the first value in the current quarter, which is Apr. For YTD, Hyperion Essbase takes the first value in the year, which is Jan.

Ending Inventory has a Last tag. For QTD, Hyperion Essbase takes the last value in the current quarter, which is May. For YTD, Hyperion Essbase takes the last value in the year, which is also May.

Calculating Rolling Values

You can use the @AVGRANGE function to calculate rolling averages and the @ACCUM function to calculate rolling year-to-date values.

For example, assume that a database contains monthly Sales data values and that the database outline includes the members AVG_Sales and YTD_Sales.

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);
```

Hyperion Essbase calculates the average Sales values across the months in the dimension tagged as time. The SKIPNONE parameter means that all values are included, even `#MISSING` values. Hyperion Essbase places the results in AVG_Sales. For more information on `#MISSING` values, see Chapter 33, "Optimizing Calculations."

Hyperion 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 use the @PRIOR function to calculate values based on a previous month's value.

For example, assume that a database contains assets data values that are stored on a month-by-month basis. You can calculate the difference between the assets values of successive months (the asset movement) by subtracting the previous month's value from the present month's value.

The following example shows three members:

- Assets for the monthly asset values
- Asset_MVNT for the asset movement values
- Opening_Balance for the asset value at the beginning of the year

For Jan, the Asset_MVNT value is calculated by subtracting the Opening_Balance value from the Jan value.

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**

Hyperion Essbase cycles through the months, performing the following calculations:

1. The IF statement and @ISMBR function check to see if the current member on the Year dimension is Jan. This check is necessary because the Asset_MVNT value for Jan cannot be calculated by subtracting the previous month's value.
2. If the current member on the Year dimension is Jan, Hyperion 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. Hyperion Essbase subtracts the previous month's assets from the current month's assets. It places the result in the current month's Asset_MVNT value.

Testing for #MISSING Values

You can test for #MISSING values in a database. For more information on #MISSING values, see Chapter 33, "Optimizing Calculations."

Assume that a database outline contains a member called Commission. Commission is paid at 10% of sales when the Sales value for the current member combination is not #MISSING. When applied to a Commission member in the database outline, the following formula calculates Commission:

```
IF(Sales <> #MISSING) Commission = Sales * .9;
ELSE Commission = #MISSING;
ENDIF;
```

If you place the formula in a calc script, you need to associate it with the commission member as follows:

```
Commission(IF(Sales <> #MISSING) Commission = Sales * .1;
ELSE Commission = #MISSING;
ENDIF;);
```

Hyperion 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, Hyperion Essbase multiplies the value in the Sales member by 0.1 and places the result in the Commission member.
3. If Sales is #MISSING, Hyperion Essbase places #MISSING in the Commission member.

Calculating an Attribute Formula

You can perform specific calculations on attribute members in a database.

Note: For more information about attribute calculations, see “Calculating Attribute Data” on page 10-30.

For example, to calculate profitability by ounce for products sized in ounces, you can use the @ATTRIBUTEVAL function in a calculation formula. In the Sample Basic database, the Ratios branch of the Measures dimension contains a member called Profit per Ounce. The formula on this member is:

```
Profit/@ATTRIBUTEVAL(Ounces);
```

Hyperion Essbase cycles through the Products dimension, performing the following calculations:

1. For each base member that is associated with a member from the Ounces attribute dimension, the @ATTRIBUTEVAL function returns the numeric attribute value (for example, 12 for the member 12 under Ounces).
2. Hyperion Essbase then divides Profit by the result of @ATTRIBUTEVAL to yield Profit per Ounce.

Note: For more information on using attributes in calculation formulas, see “Using Attributes in Calculation Formulas” on page 10-36. For more information about the @ATTRIBUTEVAL function, see the online *Technical Reference* in the `DOCS` directory.

Chapter 28

Defining the Calculation Order

This chapter describes the order in which Hyperion Essbase calculates a database. If you use dynamic calculations, see Chapter 29, “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:

Data Storage in Data Blocks

Hyperion Essbase stores data values in data blocks. Hyperion Essbase creates a data block for each unique combination of sparse dimension members (providing that at least one data value exists for the combination).

Each data block contains all the dense dimension member values for its unique combination of sparse dimension members.

In the Sample Basic database, the Year, Measures, and Scenario dimensions are dense. The Product and Market dimensions are sparse.

Figure 28-1: Dimensions from the Sample Basic Database

Note: Sample Basic also contains five attribute dimensions. These dimensions are sparse, Dynamic Calc, meaning that attribute data is not stored in the database. For more information on attributes, see Chapter 10, “Working with Attributes.”

Hyperion Essbase creates a data block for each unique combination of members in the Product and Market dimensions (providing that at least one data value exists for the combination). For example, it creates one data block for the combination of 100-10, New York. This data block contains all the Year, Measures, and Scenario values for 100-10, New York.

Figure 28-2: Product and Market Dimensions from the Sample Basic Database

In Hyperion Essbase, member combinations are denoted by the cross-dimensional operator. The symbol for the cross-dimensional operator is -> (a hyphen followed by a greater than symbol). So 100-10, New York is written 100-10->New York.

You can categorize data blocks as follows:

- Input—These blocks are created by loading data to cells in a block. Input blocks can be created for (1) sparse, level 0 member combinations or (2) sparse, upper level member combinations, when at least one of the sparse members is a parent level member. Input blocks can be level 0 or upper level blocks.
- Noninput—These blocks are created through calculations. For example, in Sample Basic, the East->Cola block is created during a sparse calculation process (that is, the block did not exist before calculation).
- Level 0—These blocks are created for sparse member combinations when all of the sparse members are level 0 members. For example, in Sample Basic, New York->Cola is a level 0 block because New York and Cola are level 0 members of their respective sparse dimensions. Level 0 blocks can be input or noninput blocks; for example, a level 0 noninput block is created during an allocation process, where data is loaded at a parent level and then allocated down to level 0.
- Upper level—These blocks are created for sparse member combinations when at least one of the sparse members is a parent level member. Upper level blocks can be input or noninput blocks.

For more information on levels and generations, and how Hyperion Essbase stores data in data blocks, see Chapter 40, “Introducing the Hyperion Essbase Kernel.”

Member Calculation Order

Hyperion Essbase calculates a database at the data block level, bringing one or more blocks into memory and calculating the required values within the block. Hyperion Essbase calculates the blocks in order, according to their block numbers. The database outline tells Hyperion Essbase how to order the blocks. Within each block, Hyperion Essbase calculates the values in order according to the hierarchy in the database outline. Therefore, overall, Hyperion Essbase calculates a database based on the database outline.

When you perform a default calculation (CALC ALL) on a database, Hyperion Essbase 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, Hyperion Essbase calculates as follows:

1. The dimension tagged as accounts
2. The dimension tagged as time
3. Other dense dimensions (in the order they are displayed in the database outline)
4. Other sparse dimensions (in the order they are displayed in the database outline)

Otherwise, Hyperion Essbase calculates as follows:

1. Dense dimensions (in the order they display in the database outline)
2. Sparse dimensions (in the order they display in the database outline)

Note: Attribute dimensions, which are not included in the database consolidation, do not affect calculation order. For more information on attribute dimensions, see Chapter 10, “Working with Attributes.”

In the Sample Basic database, the dimensions are calculated in the following order: Measures, Year, Scenario, Product, and Market.

You can override the default order by using a calc script. For more information on developing calc scripts, see Chapter 31, “Developing Calc Scripts.” For more information on accounts and time dimensions, see Chapter 30, “Calculating Time Series Data.”

Member Relationships

The order of calculation within each dimension depends on the relationships between members in the database outline. Within each branch of a dimension, level 0 values are calculated first followed by their level 1, parent value. Then the level 0 values of the next branch are calculated followed by their level 1, parent value. The calculation continues in this way until all levels are calculated.

Figure 28-3 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 29, “Dynamically Calculating Data Values.”

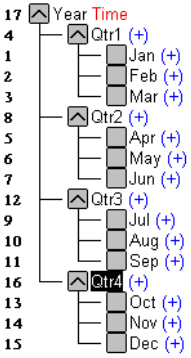


Figure 28-3: Year Dimension from the Sample Basic Database

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, the other two members in the branch.

Hyperion Essbase calculates Qtr1 by consolidating Jan, Feb, and Mar. In this example, these members are added.

Hyperion Essbase then calculates the Qtr2 through Qtr4 branches in the same way.

Finally, Hyperion Essbase calculates the Year member by consolidating the values of Qtr1 through Qtr4. Again, in this example, these members are added.

Member Consolidation

You can choose how Hyperion Essbase consolidates members by applying any unary operator (+, -, /, *, %, ~) to the members in the database outline.

If an accounts member has a time balance tag (First, Last, or Average), Hyperion Essbase consolidates it accordingly. For more information on time balance calculations, see Chapter 30, “Calculating Time Series Data.”

If a parent member has a label only tag, Hyperion Essbase does not calculate the parent from its children. If a member has a ~ tag, Hyperion Essbase does not consolidate the member up to its parent.

Note: If you use dynamic calculations, Hyperion Essbase may use a different calculation order. For information on the calculation order for dynamically-calculated values, see Chapter 29, "Dynamically Calculating Data Values."

Ordering Dimensions in the Database Outline

To ensure the required calculation results, consider the calculation order of the dimensions in the database outline if you do either of the following:

- You use unary operators to divide (/), multiply (*), or calculate percentages (%) for members in the database outline.
- You place formulas on members in the database outline.

You do not need to consider calculation order if you use only unary operators to add (+) and subtract (–) members in the database outline and you do not use formulas in the outline.

Placing Formulas on Members in the Database Outline

If you place formulas on members in the database outline, consider the calculation order of the dimensions. A formula that is attached to a member on one dimension may be overwritten by a subsequent calculation on another dimension.

For example, the Sample Basic database has a Measures dimension, tagged as accounts, and a Year dimension, tagged as time. Measures is calculated first, and Year second. If you attach a formula to Margin on the Measures dimension, Hyperion Essbase calculates the formula when it calculates the Measures dimension. Hyperion Essbase then overwrites the formula when it aggregates the Year dimension. For detailed information, see "Cell Calculation Order" on page 28-14.

Using the Unary Operators *, /, and %

If you use unary operators to multiply (*), divide (/), and calculate percentages (%) for members in the database outline, consider the calculation order of the dimensions. The required calculated values may be overwritten by a subsequent calculation on another dimension.

For example, the Sample Basic database has a Measures dimension, tagged as accounts, and a Year dimension, tagged as time. Measures is calculated first, and Year second. If you multiply members on the Measures dimension, the calculated results may be overwritten when Hyperion Essbase aggregates values on the Year dimension. For detailed information, see “Cell Calculation Order” on page 28-14.

When you use a multiplication (*), division (/), or percentage (%) operator to consolidate members, carefully order the members in the branch to achieve the required result.

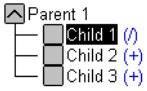


Figure 28-4: Unary Operators in the Database Outline

In the above example, assume that the user wants to divide the total of Child 2 and Child 3 by Child 1. However, if Child 1 is the first member, Hyperion Essbase starts with Child 1, taking the value of Parent 1 (currently `#MISSING`) and dividing it by Child 1. The result is `#MISSING`. Hyperion Essbase then adds Child 2 and Child 3. Obviously, this result is not the required one.

To calculate the correct result, make Child 1 the last member in the branch. For more information on `#MISSING` values, see Chapter 33, “Optimizing Calculations.”

You can apply a formula to a member on the database outline to achieve the same result. However, it is far more efficient to use unary operators on members as in the above example.

Avoiding Forward Calculation References

To obtain the calculation results you expect, ensure that the outline does not contain forward calculation references. *Forward calculation references* occur when the value of a calculating member is dependent on a member that Hyperion Essbase has not yet calculated. In these cases, Hyperion Essbase may not produce the required calculation results.

For example, consider the following Product dimension:

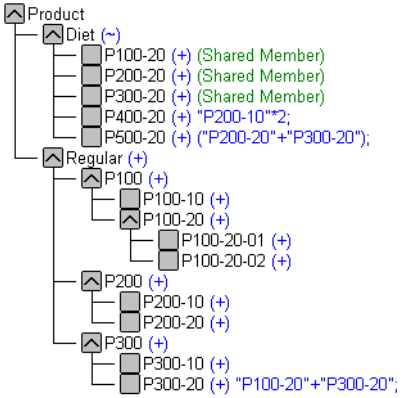


Figure 28-5: Example Product Dimension

This Product dimension has three forward calculation references. Two shared members and one non-shared member have forward calculation references:

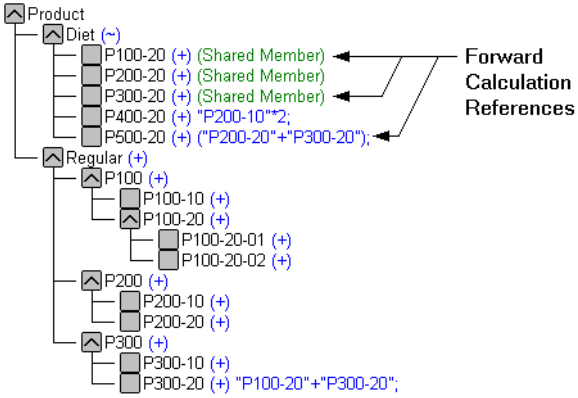


Figure 28-6: Example Product Dimension Showing Forward Calculation References

In the Hyperion Essbase Application Manager Outline Editor, you can select Outline > Verify to identify shared members with forward calculation references. Hyperion Essbase displays these members in the Verify Outline dialog box.

Note: Selecting Outline > Verify does *not* identify non-shared members that have forward calculation references.

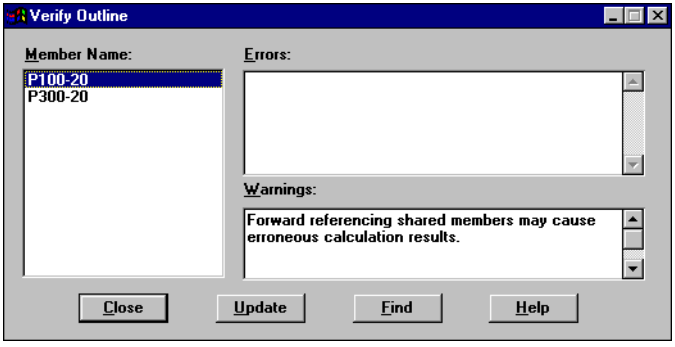


Figure 28-7: Verify Outline Dialog Box Showing Forward Calculation References

You can save and use an outline containing forward calculation references.

Consider the five members under Diet. The members P100-20, P300-20, and P500-20 have forward calculation references:

```
P100-20 (+) (Shared Member)
```

Hyperion Essbase calculates the shared member P100-20 before it calculates the real member P100-20. Because the real member P100-20 has children, Hyperion 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)
```

Hyperion Essbase calculates the shared member P300-20 before it calculates the real member P300-20. Because the real member P300-20 has a formula, Hyperion 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 Hyperion Essbase has not yet calculated. One referenced member, P300-20, has its own formula, and Hyperion Essbase needs to calculate P300-20 before it can accurately calculate P500-20. The members P200-20 and P400-20 calculate correctly, as they do not have forward calculation references:

```
P200-20 (+) (Shared Member)
```

P200-20 is *not* a forward calculation reference, even though Hyperion Essbase calculates the shared member P200-20 before it calculates the real member P200-20. The real member P200-20 has no calculation dependencies (no children and no formula). Therefore Hyperion Essbase does not need to calculate the real member before the shared member. Hyperion Essbase simply takes the value of the real member.

```
P400-20 (+) "P200-10"*2;
```

P400-20 is *not* a forward calculation reference, even though the formula that is applied to P400-20 references a member that Hyperion 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. Hyperion Essbase accurately calculates P400-20.

To get accurate calculation results for P100-20, P300-20, and P500-20, change the order of members in the outline. By placing the Diet shared members after the Regular members, you ensure that Hyperion Essbase calculates the members in the required order.

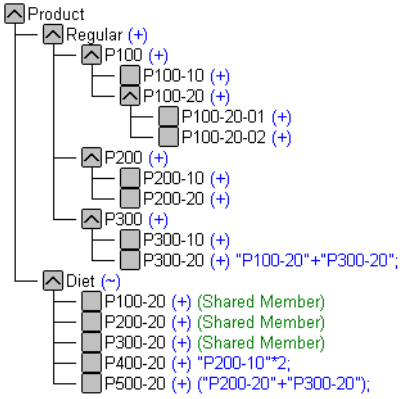


Figure 28-8: Changed Product Dimension Without Forward Calculation References

Now Hyperion Essbase calculates as follows:

- The real member P100-20 before it calculates the shared member P100-20. So, P100-20 no longer has a forward calculation reference.
- The real member P300-20 before the shared member P300-20. So, P300-20 no longer has a forward calculation reference.
- The referenced member with a formula, P300-20, before the member P500-20. So, P500-20 no longer has a forward calculation reference.

Block Calculation Order

Hyperion Essbase calculates blocks in the order in which the blocks are numbered. Hyperion Essbase takes the first sparse dimension in a database outline as a starting point. It defines the sparse member combinations from this first dimension.

In the Sample Basic database, Product is the first sparse dimension in the database outline.

Figure 28-9: Dimensions in the Sample Basic Database

Note: The attribute dimensions in the Sample Basic outline (not shown in the figure above), are not included in the database consolidation and do not affect block calculation order. For more information on attribute dimensions, see Chapter 10, “Working with Attributes.”

Product has 19 members (excluding the shared members, for which Hyperion Essbase does not create data blocks). Therefore, the first 19 data blocks in the database are numbered according to the calculation order of members in the Product dimension.

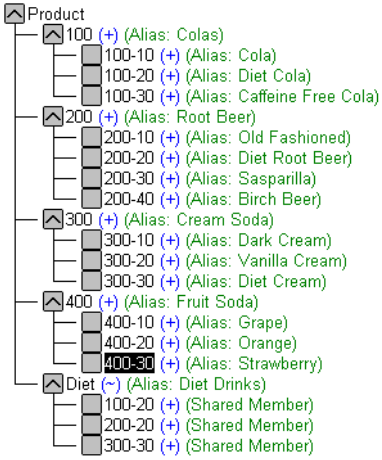


Figure 28-10: Product Dimension from the Sample Basic Database

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. Hyperion 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

Hyperion Essbase continues until blocks have been created for all combinations of sparse dimension members for which at least one data value exists.

Hyperion 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 Hyperion Essbase does not create a data block for 200-10->Massachusetts. However, Hyperion Essbase does reserve the appropriate block number for 200-10->Massachusetts in case data is loaded for that member combination in the future.

When you run a default calculation (CALC ALL) on a database, each block is processed in order, according to its block number. If you have Intelligent Calculation turned on and if the block does not need to be calculated, then Hyperion Essbase skips the block and moves on to the next block. For more information, see Chapter 34, “Using Intelligent Calculation to Optimize Calculation.”

Data Block Renumbering

Hyperion Essbase renumbers the data blocks when you do any of the following:

- 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 Hyperion Essbase calculates the cells within each block depends on how you have configured the database. How you have configured the database defines the member calculation order of dense dimension members *within each block*. It also defines the calculation order of blocks that represent sparse dimension members.

Cell Calculation Order Examples

The following examples describe the cell calculation order for different database configurations.

Example 1

Consider the simplest case in which both of the following are true:

- No dimensions have time or accounts tags.
- The setting for aggregating `#MISSING` values is turned on. For more information, see "Calculating #MISSING Values" on page 33-35.

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. Hyperion Essbase calculates the shaded cells. The numbers in bold show the calculation order for these cells. The cell with multiple consolidation paths is darkly shaded.

Year->Market	New York	Massachusetts	East
Jan	112345.00	68754.00	**3**
Feb	135788.00	75643.00	**4**
Mar	112234.00	93456.00	**5**
Qtr1	**1**	**2**	**6**

As described in "Member Calculation Order" on page 28-3, Hyperion Essbase calculates dense dimensions in the order that they display in the database outline. Assuming that the Year dimension is displayed before the Market dimension in the database outline, the Year dimension is calculated before the Market dimension.

The cells are calculated in 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 Qtr1->New York and Qtr1->Massachusetts. When consolidated on Year, it is an aggregation of Jan->East, Feb->East, and Mar->East.

Hyperion Essbase knows that Qtr1->East has multiple consolidation paths. Therefore, it calculates Qtr1->East only once and uses the consolidation path of the dimension calculated last. In the above example, this dimension is Market.

The results are shown in 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, Hyperion Essbase ignores it when calculating Qtr1->East. If you place a member formula on East in the database outline, the formula is calculated when Hyperion 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 31, “Developing Calc Scripts.”

Example 2

Consider a second case in which both of the following are true:

- No dimensions have time or accounts tags.
- The setting for aggregating `#MISSING` values is turned off (the default). For more information, see Chapter 33, “Optimizing Calculations.”

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. Hyperion Essbase calculates the shaded cells. The numbers in bold show the calculation order for these cells. The cell with multiple consolidation paths is darkly shaded.

Year->Market	New York	Massachusetts	East
Jan	112345.00	68754.00	**4**
Feb	135788.00	75643.00	**5**
Mar	112234.00	93456.00	**6**
Qtr1	**1**	**2**	**3/7**

As described in “Member Calculation Order” on page 28-3, Hyperion Essbase calculates dense dimensions in the order they are defined in the database outline. Assuming the Year dimension is positioned before the Market dimension in the database outline, the Year dimension is calculated before the Market dimension.

The cells are calculated in 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 Qtr1->New York and Qtr1->Massachusetts. Second, it is calculated as an aggregation of Jan->East, Feb->East, and Mar->East.

The results are identical to the previous case. However, Qtr1->East has been calculated twice. This fact is significant when you need to load data at parent levels. For more information, see “Example 3” on page 28-18.

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 Hyperion Essbase consolidates Qtr1->East on the Market consolidation path. If you place a member formula on East in the database outline, the result is retained because the Market consolidation path is calculated last.

Example 3

Now consider the previous case in which both of the following are true:

- No dimensions have time or accounts tags.
- The setting for aggregating `#MISSING` values is turned off (the default). For more information, see “Calculating #MISSING Values” on page 33-35.
- 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 28-3, Hyperion Essbase calculates dense dimensions in the order that they are defined in the database outline. Assuming the Year dimension is positioned before the Market dimension in the database outline, the Year dimension is calculated before the Market dimension.

The cells are calculated in the same order as in Example 2. Qtr1->East is calculated on both the Year and Market consolidation paths.

Because the setting for aggregating `#MISSING` values is turned off, Hyperion Essbase does not aggregate the `#MISSING` values. Thus, the data that is loaded at parent levels is not overwritten by the `#MISSING` values below it.

However, if any of the child data values were not `#MISSING`, these values would be consolidated and would overwrite the parent values. For example, if Jan->New York contained 50000.00, this value would overwrite the values that were loaded at parent levels.

Hyperion 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**

Hyperion Essbase needs to calculate the Qtr1->East cell twice in order to ensure that a value is calculated for the cell. If it calculated Qtr1->East according to only the last consolidation path, then the result would be `#MISSING` for Qtr1->East, which is not the required result.

Example 4

Now consider a case in which all of the following are true:

- The Year dimension is tagged as time.
- The Measures dimension is tagged as accounts.
- The setting for aggregating `#MISSING` values is turned off (the default). For more information, see “Calculating #MISSING Values” on page 33-35.

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 29, “Dynamically Calculating Data Values.”

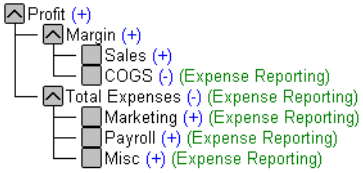


Figure 28-11: Measures Dimension in Sample Basic Database

Again, the following table shows a subset of the cells in a data block. Data values have been loaded into the input cells. Hyperion 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
Total Expenses	**2**	**5**	**8**	**11/16**
Profit	**3**	**6**	**9**	**12/17**

As described in "Member Calculation Order" on page 28-3, Hyperion Essbase calculates a dimension tagged as accounts first, followed by a dimension tagged as time. Therefore, in the above example, Measures is calculated before Year.

Three cells have multiple consolidation paths:

- Margin->Qtr1
- Total Expenses->Qtr1
- Profit->Qtr1

Because the setting for aggregating `#MISSING` values is turned off, Hyperion Essbase does not aggregate the `#MISSING` values. Thus, any data that is loaded at parent levels is not overwritten by the `#MISSING` values and Hyperion 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
Total Expenses				**28240**
Profit	**17378**	**17762**	**17803**	**52943**

From the calculation order, you can see that if you place a member formula on, for example, Margin in the database outline, its result is overwritten by the consolidation on Qtr1.

Cell Calculation Order for Formulas on a Dense Dimension

The cell calculation order within a data block is not affected by formulas on members. When Hyperion Essbase encounters a formula in a data block, it locks any other required data blocks, calculates the formula, and proceeds with the data block calculation.

When placing a formula on a dense dimension member, carefully consider the cell calculation order. As described in the examples above, the dimension calculated last overwrites previous cell calculations for cells with multiple consolidation paths. If required, you can use a calc script to change the order in which the dimensions are calculated. For more information, see Chapter 31, “Developing Calc Scripts.”

For more information on developing formulas, see Chapter 26, “Developing Formulas.”

Calculation Passes

Whenever possible, Hyperion Essbase calculates a database in one calculation pass through the database. Thus, it reads each of the required data blocks into memory only once, performing all relevant calculations on the data block and saving it. However, in some situations, Hyperion Essbase needs to perform more than one calculation pass through a database. On subsequent calculation passes, Hyperion Essbase brings data blocks back into memory, performs further calculations on them, and saves them again.

When you perform a default, full calculation of a database (CALC ALL), Hyperion Essbase attempts to calculate the database in one calculation pass. If you have dimensions that are tagged as accounts or time, Hyperion Essbase may have to do more than one calculation pass through the database.

The following table shows the number of calculation passes Hyperion Essbase performs if you have dimensions that are tagged as time or accounts, and you have at least one formula on the accounts dimension.

Dimension Tagged As:		Calculation Passes	During each calculation pass, Hyperion Essbase calculates based on:
Accounts	**Time**		
Dense or Sparse	None	1	All dimensions
Dense	Dense	1	All dimensions
Dense	Sparse	2	Pass 1: Accounts and time dimensions Pass 2: Other dimensions
Sparse	Sparse	2	Pass 1: Accounts and time dimensions Pass 2: Other dimensions
Sparse	Dense	2	Pass 1: Accounts dimension Pass 2: Other dimensions

If you are using formulas that are tagged as Two-Pass, Hyperion Essbase may need to do an *extra* calculation pass to calculate these formulas. For more information on using Two-Pass calculations, see Chapter 33, “Optimizing Calculations.”

When you use a calc script to calculate a database, the number of calculation passes Hyperion Essbase needs to perform depends upon the calc script. For more information, see “Calculation Passes” on page 28-23 and Chapter 34, “Using Intelligent Calculation to Optimize Calculation.” For more information on grouping formulas and calculations, see Chapter 33, “Optimizing Calculations.”

When you calculate a database, Hyperion Essbase automatically displays the calculation order of the dimensions for each pass through the database and tells you how many times Hyperion Essbase has cycled through the database during the calculation.

Hyperion Essbase displays this information in the ESSCMD window and in the Event Log file. To display the Event Log file, select Application > View Event Log from the Hyperion Essbase Application Manager menu.

For each data block, Hyperion Essbase decides whether to do a dense or a sparse calculation. The type of calculation it chooses depends on the type of values within the data block. When you run a default calculation (CALC ALL) on a database, each block is processed in order, according to its block number.

Hyperion 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 (if it is marked as *clean*), then Hyperion Essbase skips the block and moves on to the next block. For more information, see Chapter 34, “Using Intelligent Calculation to Optimize Calculation.”
- If the block needs recalculating, Hyperion Essbase checks to see if the block is a level 0, an input, or an upper level block. For definitions of level 0, input, and upper level blocks, see “Data Storage in Data Blocks” on page 28-1.
- If the block is a level 0 block or an input block, Hyperion Essbase performs a dense calculation on the block. Each cell in the block is calculated. For more information, see “Cell Calculation Order” on page 28-14.
- If the block is an upper level block, Hyperion 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. Hyperion 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 Hyperion Essbase chooses the Product dimension.

If the sparse member combination for the block has more than one parent member, Hyperion Essbase chooses the last dimension in the calculation order that includes a parent member. For example, if the block is for Product->East and you perform a default calculation on the Sample Basic database, Hyperion 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 28-3.

Based on the chosen sparse dimension, Hyperion 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, Hyperion Essbase performs a formula calculation on the sparse dimension. Hyperion Essbase evaluates each cell in the data block. The formula affects only the member on the sparse dimension, so overall calculation performance is not significantly affected.
- If the chosen sparse dimension is a default consolidation, Hyperion Essbase aggregates the values, taking the values of the previously calculated child data blocks.

Calculating Shared Members

Shared members are those that share data values with other members. For example, in the Sample Basic database, Diet Cola, Diet Root Beer, and Diet Cream are consolidated under two different parents. They are consolidated under Diet. They are also consolidated under their individual product types: Colas, Root Beer, and Cream Soda.

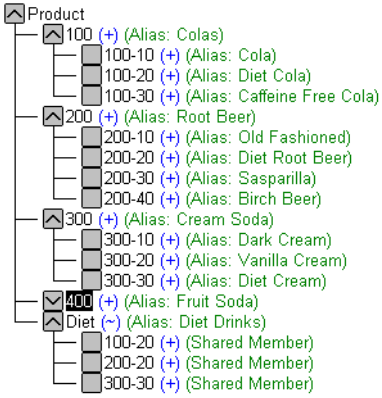


Figure 28-12: Calculating Shared Members

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 a database and the subset includes a shared member, Hyperion Essbase calculates the real member.

Chapter 29

Dynamically Calculating Data Values

This chapter explains how you calculate data values dynamically and how you benefit from doing so. Dynamically calculating some of the values in a database can significantly improve the performance of an overall database calculation.

The information in this chapter assumes that you are familiar with the concepts of member combinations, dense and sparse dimensions, and data blocks. For this information, see Part I, "Designing Hyperion Essbase Applications."

This chapter includes the following sections:

Understanding Dynamic Calculations

When you design your overall database calculation, it may be more efficient to calculate some member combinations when you retrieve their data, instead of pre-calculating the member combinations during the regular database calculation.

In Hyperion Essbase, you can define a member to have a *dynamic calculation*. This definition tells Hyperion Essbase to calculate a data value for the member "on-the-fly" as users request it. Dynamic calculation shortens regular database calculation time, but may increase retrieval time for the dynamically calculated data values. See "Reducing the Impact on Retrieval Time" on page 29-10 for more information.

In Hyperion Essbase you specify dynamic calculations on a per-member basis. You can define a member in your database outline as one of two types of a dynamically calculated member:

- Dynamic Calc
- Dynamic Calc And Store

Understanding Dynamic Calc Members

For a member that is tagged as Dynamic Calc, Hyperion Essbase does not calculate its data value during the regular database calculation; for example, during a CALC ALL. Instead, Hyperion Essbase calculates the data value upon retrieval; for example, when you retrieve the data into Hyperion Essbase Spreadsheet Add-in.

Specifically, Hyperion Essbase calculates a data value dynamically when you request the data value in either of two ways:

- By retrieving the data value into Hyperion Essbase Spreadsheet Add-in
- By running a report script that displays the data value

Hyperion Essbase does not store the calculated value; it recalculates the value for any subsequent retrieval.

Understanding Dynamic Calc And Store Members

Hyperion Essbase calculates the data value for a member that is tagged as Dynamic Calc And Store when you retrieve the data, in the same way as for a Dynamic Calc member. For a Dynamic Calc And Store member, however, Hyperion Essbase stores the data value that is calculated dynamically. Subsequent retrievals of that data value do not require recalculation, unless Hyperion Essbase detects that the value needs recalculating.

Recalculation of Data

When Hyperion 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.

Hyperion Essbase places the indicator on the data block containing the value and not on the data value itself. In other words, Hyperion Essbase tracks Dynamic Calc And Store members at the data block level. For more information on data blocks, see Chapter 4, “Basic Architectural Elements.”

If the data block needs recalculating, Hyperion Essbase detects the need and places an indicator on the data block when any of the following occur:

- You do a regular (batch) calculation.
- You restructure the database.
- You use the CLEARBLOCK DYNAMIC calculation command. For more information, see the online *Technical Reference* in the `DOCS` directory.

Hyperion Essbase recalculates the indicated data blocks when you next retrieve the data value.

Effect of Updated Values on Recalculation

Hyperion Essbase does *not* detect that a data block needs recalculating and does *not* place an indicator on the data block when you update the data; that is, updated blocks are recalculated only during the next batch calculation. Consider the following two scenarios:

- You do a data load.
- You do a Lock and Send from Hyperion Essbase Spreadsheet Add-in.

If you load data into the children of a Dynamic Calc And Store member, and the member is a consolidation of its child members, Hyperion Essbase does *not* know to recalculate the Dynamic Calc And Store member during the next retrieval. The parent member is recalculated only during the next batch calculation.

After loading data, you need to do a regular batch calculation of the database or use the CLEARBLOCK DYNAMIC calculation command to ensure that the Dynamic Calc And Store members are recalculated. For more information on CLEARBLOCK DYNAMIC, see “Clearing Data and Data Blocks” on page 29-20 and the online *Technical Reference* in the `DOCS` directory.

Retrieving the Parent Value of Dynamically Calculated Child Values

If you retrieve a parent value that is calculated from Dynamic Calc or Dynamic Calc And Store child members, Hyperion Essbase must dynamically calculate the child member combinations before calculating the parent value. Hyperion Essbase does not store the child values, even if they are Dynamic Calc And Store members.

For example, assume that Market is a parent member and that East and West are Dynamic Calc And Store child members that consolidate up to Market. When you retrieve a data value for Market, Hyperion Essbase calculates East and West, even though you have not specifically retrieved them. However, Hyperion Essbase does not store the values of East or West.

Benefitting from Dynamic Calculations

Dynamically calculating some database values can significantly improve the performance of an overall database calculation.

When you tell Hyperion Essbase to calculate some data values dynamically, you achieve the following advantages:

- Reduce the regular calculation time of the database because Hyperion Essbase has fewer member combinations to calculate.
- Reduce disk usage because Hyperion Essbase stores fewer calculated data values. Database size and index size are reduced. For more information on database and index sizes, see Chapter 40, "Introducing the Hyperion Essbase Kernel."
- Reduce database restructure time. For example, adding or deleting a Dynamic Calc member in a dense dimension does not change the data block size, and so Hyperion Essbase does not need to restructure the database for such additions and deletions. For more information, see "Restructuring a Database" on page 29-23.
- Reduce the time that is required to back up the database. Because database size is reduced, Hyperion Essbase takes less time to do a backup.

Data values that Hyperion 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 29-10.

Using Dynamic Calculations

You can tag any member as Dynamic Calc or Dynamic Calc And Store, except the following:

- Level 0 members that do not have a formula
- Label-Only members
- Shared members

Which members you choose to calculate dynamically depends on your database structure and on the balance between (1) your need for reduced calculation time and disk usage and (2) your need for speedy data retrieval for users. See "Choosing Which Values to Calculate Dynamically" on page 29-6.

Outline Editor in Hyperion Essbase Application Manager shows which members are Dynamic Calc and which members are Dynamic Calc And Store.

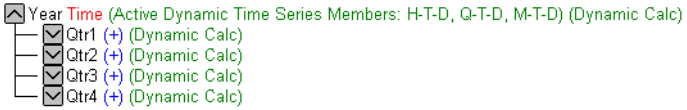


Figure 29-1: Sample Basic Outline Showing Dynamic Calc Members

In Hyperion Essbase Spreadsheet Add-in, users can display visual cues to distinguish dynamically calculated values. For more information, see the *Hyperion Essbase Spreadsheet Add-in User's Guide*.

Spreadsheet designers may want to develop spreadsheets that include dynamically calculated values in data-less mode, so that Hyperion Essbase does not dynamically calculate values while spreadsheets are being built.

Choosing Which Values to Calculate Dynamically

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

- 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 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.

Tagging 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 Hyperion Essbase to perform an expensive calculation. Formulas containing, for example, financial functions or cross-dimensional operators (->) are complex formulas.

Tagging Sparse Dimension Members

Consider making the following changes to 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 Hyperion Essbase to perform an expensive calculation. For example, any formula that contains a financial function is a complex formula. For more information, see “Are you using a complex formula?” on page 33-7.
- Tag upper level, members in a dimension that you frequently restructure as Dynamic Calc or Dynamic Calc And Store.
- Do *not* tag upper level, sparse dimension members that have 20 or more descendants as Dynamic Calc or Dynamic Calc And Store.

For more information, see “Choosing Between Dynamic Calc and Dynamic Calc And Store” on page 29-13.

Tagging 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.

Calc Scripts and Dynamic Calculations

When Hyperion 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 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
```

Formulas and Dynamically Calculated Members

You *can* include a dynamically calculated member in a formula when you apply the formula to the database outline. For example, if Qtr1 is a Dynamic Calc member, you could place the following formula on Qtr1 in the database outline:

```
Qtr1 = Jan + Feb;
```

You *cannot* make a dynamically calculated member the target of a formula calculation in a calc script; Hyperion 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, Hyperion 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;
```

This formula is valid because Hyperion Essbase is not assigning a value to the dynamically calculated member.

Note: When you reference a dynamically calculated member in a formula in the database outline or in a calc script, Hyperion Essbase interrupts the regular calculation to do the dynamic calculation. This interruption can significantly lower calculation performance.

Dynamically Calculated Children of Regular Members

If the calculation of a regular member depends on the calculation of Dynamic Calc or Dynamic Calc And Store child members, then Hyperion Essbase has to calculate the child members during the regular database calculation in order to calculate the parent. Therefore, there is no reduction in regular calculation time. This requirement applies to sparse dimension and dense dimension members.

For example, consider the following outline:

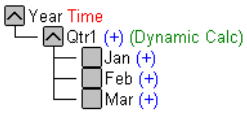


Figure 29-2: Sample Basic Outline Showing Qtr1 as a Dynamic Calc Member

Jan, Feb, and Mar are regular members, Qtr1 is a Dynamic Calc member, and Year is a regular member. When Hyperion Essbase calculates Year during the regular database calculation, it has to calculate Qtr1 in order to consolidate it up to Year. Hyperion 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 a dense dimension member is not significant unless the member contains a complex formula. See “Choosing Which Values to Calculate Dynamically” on page 29-6.

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, Hyperion Essbase calculates a *retrieval factor* for the database outline when you save the outline. Hyperion Essbase calculates this retrieval factor based on the dynamically calculated data block that is the most expensive for Hyperion Essbase to calculate. The retrieval factor is the number of data blocks that Hyperion Essbase has to retrieve from disk or from the database in order to calculate the most expensive block. If the database has only dense dimension Dynamic Calc or Dynamic Calc And Store members (no Dynamic Calc or Dynamic Calc And Store members in 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 may reduce retrieval time, because the database size and index size are reduced.

Displaying a Retrieval Factor

When you add Dynamic Calc or Dynamic Calc And Store members to your database outline and save the outline, Hyperion Essbase calculates an estimated retrieval factor for the most expensive dynamically calculated data block. Hyperion Essbase displays the value in the application event log file.

➤ To view an estimated retrieval factor in Hyperion Essbase Application Manager:

1. In the application desktop window, select an application and database.
2. Select Application > View Event Log.
3. In the View Log File dialog box, select Display All or Date.
4. Click OK.

   Hyperion 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 29-3: Retrieval Factor in the Application Event Log

This message tells you that Hyperion Essbase needs to retrieve one 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 a database outline and save the outline, Hyperion Essbase provides a summary of how many members are tagged as Dynamic Calc and Dynamic Calc And Store. Hyperion Essbase displays the summary in the application event log file.

➤ To view a summary of dynamically calculated members in Hyperion Essbase Application Manager:

1. In the application desktop window, select an application and database.
2. Select Application > View Event Log.
3. In the View Log File dialog box, select Display All or Date.
4. Click OK.

   Hyperion Essbase displays the Log Viewer window for the database you selected.

A message similar to the following displays:

```
[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 29-4: Event Log Summary of Dynamic Calc and Dynamic Calc And Store Members

This message tells you that there are two Dynamic Calc members in the first dimension in the database outline and four in the fourth dimension in the database outline. There are no Dynamic Calc And Store members.

Hyperion Essbase includes Dynamic Time Series members in these numbers. For more information, see Chapter 30, “Calculating Time Series Data.”

Increasing the Retrieval Buffer Size

When you retrieve data into Hyperion Essbase Spreadsheet Add-in or use the Report Writer to retrieve data, Hyperion Essbase uses the retrieval buffer to optimize the retrieval. Hyperion Essbase processes the data in sections. Increasing the retrieval buffer size can significantly reduce retrieval time because Hyperion Essbase can process larger sections of data at one time.

By default, the retrieval buffer size 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 Hyperion Essbase Application Manager:

1. In the application desktop window, select an application and database.
2. Select Database > Settings.

   Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

For more information on this setting, see Chapter 38, "Optimizing Your Reports."

Choosing Between Dynamic Calc and Dynamic Calc And Store

In most cases you can optimize calculation performance and lower disk usage by using Dynamic Calc members instead of Dynamic Calc And Store members. However, in specific situations, using Dynamic Calc And Store members is optimal.

Recommendations for Sparse Dimension Members

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 Hyperion Essbase calculates data values for a member combination that includes a Dynamic Calc member, Hyperion Essbase calculates only the requested values of the relevant data block. These values can be a subset of the data block.

However, when Hyperion Essbase calculates data values for a member combination that includes a Dynamic Calc And Store member, Hyperion Essbase needs to calculate and store the whole data block, even if the requested data values are a subset of the data block. Thus, the calculation takes longer and the initial retrieval time is greater.

Hyperion Essbase stores only the data blocks that contain the requested data values. If Hyperion 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, Hyperion Essbase calculates and stores the Market->Cola data block. In order to calculate and store Market->Cola, Hyperion Essbase calculates the intermediate data blocks: East->Cola, West->Cola, South->Cola, and Central->Cola. Hyperion Essbase does not store these intermediate data blocks.

Recommendations for Members with Specific Characteristics

Using Dynamic Calc And Store may slow initial retrieval; however, subsequent retrievals are faster than for Dynamic Calc members. Use Dynamic Calc And Store instead of Dynamic Calc for the following members:

- An upper-level sparse dimension member with children on a remote database. Hyperion Essbase needs to retrieve the value from the remote database, which increases retrieval time. For more information on partitions, see “Dynamically Calculating Data in Partitions” on page 29-25.
- A sparse dimension member with a complex formula. A complex formula requires Hyperion Essbase to perform an expensive calculation. For example, any formula that contains a financial function or a cross-dimensional member is a complex formula.
- If users frequently retrieve an upper level, sparse dimension member, speedy retrieval time is very important.

For example, in the Sample Basic database, if most users retrieve data at the Market level, you probably want to tag Market as Dynamic Calc And Store and its children as Dynamic Calc.

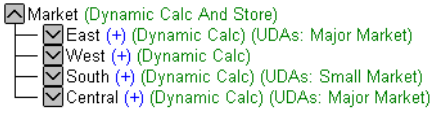


Figure 29-5: Sample Basic Outline Showing Market as a Dynamic Calc And Store Member

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 retrieval time and in regular calculation time. In addition, database size (disk usage) does not decrease significantly because Hyperion Essbase reserves space for the member's data values in the data block.

Recommendations for Data with Many Concurrent Users

Use Dynamic Calc members for data with concurrent users. If many users are concurrently retrieving Hyperion Essbase data, the initial retrieval time for Dynamic Calc And Store members can be significantly higher than for Dynamic Calc members.

Dynamic Calc And Store member retrieval time increases as the number of concurrent user retrievals increases. However, Dynamic Calc member retrieval time does not increase as concurrent user retrievals increase.

If many users are concurrently accessing data, you may see significantly lower retrieval times if you use Dynamic Calc members instead of Dynamic Calc And Store members.

Considering the Effects of Dynamic Calculations

Using dynamically calculated data values changes the order in which Hyperion Essbase calculates the values and can have implications for the way you administer a database.

Calculation Order for Dynamic Calculations

When Hyperion Essbase dynamically calculates data values, it calculates the data in an order that is different from the regular database calculation order. For detailed information on calculation order, see Chapter 28, “Defining the Calculation Order.”

During regular calculations, Hyperion 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, Hyperion 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, Hyperion Essbase bases the sparse calculation on the time dimension.
    b. Otherwise, Hyperion Essbase uses the dimension that it uses for a regular calculation. For more information, see Chapter 28, “Defining the Calculation Order.”
2. Dense dimensions:
    a. Dimension tagged as accounts, if dense
    b. Dimension tagged as time, if dense
    c. Time series calculations
    d. Remaining dense dimensions
    e. Two-pass calculations

Calculation Order for Dynamically Calculating Two-Pass Members

Consider the following information to ensure that Hyperion Essbase produces the required calculation result when it dynamically calculates data values for members that are tagged as two-pass. For more information on two-pass calculations, see Chapter 33, “Optimizing Calculations.”

If more than one Dynamic Calc or Dynamic Calc And Store dense dimension member is tagged as two-pass, Hyperion Essbase performs the regular dynamic calculation and then calculates the two-pass members in the following order:

1. Two-pass members in the accounts dimension, if any exist.
2. Two-pass members in the time dimension, if any exist.
3. Two-pass members in the remaining dense dimensions in the order in which the dimensions appear in the outline.

For example, in the Sample Basic database, assume the following:

- Margin % in the dense Measures dimension (the dimension tagged as accounts) is tagged as both Dynamic Calc and two-pass.
- Variance in the dense Scenario dimension is tagged as both Dynamic Calc and two-pass.

Hyperion Essbase calculates the accounts dimension member first. So, Hyperion Essbase calculates Margin % (from the Measures dimension) and then calculates Variance (from the Scenario dimension).

If Scenario is a sparse dimension, Hyperion Essbase calculates Variance first, following the regular calculation order for dynamic calculations. See “Calculation Order for Dynamic Calculations” on page 29-16. Hyperion Essbase then calculates Margin %.

This calculation order does not produce the required result because Hyperion Essbase needs to calculate Margin %->Variance using the formula on Margin %, and not the formula on Variance. You can avoid this problem by making Scenario a dense dimension. This problem does not occur if the Measures dimension (the accounts dimension) is sparse, because Hyperion Essbase then still calculates Margin % first.

Calculation Order for Asymmetric Data

Because the calculation order used for dynamic calculations differs from the calculation order used for regular database calculations, in some database outlines you may get different calculation results if you tag certain members as Dynamic Calc or Dynamic Calc And Store. These differences happen when Hyperion Essbase dynamically calculates asymmetric data.

Symmetric data calculations produce the same results no matter which dimension is calculated. Asymmetric data calculations calculate differently along different dimensions.

Table 29-1: 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 calculation for Qtr1->Profit produces the same result whether you calculate along the dimension tagged as time or the dimension tagged as accounts. Calculating along the time dimension, you add the values for Jan, Feb, and Mar: 50+100+150=300. Calculating along the accounts dimension, you subtract Qtr1->COGS from Qtr1->Sales: 600–300=300.

Table 29-2: Example of an Asymmetric Calculation

Market->Accounts	New York	Florida	Connecticut	East
UnitsSold	10	20	20	50
Price	5	5	5	15
Sales (Price * UnitsSold)	50	100	100	250

The calculation for East->Sales produces the correct result when you calculate along the Market dimension, but produces an incorrect result when you calculate along the accounts dimension. Calculating along the Market dimension, 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->UnitsSold: 15 * 50=750, which is incorrect.

In the following outline, East is a sparse dimension, and Accounts is a dense dimension:

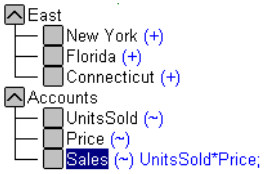


Figure 29-6: Outline Showing Calculation Order of Dynamic Calculations

If East and Sales are tagged as Dynamic Calc, then Hyperion 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, Hyperion Essbase produces the correct result by calculating as follows:

1. The dense Accounts dimension, calculating the values for UnitsSold, Price, and Sales for New York, Florida, and Connecticut.
2. The sparse East dimension, by aggregating the calculated values for UnitsSold, Price, and Sales for New York, Florida, and Connecticut to obtain the Sales values for East.

If East and Sales are Dynamic Calc members, Hyperion Essbase produces an incorrect result by calculating as follows:

1. The sparse East dimension, by aggregating the values for UnitsSold, Price, and Sales for New York, Florida, and Connecticut to obtain the values for East.
2. The values for East->Sales, by taking the aggregated values in the East data blocks and performing a formula calculation with these values to obtain the value for Sales.

To avoid this problem and ensure that you obtain the required results, do not tag the Sales member as Dynamic Calc or Dynamic Calc And Store.

Using Dynamic Calculations with Standard Procedures

The following sections discuss the effects of using dynamic calculations with standard Hyperion Essbase procedures.

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 Hyperion Essbase knows to recalculate the blocks. The CLEARDATA command has no effect on data values for Dynamic Calc members.

Copying Data

You cannot copy data to a dynamically calculated data value. You cannot specify a Dynamic Calc or Dynamic Calc And Store member as the target for the DATACOPY calculation command.

Converting Currencies

You cannot specify a Dynamic Calc or Dynamic Calc And Store member as the target for the CCONV command.

Loading Data

When you load data, Hyperion Essbase does not load data into member combinations that contain a Dynamic Calc or Dynamic Calc And Store member. Hyperion Essbase skips these members during data load. Hyperion Essbase does not display an error message.

To place data into Dynamic Calc and Dynamic Calc And Store members, after loading data, you need to ensure that Hyperion Essbase recalculates Dynamic Calc And Store members. To do this, see "Effect of Updated Values on Recalculation" on page 29-4.

Exporting Data

Hyperion Essbase does not calculate dynamically calculated values before exporting data. Hyperion Essbase does not export values for Dynamic Calc members. Hyperion 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

Hyperion 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 the `DOCS` directory.

Including Dynamic Calc and Dynamic Calc And Store Members in Calc Scripts

When calculating a database, Hyperion Essbase skips the calculation of any Dynamic Calc or Dynamic Calc And Store members. Hyperion 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 "Calc Scripts and Dynamic Calculations" on page 29-8.

Creating Dynamic Calc and Dynamic Calc And Store Members

In Hyperion Essbase Application Manager Outline Editor, you can tag members as Dynamic Calc or as Dynamic Calc And Store. When you build a dimension, you can use properties to indicate Dynamic Calc and Dynamic Calc And Store members.

➤ To tag a member as Dynamic Calc using Hyperion Essbase Application Manager:

1. In Outline Editor, open the database.
2. Select the member that you want to tag as Dynamic Calc.
3. Click the  button.

   Alternatively, in the Outline Editor menu, select Edit > Properties. Then, in the Data Storage group of the Member Specification dialog box, select Dynamic Calc.

   Hyperion Essbase labels the member as Dynamic Calc.

➤ To tag a member as Dynamic Calc And Store using Hyperion Essbase Application Manager:

1. In Outline Editor, open the database.
2. Select the member that you want to tag as Dynamic Calc And Store.
3. Click the  button.

   Alternatively, in the Outline Editor menu, select Edit > Properties. Then, in the Data Storage group of the Member Specification dialog box, select Dynamic Calc And Store.

   Hyperion Essbase labels the member as Dynamic Calc And Store.

➤ To remove a Dynamic Calc or Dynamic Calc And Store tag using Hyperion Essbase Application Manager:

1. In Outline Editor, open the database.
2. Select the member from which you want to remove the Dynamic Calc or Dynamic Calc And Store tag.
3. Click the  button.

   Alternatively, in the Outline Editor menu, select Edit > Properties. Then, in the Data Storage group of the Member Specification dialog box, select Store Data.

Building Dimensions with Dynamic Calc Members

You can build dimensions with Dynamic Calc and Dynamic Calc And Store members. In the dimension build data file, use the property X for Dynamic Calc and the property V for Dynamic Calc And Store. For more information, see “Setting Member Properties” on page 14-24

Restructuring a Database

When you add a Dynamic Calc member to a dense dimension, Hyperion Essbase does not reserve space in the data block for the member’s values. Therefore, Hyperion Essbase does not need to restructure the database. However, when you add a Dynamic Calc And Store member to a dense dimension, Hyperion Essbase does reserve space in the relevant data blocks for the member’s values and therefore needs to restructure the database.

When you add a Dynamic Calc or a Dynamic Calc And Store member to a sparse dimension, Hyperion Essbase updates the index, but does not change the relevant data blocks. For more information on the database index, see Chapter 40, “Introducing the Hyperion Essbase Kernel.”

Hyperion Essbase can save changes to your database outline significantly faster if it does not have to restructure the database.

In the following cases, Hyperion Essbase does not restructure the database. Hyperion Essbase only has to save the database outline, which is very fast.

Hyperion Essbase does *not* restructure the database or change the index when you do any of the following:

- Add, delete, or move a dense dimension Dynamic Calc member. (But Hyperion 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, Hyperion Essbase does not restructure the database, but does have to restructure the database index. Restructuring the index is significantly faster than restructuring the database.

Hyperion Essbase restructures only the database index when you do either of the following:

- Add, delete, or move sparse dimension Dynamic Calc or Dynamic Calc And Store members
- Change a regular dense dimension member to a Dynamic Calc And Store member

However, Hyperion Essbase does restructure your database when you do any of the following:

- Add, delete, or move a dense dimension Dynamic Calc And Store member. (But Hyperion Essbase does *not* restructure the database if the member is Dynamic Calc.)
- Change a dense dimension Dynamic Calc And Store member to a Dynamic Calc member
- Change a dense dimension Dynamic Calc member to a Dynamic Calc And Store member
- Change a dense dimension regular member to a Dynamic Calc member
- Change a dense dimension Dynamic Calc member to a regular member
- Change a sparse dimension regular member to a Dynamic Calc or Dynamic Calc And Store member

For detailed information on the types of database restructuring, see Chapter 40, "Introducing the Hyperion Essbase Kernel."

Dynamically Calculating Data in Partitions

You can define Dynamic Calc and Dynamic Calc And Store members in transparent, replicated, or linked regions of your partitions. For more information on partitions, see Chapter 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. Because Hyperion Essbase needs to retrieve the child values from the other database, retrieval time is increased. You could use Dynamic Calc instead of Dynamic Calc And Store; however, the impact on subsequent retrieval time might be too great.

For example, assume that your local database is the Corporate database, which has transparent partitions to the regional data for East, West, South, and Central. You could tag the parent member Market as Dynamic Calc And Store.

In a transparent partition, the definition on the remote database takes precedence over any definition on the local database. For example, if a member is tagged as Dynamic Calc in the local database but not in the remote database, Hyperion Essbase retrieves the value from the remote database and does not do the local calculation.

If you are using a replicated partition, then you might want to use Dynamic Calc members instead of Dynamic Calc And Store members. When calculating replicated data, Hyperion Essbase does not retrieve the child blocks from the remote database, and therefore the impact on retrieval time is not great.

Note: When Hyperion 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, Hyperion Essbase replicates the data in the data block. However, for dynamically calculated data, data blocks and time stamps do not exist. Therefore Hyperion Essbase always replicates dynamically calculated data.

Chapter 30

Calculating Time Series Data

This chapter explains how to calculate time series data. For example, you can do inventory tracking by calculating the first and last values for a specific time period. You can also calculate period-to-date values.

This chapter includes the following sections:

Calculating First, Last, or Average Values

Using time balance and variance reporting tags on the dimension tagged as accounts, you can tell Hyperion Essbase how to perform time balance calculations on accounts data.

Hyperion Essbase usually calculates a parent in the dimension tagged as time by consolidating or calculating the formulas on the parent's children. However, you can use accounts tags, such as time balance and variance reporting tags, to consolidate a different kind of value. For example, if you tag a parent member in the accounts dimension with a time balance property of First, Hyperion Essbase calculates the member by consolidating the value of the member's first child. For example, in the Sample Basic database, the Opening Inventory member in the Measures dimension (the accounts dimension) has a time balance property of First. This member represents the inventory at the beginning of the time period. If the time period is Qtr1, Opening Inventory represents the inventory available at the beginning of Jan (the first member in the Qtr1 branch).

To use accounts tags, you must have a dimension tagged as accounts and a dimension tagged as time. You use the First, Last, and Average tags (time balance properties) and the Expense tag (variance reporting property) only on members of a dimension tagged as accounts. The dimensions you tag as time and accounts can be either dense or sparse dimensions.

Note: If you are using Intelligent Calculation, changing accounts tags in the database outline does not cause Hyperion Essbase to restructure the database. You may have to tell Hyperion Essbase explicitly to recalculate the required data values. See Chapter 34, “Using Intelligent Calculation to Optimize Calculation.”

Specifying Accounts and Time Dimensions

When you tag a dimension as accounts, Hyperion Essbase knows that the dimension contains members with accounts tags. When you tag a dimension as time, Hyperion Essbase knows that this dimension is the one on which to base the time periods for the accounts tags.

In the Sample Basic database, the Measures dimension is tagged as accounts, and the Year dimension is tagged as time.

Figure 30-1: Sample Basic Outline Showing Accounts and Time Tags

For information on tagging accounts and time dimensions, see Chapter 9, “Setting Dimension and Member Properties.”

Reporting the Last Value for Each Time Period

For an accounts dimension member, you can tell Hyperion Essbase to move the last value for each time period up to the next level. To report the last value for each time period, set the member's time balance property as Last. (In the database outline, the TB Last tag is displayed.)

For example, in the Sample Basic database, the accounts member Ending Inventory consolidates the value for the last month in each quarter and uses that value for that month's parent. For example, the value for Qtr1 is the same as the value for Mar.

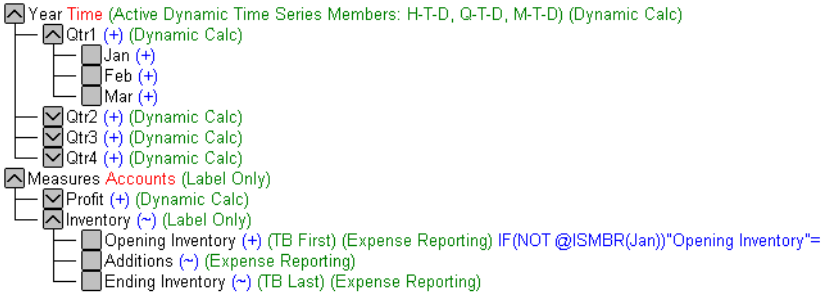


Figure 30-2: Sample Basic Outline Showing Last Tag

For information on tagging an accounts member as Last, see Chapter 9, "Setting Dimension and Member Properties."

By default, Hyperion Essbase does not skip `#MISSING` or zero (0) values when calculating a parent value. You can choose to skip these values. See "Skipping #MISSING and Zero Values" on page 30-5.

Reporting the First Value for Each Time Period

For an accounts dimension member, you can tell Hyperion Essbase to move the first value for each time period up to the next level. To report the first value for each time period, set the member's time balance property as First. (The tag displays as TB First in the database outline.)

For example, in the Sample Basic database, the accounts member Opening Inventory consolidates the value of the first month in each quarter and uses that value for that month's parent. For example, the value for Qtr1 is the same as the value for Jan.

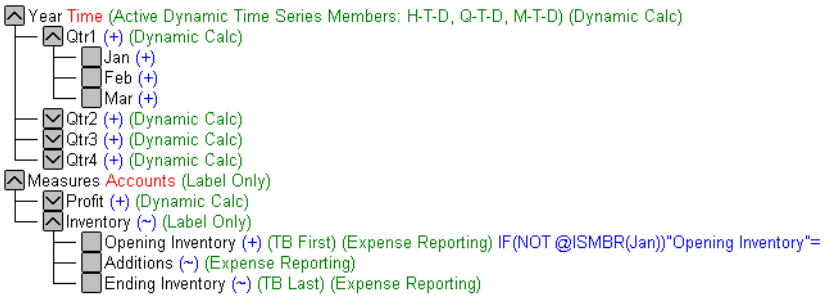


Figure 30-3: Sample Basic Outline Showing First Tag

For information on tagging an accounts member as First, see Chapter 9, "Setting Dimension and Member Properties."

By default, Hyperion Essbase does not skip `#MISSING` or zero (0) values when calculating a parent value. You can choose to skip these values. See "Skipping #MISSING and Zero Values" on page 30-5.

Reporting the Average Value for Each Time Period

For an accounts dimension member, you can tell Hyperion Essbase to average values across time periods and consolidate the average up to the next level. For example, you can tell Hyperion Essbase to average the values for Jan, Feb, and Mar and then use that value for the Qtr1 value. To report the average value for each time period, set the member's time balance property as Average.

For information on tagging an accounts member as Average, see Chapter 9, “Setting Dimension and Member Properties.”

By default, Hyperion Essbase does not skip #MISSING or zero (0) values when calculating a parent value. When calculating the average, Hyperion Essbase aggregates the child values and divides by the number of children, regardless of whether the children have #MISSING or zero values. You can tell Hyperion Essbase to skip #MISSING and zero values. See “Skipping #MISSING and Zero Values” on page 30-5.

Skipping #MISSING and Zero Values

You can tell Hyperion Essbase how to treat #MISSING and zero (0) values when doing time balance calculations. A #MISSING value is a marker in Hyperion Essbase that indicates that the data in this location does not exist, does not contain any meaningful value, or was never entered.

By default, Hyperion Essbase does not skip #MISSING or 0 (zero) values when calculating a parent value.

You can override this default using the following tags:

To...	Use this Tag...
Skip #MISSING values when calculating a parent value	**Skip Missing** In Outline Editor, select the parent member from the accounts dimension and click the  button.
Skip 0 values when calculating a parent value	**Skip Zeros** In Outline Editor, select the parent member from the accounts dimension and click the  button.
Skip #MISSING values and 0 values when calculating a parent value	**Skip Missing and Skip Zeros** In Outline Editor, select the parent member from the accounts dimension and click the  button.

For example, if you tag an accounts dimension member as Last and Skip Missing, then Hyperion Essbase consolidates the last non-missing child to the parent. Consider the following example:

Accounts->Time	Jan	Feb	Mar	Qtr1
Accounts Member (**Last, Skip Missing**)	60	70	#MI	70

Considering the Effects of First, Last, and Average Tags

The following table shows how Hyperion Essbase consolidates the time dimension based on the time balance (TB) First, Last, and Average tags on accounts dimension members.

Accounts->Time	Jan	Feb	Mar	Qtr1	
Accounts Member1	11	12	13	36	Value of Jan+Feb+Mar
Accounts Member2 (**TB First**)	20	25	21	20	Value of Jan
Accounts Member3 (**TB Last**)	25	21	30	30	Value of Mar
Accounts Member4 (**TB Average**)	20	30	28	26	Average of Jan, Feb, Mar

Placing Formulas on Time and Accounts Dimensions

If you place a member formula on a time or accounts dimension, it may be overwritten by a time balance calculation.

Consider the following example from Sample Basic, in which Opening Inventory is tagged as First:

Measures->Year	Jan	Feb	Mar	Qtr1
Opening Inventory: **First**	30000	28000	27000	30000

Because Opening Inventory is tagged as First, Hyperion Essbase calculates Opening Inventory for Qtr1 by taking the Opening Inventory for Jan value. Any member formula that is placed on Qtr1 in the database outline is overwritten by this time balance calculation.

Calculating Period-to-Date Values

You can calculate period-to-date values for your data. For example, you can calculate the sales values for the current quarter up to the current month. If the current month is May, using a standard calendar quarter, the quarter total is the total of the values for April and May.

In Hyperion Essbase, you can calculate period-to-date values in two ways:

- During a batch calculation, using the @PTD function
- Dynamically, when a user requests the values, using Dynamic Time Series members

This section explains how to use Dynamic Time Series members to dynamically calculate period-to-date values. Using Dynamic Time Series members is the most efficient method in almost all cases. For an example using the @PTD function to calculate period-to-date values, see Chapter 27, “Examples of Formulas.”

Using Dynamic Time Series Members

In order to calculate period-to-date values dynamically, you need to use a Dynamic Time Series member for a period on the dimension tagged as time. See “Specifying Accounts and Time Dimensions” on page 30-2.

You do not create the Dynamic Time Series member directly in the database outline. Instead, you enable a predefined Dynamic Time Series member and associate it with an appropriate generation number. This procedure creates a Dynamic Time Series member for you.

For example, if you want to calculate quarter-to-date values, you enable the Q-T-D member and associate it with the generation to which you want to apply the Dynamic Time Series member. In Sample Basic, this is generation number 2, which contains the Qtr1, Qtr2, Qtr3, and Qtr4 members. Hyperion 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 30-9.

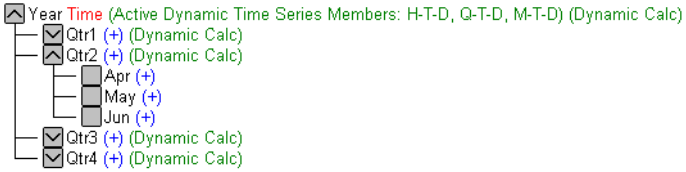


Figure 30-4: Sample Basic Outline Showing Time Dimension

Dynamic Time Series members are not displayed as members in the database outline. Instead, Hyperion Essbase lists the currently active Dynamic Time Series members in a comment on the time dimension. In the following outline, H-T-D (history-to-date), Q-T-D (quarter-to-date), and M-T-D (month-to-date) are active. H-T-D is associated with generation 1; Q-T-D and M-T-D are associated with generation 2.

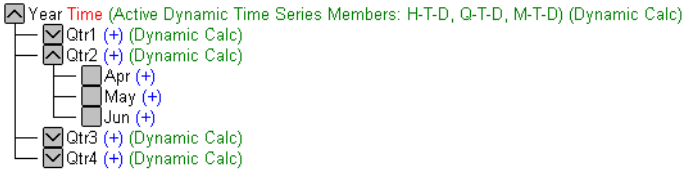


Figure 30-5: Sample Basic Outline Showing Dynamic Time Series

Hyperion Essbase provides eight predefined Dynamic Time Series members:

- H-T-D History-to-date
- Y-T-D Year-to-date
- S-T-D Season-to-date
- P-T-D Period-to-date

- Q-T-D Quarter-to-date
- M-T-D Month-to-date
- W-T-D Week-to-date
- D-T-D Day-to-date

These eight members provide up to eight levels of period-to-date reporting. How many members you use and which members you use depends on your data and your database outline.

For example, if your database contains hourly, daily, weekly, monthly, quarterly, and yearly data, you might want to report day-to date (D-T-D), week-to-date (W-T-D), month-to-date (M-T-D), quarter-to-date (Q-T-D), and year-to-date (Y-T-D) information.

If your database contains monthly data for the past 5 years, you might want to report year-to-date (Y-T-D) and history-to-date (H-T-D) information, up to a specific year.

If your database tracks data for seasonal time periods, you might want to report period-to-date (P-T-D) or season-to-date (S-T-D) information.

You can associate a Dynamic Time Series member with any generation in the time dimension except the highest generation number, irrespective of the data. For example, if you choose, you can use the P-T-D member to report quarter-to-date information. You cannot associate Dynamic Time Series members with level 0 members of the time dimension.

Enabling Dynamic Time Series Members

To use Dynamic Time Series members, you need to enable them. If required, you can specify aliases for Dynamic Time Series members. See “Specifying Alias Names for Dynamic Time Series Members” on page 30-13.

➤ To enable Dynamic Time Series members using Hyperion Essbase Application Manager:

1. In Outline Editor, open the database.
2. Select Outline > Dynamic Time Series.

   Hyperion Essbase displays the Dynamic Time Series Member Information dialog box.

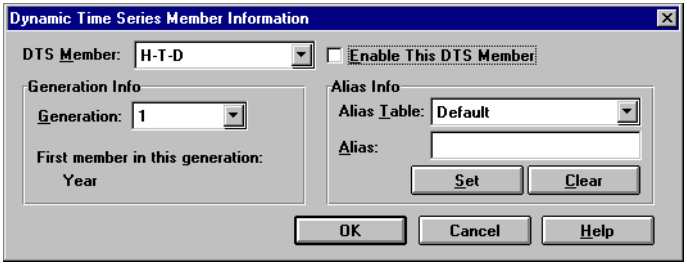


Figure 30-6: Dynamic Time Series Member Information Dialog Box for Sample Basic

3. In the DTS Member list, select the required Dynamic Time Series member. For example, select Q-T-D (quarter-to-date).
4. In the Generation list, select the appropriate generation number. For example, in the Sample Basic database, select generation 2 to associate the Q-T-D member with that generation.

   Note: How many generations the Generation list displays depends on how many generations are in the time dimension. You cannot associate Dynamic Time Series members with the highest generation (level 0 members).

5. Check Enable This DTS Member.

   In Sample Basic, the DTS member may already be enabled by default.

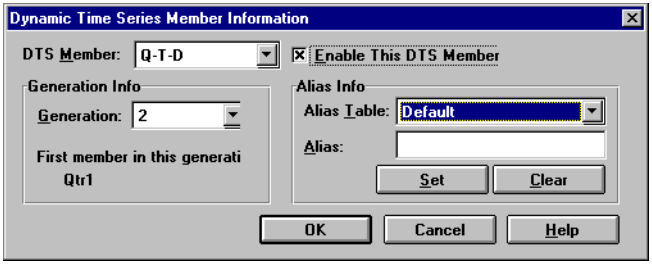


Figure 30-7: Dynamic Time Series Member Information Dialog Box Showing Q-T-D on Sample Basic

6. If desired, you can create an alias name for the DTS member. See "Specifying Alias Names for Dynamic Time Series Members" on page 30-13.
7. To enable the Dynamic Time Series member, click OK.

   In the database outline, Hyperion Essbase adds a comment to the dimension tagged as time. The following figure shows the Sample Basic database with H-T-D, Q-T-D, and M-T-D defined.

Figure 30-8: Sample Basic Outline Showing Dynamic Time Series Members

Disabling Dynamic Time Series Members

To disable a Dynamic Time Series member, you tell Hyperion Essbase not to use the predefined member.

➤ To disable a Dynamic Time Series member using Hyperion Essbase Application Manager:

1. In Outline Editor, open the database.
2. Select Outline > Dynamic Time Series.

   Hyperion Essbase displays the Dynamic Time Series Member Information dialog box.

3. In the DTS Member list, select the Dynamic Time Series member you want to disable.

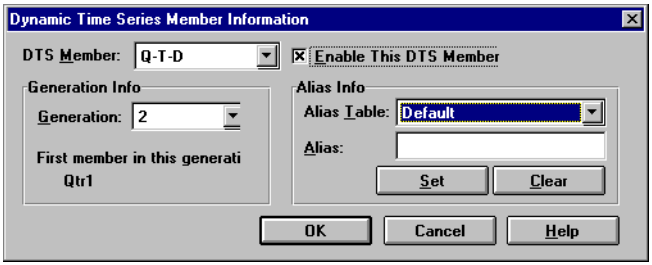


Figure 30-9: Dynamic Time Series Member Information Dialog Box Showing Enabled Q-T-D

4. Clear Enable This DTS Member.

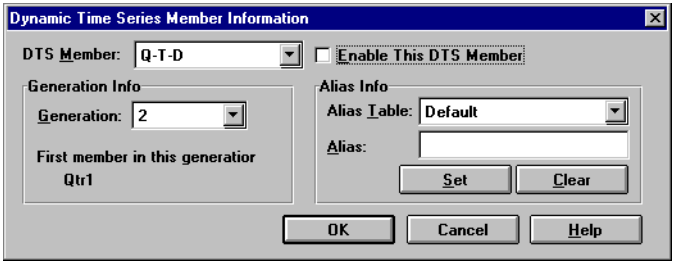


Figure 30-10: Dynamic Time Series Member Information Dialog Box Showing Q-T-D Disabled

5. Click OK.

Specifying Alias Names for Dynamic Time Series Members

You can specify alias names for predefined Dynamic Time Series members. You can then use the alias names to retrieve the Dynamic Time Series members in Hyperion Essbase Spreadsheet Add-in or in a report.

You can create up to eight alias names for each Dynamic Time Series member. Hyperion Essbase saves each alias name in the Dynamic Time Series alias table that you specify.

➤ To specify an alias name for a Dynamic Time Series member:

1. In the Dynamic Time Series Member Information dialog box, enable the DTS member. See “Enabling Dynamic Time Series Members” on page 30-9.
2. In the Alias Table list, select the alias table in which you want to save the alias name.
3. In the Alias text box, type the alias name for the DTS member.

   For example, type **`QtrToDate`**.

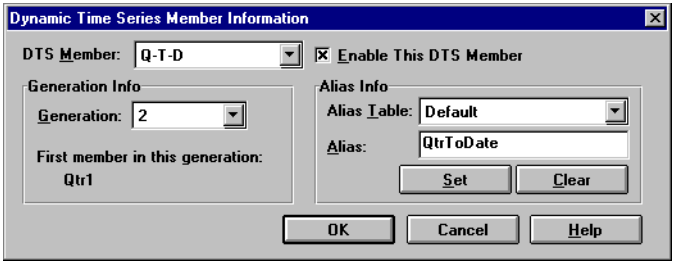


Figure 30-11: Dynamic Time Series Member Information Dialog Box Showing Q-T-D Alias

4. Click Set.
5. To save the alias name to the chosen alias table, click OK.

For more information on specifying and displaying alias names, see Chapter 8, “Creating and Changing Database Outlines.”

Applying Predefined Generation Names to Dynamic Time Series Members

When you enable a Dynamic Time Series member and associate it with a generation number, Hyperion Essbase creates a predefined generation name for that generation number. These generation names display in the Generation and Level Names dialog box. To open this dialog box, select Outline > Gen/Level Names in Hyperion Essbase 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 their corresponding generation names:

Member	Generation Name	Member	Generation
H-T-D	History	Q-T-D	Quarter
Y-T-D	Year	M-T-D	Month
S-T-D	Season	W-T-D	Week
P-T-D	Period	D-T-D	Day

These member and generation names are reserved for use by Hyperion Essbase. If you use one of these generation names to create a generation name on the time dimension, Hyperion Essbase automatically creates and enables the corresponding Dynamic Time Series member for you.

For example, in Sample Basic, you can create a generation name called Quarter for generation number 2. Quarter contains quarterly data in the members Qtr1, Qtr2, and so on. When you create the generation name Quarter, Hyperion Essbase creates and enables a Dynamic Time Series member called Q-T-D.

Retrieving Period-to-Date Values

When you retrieve a Dynamic Time Series member, you need to tell Hyperion Essbase the time period up to which you want to calculate the period-to-date value. This time period is known as the *latest time period* and must be a level 0 member on the time dimension.

➤ To specify the latest time period:

- For a specific member, in Hyperion Essbase Spreadsheet Add-in, specify the latest period member name. Place that name after the Dynamic Time Series member or alias name. For example, Q-T-D(May) returns the quarter-to-date value by adding values for April and May.
- For a retrieval, do one of the following:
  - Use the <LATEST command in Report Writer. For more information, see Part VI, “Reporting on Data.”
  - Specify the Latest Time Series option in the Spreadsheet Add-in Essbase Options dialog box. For more information, see the *Hyperion Essbase Spreadsheet Add-in User’s Guide*.

The member-specific setting—for example, Q-T-D(May)—takes precedence over the <LATEST or Latest Time Series option setting.

The following example shows Sample Basic data. Q-T-D(May) displays the period-to-date value for May that is obtained by adding the values for Apr and May (8644 + 8929 = 17573):

	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 30-12: Hyperion Essbase Spreadsheet Add-in Showing Period-To-Date Value for May

Using Dynamic Time Series Members in Partitions

If Dynamic Time Series members are part of the shared area between databases, you need to define the Dynamic Time Series members in both databases, just as you would for regular members. For example, if your partition definition includes Qtr1, Qtr2, Qtr3, Qtr4, and the Dynamic Time Series member Q-T-D, then you need to define the Q-T-D member in both the source database and the target database.

If a Dynamic Time Series member is *not* part of the shared area between databases, Hyperion Essbase gets the data for that member from the source database. You do not need to define the Dynamic Time Series member in both databases. However, this configuration is generally less efficient than including the Dynamic Time Series member in the partition definition.

For more information on partitioning, see Chapter 6, “Designing Partitioned Applications.”

Chapter

31

Developing Calc Scripts

This chapter explains how to develop calc scripts and how to use them to control how Hyperion Essbase calculates a database. It provides some examples of calc scripts, which you may want to adapt for your own use. For more examples, see Chapter 32, “Examples of Calc Scripts.”

This chapter includes the following sections:

For information on developing formulas, see Chapter 26, “Developing Formulas.”

Using 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 the calculations that are defined by the database outline.

For example, the following calc script calculates the Actual values in the Sample Basic database.

Figure 31-1: Calc Script Editor

You can use a calc script to specify exactly how you want Hyperion Essbase to calculate a database. For example, you can calculate part of a database or copy data values between members. You can design and run custom database calculations quickly by separating calculation logic from the database outline.

For most database calculations, a default calculation provides the required results. However, in certain cases, you may need to write a calc script to control how Hyperion Essbase calculates a database.

For example, you need to write a calc script if you want to do any of the following:

- Use the FIX command to calculate a subset of a database. For more information, see “Calculating a Subset of a Database” on page 31-54 and the online *Technical Reference* in the `DOCS` directory.
- Change the calculation order of the dense and sparse dimensions in a database.
- Perform a complex calculation in a specific order or perform a calculation that requires multiple iterations through the data. For example, some two-pass calculations require a calc script. For more information, see Chapter 33, “Optimizing Calculations.”
- Perform any two-pass calculation on a dimension without an accounts tag. For more information, see Chapter 33, “Optimizing Calculations.”
- Perform a currency conversion. For more information, see Part VIII, “Designing and Building Currency Applications.”

- Calculate member formulas that differ from formulas in the database outline. Formulas in a calc script override formulas in the database outline.
- Use an API interface to create a custom calculation dynamically.
- Use logic in a calculation; for example, if you want to use the IF … ELSE … ENDIF or the LOOP … ENDLOOP commands. For more information, see "Controlling the Flow of Calculations" on page 31-12 and the online *Technical Reference* in the `DOCS` directory.
- Clear or copy data from specific members. For more information, see "Controlling the Flow of Calculations" on page 31-12, "Copying Data" on page 31-54, and the online *Technical Reference* in the `DOCS` directory.
- Define temporary variables for use in a database calculation. For more information, see "Declaring Data Variables" on page 31-12 and the online *Technical Reference* in the `DOCS` directory.
- Force a recalculation of data blocks after you have changed a formula or an accounts property on the database outline. For more information, see Chapter 34, "Using Intelligent Calculation to Optimize Calculation."
- Control how Hyperion Essbase uses the Intelligent Calculation feature when calculating a database. For more information, see Chapter 34, "Using Intelligent Calculation to Optimize Calculation."

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 Data."

This example is based on the Sample Basic database, which is supplied with the Hyperion Essbase server installation. This example increases all budget values by 5%. The example assumes that you have Hyperion Essbase Spreadsheet Add-in installed on your machine.

➤ To create the example calc script:

1. Start Hyperion Essbase Application Manager and connect to the Hyperion Essbase server.
2. Select the Sample application and the Basic database, and click the Calc Scripts button,  .

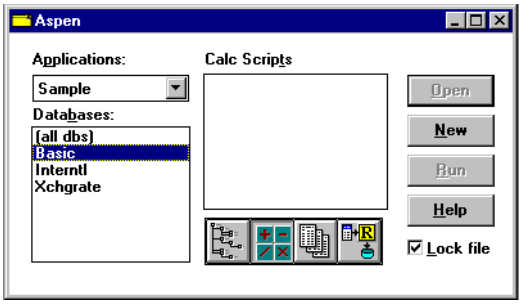


Figure 31-2: Application Desktop Window

If you do not have Sample Basic installed, contact the Hyperion Essbase administrator.

If another user has Sample Basic open and locked, you can clear “Lock file” in the bottom right corner of the application desktop window. However, if you clear “Lock file”, you cannot save your work.

3. Click New to open Calc Script Editor.

4. Type the following calc script. This script increases the expense values of Budget->Marketing by 5%.

```
FIX(Budget)
Marketing = Marketing * 1.05;
ENDFIX
```

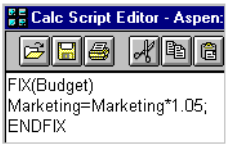


Figure 31-3: Simple Calc Script

For more information on the FIX command, see “Using the FIX Command” on page 31-55 and the online *Technical Reference* in the `DOCS` directory.

5. Click the Check Syntax button,  , to verify that the syntax of the formula you entered is correct.

   The message “No errors” should be displayed at the bottom of Calc Script Editor.

6. Click the Save button,  , to save the calc script. Type **Mycalc1** for the calc script object name, and save the calc script on the server (the default).

7. Close the Calc Script Editor window. Mycalc1 is displayed in the list of calc scripts for Sample Basic.

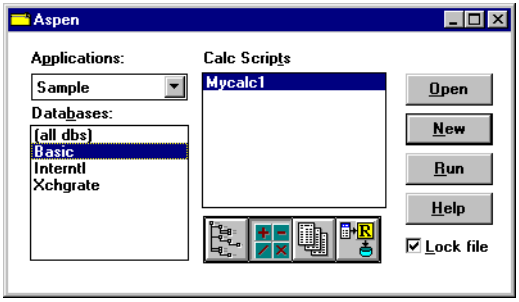


Figure 31-4: Application Desktop Window Showing Calc Script

Calculating Sample Basic Data

You are now ready to calculate the Sample Basic database, increasing the Budget values by 5%. However, first take a look at the Sample Basic Budget values before running the calculation.

➤ To review the Budget values of Sample Basic:

1. Open Hyperion Essbase Spreadsheet Add-in, and select Essbase > Connect to connect to Sample Basic.

   This example assumes that you have not changed the default Hyperion Essbase Spreadsheet Add-in Retrieval Options. For more information, see the *Hyperion Essbase Spreadsheet Add-in User's Guide*.

2. Select Essbase > Retrieve.

   Hyperion Essbase displays the data value from the top level of each dimension.

	A	B	C	D	E
1		Measures	Product	Market	Scenario
2	Year	105522			

Figure 31-5: Hyperion Essbase Spreadsheet Add-in Showing Initial Data

3. Double-click Scenario to display its members.
4. Select Budget and select Essbase > Keep Only.

   Hyperion Essbase displays the Budget values.

5. Double-click Year.

   Hyperion 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 31-6: Hyperion Essbase Spreadsheet Add-in Showing Retrieved Budget Data for Each Quarter

6. Double-click Measures, then Profit, and then Total Expenses to display the Marketing member.
7. Select Marketing and select Essbase > Keep Only.

   Hyperion 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 31-7: Hyperion Essbase Spreadsheet Add-in Showing Retrieved Data for Budget->Marketing

Running a Calc Script

Now you are ready to run the Mycalc1 calc script, which increases the Budget->Marketing values by 5%.

➤ To run the Mycalc1 calc script:

1. Minimize but do not close the Hyperion Essbase Spreadsheet Add-in window.
2. In Hyperion Essbase Application Manager, connect to the Hyperion Essbase server, if you are not already connected.

3. Select the Sample application and the Basic database, and click the Calc Scripts button,  .

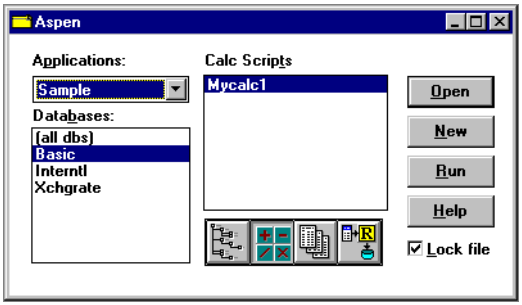


Figure 31-8: Application Desktop Window Showing Calc Script

4. Select Mycalc1 and click Run.
5. When Hyperion Essbase prompts you to select a database, ensure that Sample Basic is selected in the Select Database dialog box. Click OK.

   Hyperion Essbase calculates the database.

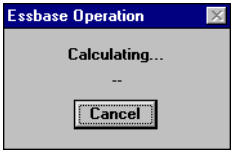


Figure 31-9: Calculating Message Box

Checking a Calculation

After Hyperion Essbase finishes a calculation, you can check the dimensions calculated and the calculation time in the application event log.

➤ To review the application event log:

1. In Hyperion Essbase Application Manager, select Application > View Event Log.
2. In the View Log File dialog box, select Date to view the entries for the current date.
3. When Hyperion Essbase displays the application event log, scroll to the end of the file to see the entries for a calculation. The entries will be similar to the following:

```
[Tue Feb 11 10:57:45 1999]Local/Sample/Basic/Joanne/Info(1013091)
Received Command [Calculate] from user [Joanne]

[Tue Feb 11 10:57:48 1999]Local/Sample/Basic/Joanne/Info(1012668)
Calculating [ Measures(Marketing)] with fixed members [Scenario(Budget)]

[Tue Feb 11 10:58:08 1999]Local/Sample/Basic/Joanne/Info(1012550)
Total Calc Elapsed Time : [19.989] seconds

[Tue Feb 11 10:58:14 1999]Local/Sample/Basic/Joanne/Info(1019018)
Writing Parameters For Database [Basic]
```

Figure 31-10: Application Event Log Showing Calculation Messages

From the entries, you can see that Hyperion Essbase calculated data values for the Marketing member on the Measures dimension, fixing on the Budget values. Hyperion Essbase calculated the database in 19.989 seconds.

Note: Check the “fixed members [ ]” part of the message to ensure that all members you fixed on were actually included in the calculation.

These entries are an example of the default level of messages that Hyperion 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 the `DOCS` directory.

4. Close the application log viewer window.

➤ To view newly calculated data:

1. Maximize the Hyperion Essbase Spreadsheet Add-in window.

   The pre-calculation data values should still be displayed. If the pre-calculation data values are not displayed, repeat the steps in “Calculating Sample Basic Data” on page 31-6.

2. Select Essbase > Retrieve to display the new data 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 31-11: Hyperion Essbase Spreadsheet Add-in Showing Retrieved New Data

As you can see, the data values have increased by 5%. The calculation is successful. If required, you can reload the default data into Sample Basic. See Part IV, “Loading Data.”

Building a Calc Script in Calc Script Editor

You use Calc Script Editor to build a calc script. You can type the calc script directly into the text area of Calc Script Editor, or you can use the user interface features of Calc Script Editor to build the calc script.

Calc scripts are ASCII text. If desired, you can create a calc script in the text editor of your choice and paste it into Calc Script Editor.

Hyperion Essbase provides a flexible set of commands that you can use to control how a database is calculated. You can construct calc scripts from commands and formulas. Several types of commands are discussed in the following sections, including the following:

- Computation
- Control flow
- Data declaration
- Global settings

Implementing Outline Calculations

You can use the following calculation commands to perform a database calculation that is based on the structure and formulas in the database outline.

Note: For a complete list of calculation commands and syntax, see the online *Technical Reference* in the `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 two-pass on the dimension tagged as accounts	CALC TWOPASS
The formula applied to a member in the database outline, where *membername* is the name of the member to which the formula is applied	*membername*
All members tagged as Average on the dimension tagged as accounts (see Chapter 30, “Calculating Time Series Data”)	CALC AVERAGE
All members tagged as First on the dimension tagged as accounts (see Chapter 30, “Calculating Time Series Data”)	CALC FIRST
All members tagged as Last on the dimension tagged as accounts (see Chapter 30, “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 the `DOCS` directory.

To...	Use...
Calculate a subset of a database	FIX … ENDFIX
Specify the number of times that commands are iterated	LOOP … ENDLOOP

You can also use the IF and ENDIF commands to specify conditional calculations. See “Controlling the Flow of Calculations” on page 31-12.

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, to set their initial values. *Temporary variables* store the results of intermediate calculations.

You can also use substitution variables in a calc script. See “Using Substitution Variables” on page 31-52.

To...	Use...
Declare one-dimensional array variables	ARRAY
Declare a temporary variable that contains a single value	VAR

For detailed information on the these commands, see the online *Technical Reference* in the `DOCS` directory.

Values stored in temporary variables exist only while the calc script is running. You cannot report on the values of temporary variables.

Typically, arrays are used to store variables as part of a member formula. The size of the array variable is determined by the number of members in the corresponding dimension. For example, if the Scenario dimension has four members, the following command creates an array called `Discount` with four entries. You can use more than one array at a time.

```
ARRAY Discount[Scenario];
```

Specifying Global Settings for a Database Calculation

You can use the following commands to define calculation behavior.

Note: For a complete list of commands, see the online *Technical Reference* in the `DOCS` directory.

To…	Use…
Specify how Hyperion 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 33, “Optimizing Calculations”)	SET CALCHASHTBL
Optimize the calculation of sparse dimension formulas in large database outlines (see Chapter 33, “Optimizing Calculations”)	SET FRMLBOTTOMUP
Display messages to trace a calculation.	SET MSG SET NOTICE
Turn on and turn off Intelligent Calculation (see Chapter 34, “Using Intelligent Calculation to Optimize Calculation”)	SET UPDATECALC
Control how Hyperion Essbase marks data blocks for the purpose of Intelligent Calculation (see Chapter 34, “Using Intelligent Calculation to Optimize Calculation”)	SET CLEARUPDATESTATUS

To…	Use…
Specify the maximum number of blocks that Hyperion 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. A SET command in a calc script stays in effect until the next occurrence of the same SET command.

For example, consider the following calc script:

```
SET MSG DETAIL;
CALC DIM(Year);

SET MSG SUMMARY;
CALC DIM(Measures);
```

Hyperion Essbase displays messages at the detail level when calculating the Year dimension. However, when calculating the Measures dimension, Hyperion Essbase displays messages at the summary level.

Now, consider this calc script:

```
SET AGGMISSG ON;
Qtr1;

SET AGGMISSG OFF;
East;
```

Hyperion Essbase calculates member combinations for Qtr1 with SET AGGMISSG (aggregate missing values) turned on. Hyperion Essbase then does a second calculation pass through the database and calculates member combinations for East with SET AGGMISSG turned off. For more information on the setting for aggregating missing values, see the SET AGGMISSG command in the online *Technical Reference* in the `DOCS` directory. For more information on calculation passes, see Chapter 33, “Optimizing Calculations.”

Adding Comments

You can include comments to annotate calc scripts. Hyperion Essbase ignores these comments when it runs the calc script.

To include a comment, start the comment with /* and end the comment with */. For example, consider the following comment:

```
/*   This is a calc script comment
     that spans two lines.*/
```

Composing 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 (;), as shown in the following four examples:

  ```
  CALC DIM(Product, Measures);

  DATACOPY Plan TO Revised_Plan;

  "Market Share" = Sales % Sales->Market;

  IF
    (Sales <> #MISSING) Commission = Sales * .9;
  ELSE
    Commission = #MISSING;
  ENDIF;
  ```

  You do not need to end the following commands with semicolons: IF, ELSE, ELSEIF, FIX, ENDFIX, LOOP, and ENDLOOP.

- Enclose the member name in double quotation marks ("") if the member name meets any of the following conditions:

  – Contains spaces; for example,
  `"Opening Inventory" = "Ending Inventory" - Sales + Additions;`

  – Is the same as an operator or function name.

  – Includes any non-alphanumeric character; for example, hyphens (-), asterisks (*), and slashes (/).

  – Is all numeric or starts with a numeral; for example, “100” or “10Prod”.

  - Begins with an ampersand (&). The leading ampersand (&) is reserved for substitution variables. If a member name begins with &, enclose it in quotation marks. Do not enclose substitution variables in quotation marks in a calc script.
  - Contains a dot (.); for example, “1998.Jan” or “.100”.

  For a complete list of member names that must be enclosed in quotation marks, see “Rules for Naming Dimensions and Members” on page 8-15.

- If you are using an IF statement or an interdependent formula, enclose the formula in parentheses and associate it with the specified member. For example, the following formula is associated with the Commission member in the database outline:

  ```
  Commission
  (IF(Sales < 100)
     Commission = 0;
  ENDIF;)
  ```

- End each IF statement in a formula with an ENDIF statement. For example, the following formula contains a simple IF...ENDIF statement:

  ```
  Profit
  (IF (Sales > 100)
     Commission = Sales * .1;
  ENDIF;)
  ```

- If you are using an IF statement that is nested within another IF statement, end each IF with an ENDIF. For example, consider the following example:

  ```
  "Opening Inventory"
  (IF (@ISMBR(Budget))
     IF (@ISMBR(Jan))
     "Opening Inventory" = Jan;
     ELSE
     "Opening Inventory" = @PRIOR("Ending Inventory");
     ENDIF;
  ENDIF;)
  ```

- You do not need to end ELSE or ELSEIF statements with ENDIFs. For example, consider the following:

```
Marketing
(IF (@ISMBR(@DESCENDANTS(West)) OR @ISMBR(@DESCENDANTS(East)))
    Marketing = Marketing * 1.5;
ELSEIF(@ISMBR(@DESCENDANTS(South)))
    Marketing = Marketing * .9;
ELSE Marketing = Marketing * 1.1;
ENDIF;)
```

  Note: If you use ELSE IF (with a space in between) rather than ELSEIF (one word) in a formula, you must supply an ENDIF for the IF statement.

- Although ending ENDIF statements with a semicolon (;) is not required, it is good practice to follow each ENDIF statement in a formula with a semicolon.
- End each FIX statement with an ENDFIX statement. For example:

```
FIX(Budget,@DESCENDANTS(East))
CALC DIM(Year, Measures, Product);
ENDFIX
```

  The FIX and ENDFIX statements do not need to be followed by a semicolon (;).

When you write a calc script, you can use the Calc Script Editor syntax checker to check the syntax. For more information, see “Checking Syntax” on page 31-48.

Note: For detailed information on calc script syntax, see the online *Technical Reference* in the `DOCS` directory.

Opening Calc Script Editor

Open Calc Script Editor to create a new calc script or open an existing calc script.

For information on opening an existing calc script, see “Changing a Calc Script” on page 31-20.

➤ To open Calc Script Editor:

1. Open Hyperion Essbase Application Manager and connect to the Hyperion Essbase server.
2. In the application desktop server window, select the desired application and database.
3. Click the Calc Scripts button,  .

   Hyperion Essbase displays a list of all the calc scripts associated with the application and database that you selected.

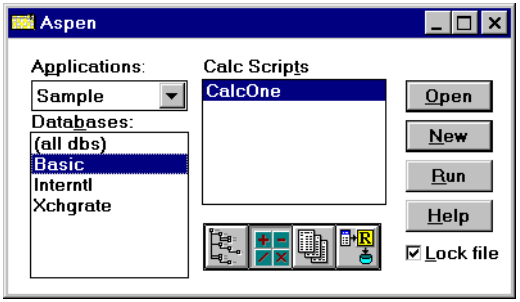


Figure 31-12: Application Desktop Window

4. To create a new calc script, click New.

   To open an existing calc script, select it in the Calc Scripts list and click Open.

   Hyperion Essbase opens Calc Script Editor:

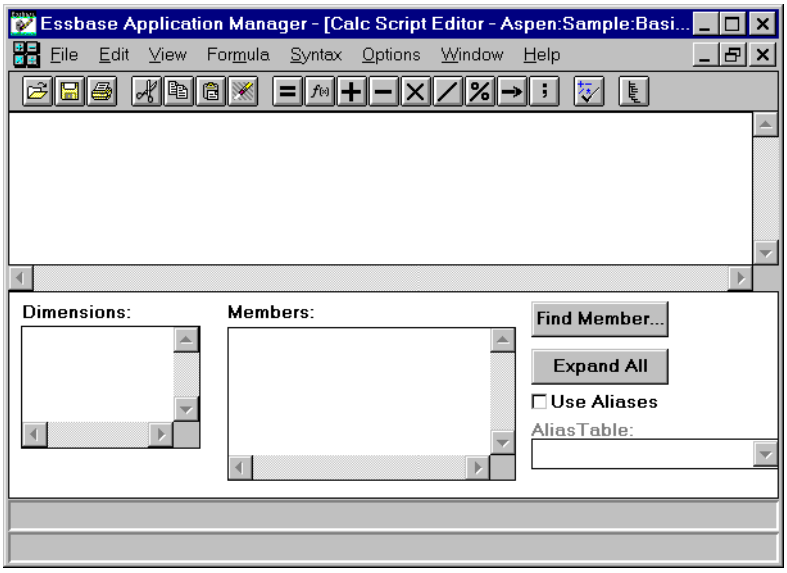


Figure 31-13: Calc Script Editor

Adding a Calc Script

You can use Hyperion Essbase Application Manager to add a new calc script.

➤ To add a calc script:

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 Calc Scripts button,  , and then click New.

   Alternatively, from the Hyperion Essbase Application Manager menu, select File > New > Calc Script.

   Hyperion Essbase opens Calc Script Editor. You can now build a calc script. Hyperion Essbase prompts you to name a calc script when you save it. See “Saving a Calc Script” on page 31-24.

Changing a Calc Script

To change a calc script, open it in Calc Script Editor. How you do that depends on where the calc script is stored.

➤ To open a calc script that 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 Hyperion Essbase server called Aspen. The Sample Basic database is selected.

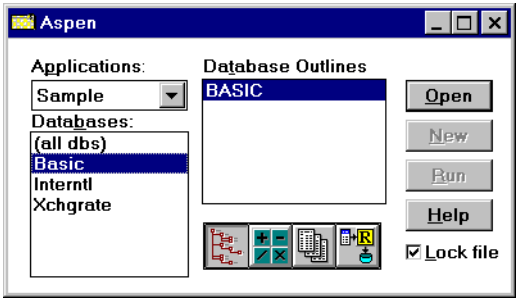


Figure 31-14: Application Desktop Server Window

2. Click the Calc Scripts button,  .

   Hyperion 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 Hyperion Essbase and *appname* and *dbname* are the current application and database.

   For example, assuming that the Hyperion Essbase install directory is `C:\ESSBASE`, if you select Sample Basic, Hyperion Essbase displays the `.CSC` files in the `C:\ESSBASE\APP\SAMPLE\BASIC` directory.

3. In the Calc Scripts list, select the calc script you want to modify.
4. Click Open.

   Hyperion Essbase opens Calc Script Editor.

You can now edit the calc script.

➤ To open a calc script that is on a different server:

1. Ensure that the focus is on the application desktop server window.
2. From the Hyperion Essbase Application Manager menu, select File > Open.

   Hyperion Essbase displays the Open Server Object dialog box.

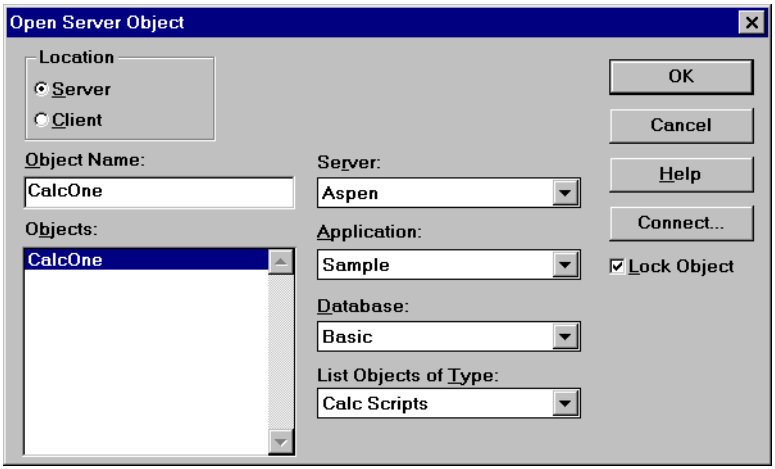


Figure 31-15: Open Server Object Dialog Box

3. Click Connect 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. In the Objects list, select the required calc script, and click OK.

   Hyperion Essbase displays the calc script in Calc Script Editor.

➤ To open a calc script that is on a client machine:

1. In the application desktop client window, select the application and database that contains the calc script.

2. Click the Calc Scripts button,  .

   Hyperion 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 Hyperion Essbase, and *appname* and *dbname* are the current application and database on your client machine.

   For example, assuming that the Hyperion Essbase install directory is `C:\ESSBASE`, if you have an application called MYAPP01 and a database called MYDB01 on your client machine, Hyperion 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.

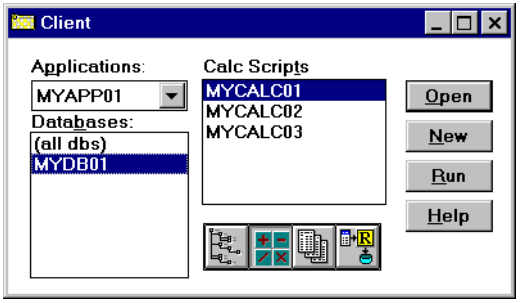


Figure 31-16: Application Desktop Client Window

3. In the Calc Scripts list, select the required calc script.
4. Click Open.

   Hyperion Essbase opens Calc Script Editor. You can now edit the calc script.

➤ To open a calc script that is on a client machine but is not saved as a Hyperion Essbase object:

1. From the Hyperion Essbase Application Manager menu, select File > Open.

   Hyperion Essbase displays the Open Client Object dialog box.

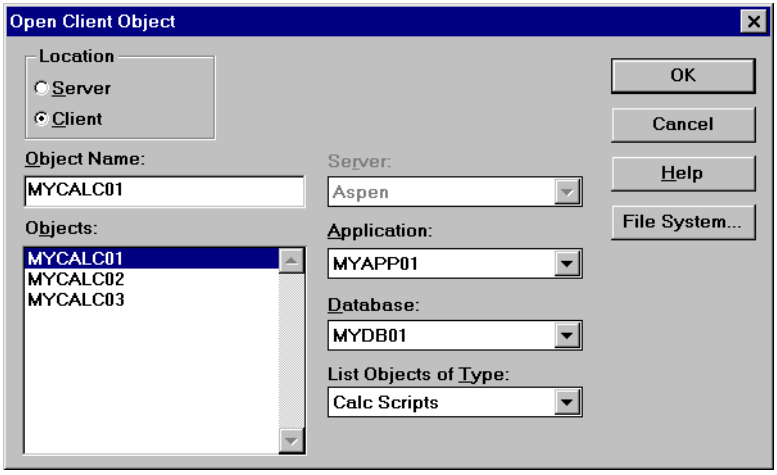


Figure 31-17: Open Client Object Dialog Box

2. Click File System to connect to the other server.

   Hyperion Essbase displays the Open Client File dialog box.

3. Select the file that contains the required calc script, and click OK.

   Hyperion Essbase displays the calc script in Calc Script Editor.

Saving a Calc Script

You can save a calc script as either of the following:

- An object on the Hyperion Essbase server or on a client machine. You can associate the calc script object with either of the following:
  - An application and all the databases within the application, which 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 Hyperion 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 may want to save it on your client machine. Then move the completed script to the Hyperion Essbase server.

When you save a calc script from Calc Script Editor, by default Hyperion 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 Hyperion Essbase Application Manager or from Hyperion Essbase Spreadsheet Add-in, it must have a `.CSC` extension. However, a calc script is an ASCII file, and you can use ESSCMD to run any ASCII file as a calc script.

A calc script can also be a string defined in memory. You can access this string via the API on the Hyperion Essbase client or Hyperion Essbase server. Thus, from dialog boxes, you can dynamically create a calc script that is based on user selections.

➤ To save a calc script as an object on the Hyperion Essbase server:

1. In Calc Script Editor, click the Save button,  .

   Hyperion Essbase displays the Save Server Object dialog box.

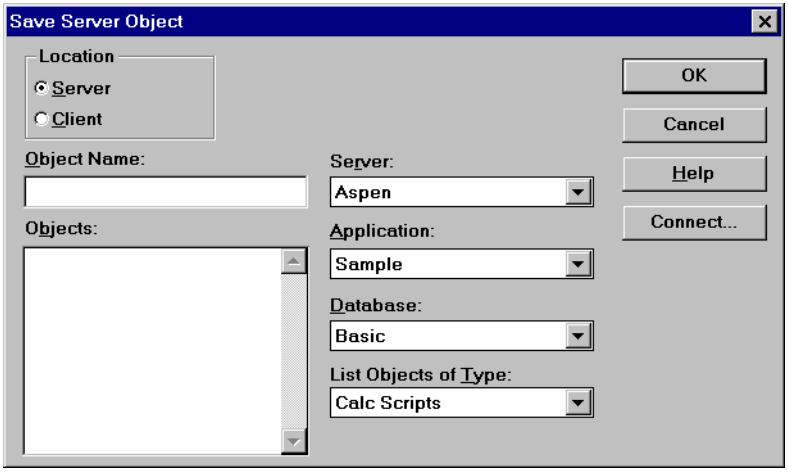


Figure 31-18: Save Server Object Dialog Box

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. In the Application list, select the required application.

   b. In the Database list, select “(all dbs)” (all databases).

   To associate the calc script with a database:

   a. In the Application list, select the application containing the database.

   b. In the Database list, select the required database.

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.

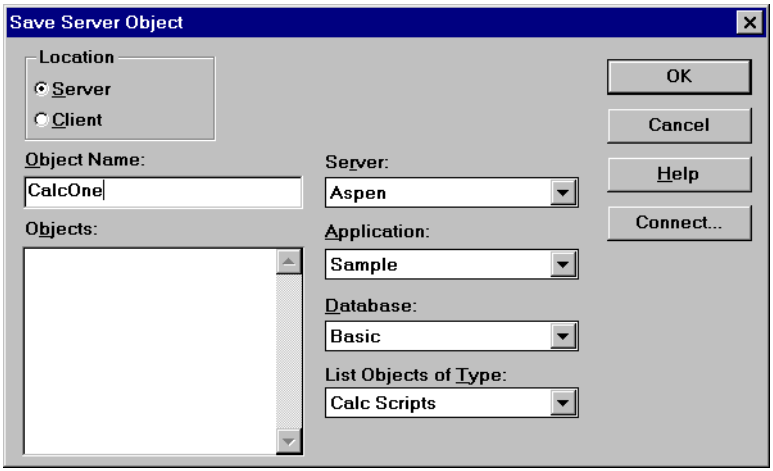


Figure 31-19: Save Server Object Dialog Box

4. Click OK.

   Hyperion Essbase saves the calc script as a calculation script object on the Hyperion Essbase server.

Calc script objects associated with an application are saved in the *\ARBORPATH\APP\appname* directory on the Hyperion Essbase server machine. Calc script objects associated with a database are saved in the *\ARBORPATH\APP\appname\dbname* directory on the Hyperion Essbase server machine. *ARBORPATH* is the Hyperion Essbase install directory, and *appname* and *dbname* are the application and database with which you have associated the calc script.

For example, consider the following:

- If you associate a calc script called CalcTwo with the Sample Basic database and the Hyperion Essbase install directory is `C:\ESSBASE`, Hyperion Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\APP\SAMPLE\BASIC`.
- If you associate a calc script called CalcTwo with the Sample application and the Hyperion Essbase install directory is `C:\ESSBASE`, Hyperion Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\APP\SAMPLE`.

➤ To save a calc script as an object on your client machine:

1. In Calc Script Editor, click the Save button, .

   If the calc script is new, Hyperion Essbase displays either the Save Server Object dialog box or the Save Client Object dialog box, depending on whether you opened Calc Script Editor from the application desktop server window or the client window.

   If Hyperion Essbase displays the Save Server Object dialog box, under Location, select Client. The Save Client Object dialog box replaces the Save Server Object dialog box.

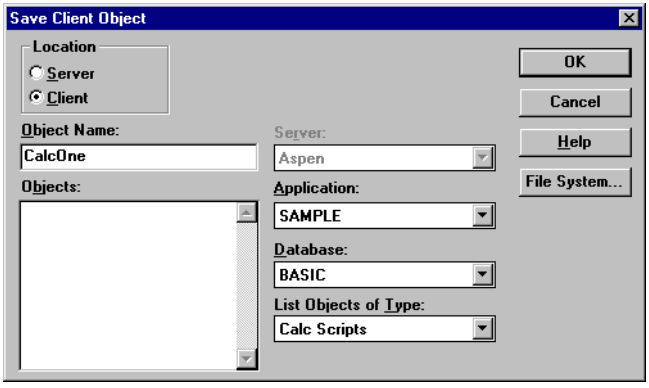


Figure 31-20: Save Client Object Dialog Box

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. In the Application list, select the required application.

   b. In the Database list, select "(all dbs)" (all databases).

   To associate the calc script with a database:

   a. In the Application list, select the application containing the database.

   b. In the Database list, select the required database.

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.

   Hyperion Essbase saves the calc script as a calc script object on your client machine.

Calc script objects associated with an application are saved in the `\ARBORPATH\CLIENT\appname` directory on the Hyperion Essbase client machine. Calc script objects associated with a database are saved in the `\ARBORPATH\CLIENT\appname\dbname` directory on the Hyperion Essbase client machine. *ARBORPATH* is the Hyperion Essbase install directory, and *appname* and *dbname* are the application and database with which you associate the calc script.

For example, consider the following:

- If you associate a calc script called CalcTwo with the MYDB01 database in the MYAPP01 application and the Hyperion Essbase install directory is `C:\ESSBASE`, Hyperion Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\CLIENT\MYAPP01\MYDB01`.
- If you associate a calc script called CalcTwo with the MYAPP01 application and the Hyperion Essbase install directory is `C:\ESSBASE`, Hyperion Essbase saves CalcTwo as `CALCTWO.CSC` in `C:\ESSBASE\CLIENT\MYAPP01`.

➤ To save a calc script in the file system of a client machine:

1. In Calc Script Editor, click the Save button,  .

   If the calc script is new, Hyperion Essbase displays either the Save Server Object dialog box or the Save Client Object dialog box, depending on whether you opened Calc Script Editor from the application desktop server window or the client window.

   If Hyperion Essbase displays the Save Server Object dialog box, under Location, select Client. The Save Client Object dialog box replaces the Save Server Object dialog box.

2. In the Save Client Object dialog box, click File System.

   Hyperion Essbase displays the Save Client File dialog box.

3. Enter the required directory and file name, and click OK.

   Hyperion Essbase saves the calc script in the directory you specified.

➤ To copy a calc script from a client machine to the Hyperion Essbase server:

1. In Calc Script Editor, open the calc script that you want to copy. For more information, see “Changing a Calc Script” on page 31-20.

2. From the Hyperion Essbase Application Manager menu, select File > Save As.

   Hyperion Essbase displays the Save Client Object dialog box.

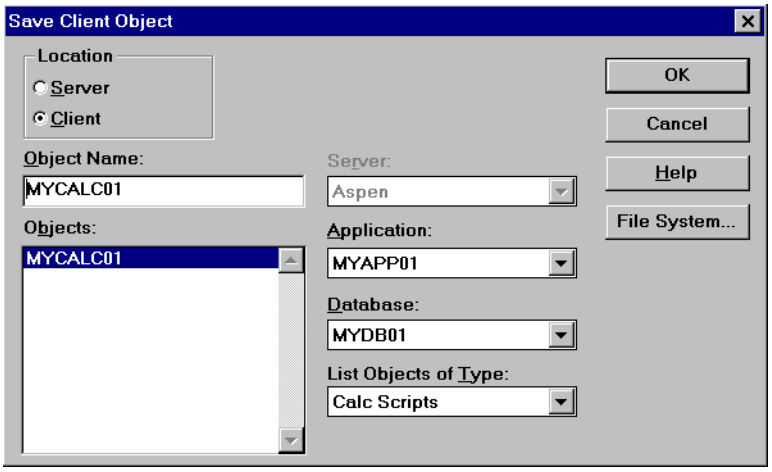


Figure 31-21: Save Client Object Dialog Box

3. Under Location, select Server.

   The Save Server Object dialog box replaces the Save Client Object dialog box.

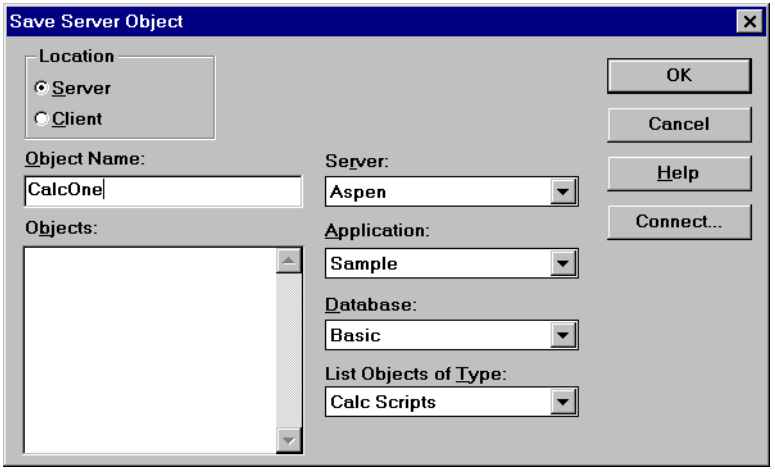


Figure 31-22: Save Server Object Dialog Box

4. Select the application or database with which you want to associate the calc script. For more information, see “Saving a Calc Script” on page 31-24.
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.

   Hyperion Essbase saves the calc script on the Hyperion Essbase server. For more information, see“Saving a Calc Script” on page 31-24.

Running a Calc Script

You can run a calc script from any of the following:

- Hyperion Essbase Application Manager
- Hyperion Essbase Spreadsheet Add-in (See the *Hyperion Essbase Spreadsheet Add-in User’s Guide*.)
- ESSCMD

If you run a calc script from the Hyperion Essbase Application Manager, you can run it on your Hyperion Essbase client machine or on the Hyperion Essbase server.

When you run a calc script from Hyperion Essbase Application Manager or from Hyperion Essbase Spreadsheet Add-in, you can view the calculation messages in the event log file. When you use ESSCMD to run a calc script, Hyperion Essbase displays the messages in the ESSCMD window. To display the event log file, select Application > View Event Log from the Hyperion Essbase Application Manager menu.

Hyperion Essbase displays both of the following:

- The calculation order of the dimensions for each pass through the database
- The total calculation time

You can use these messages to tune a database during calculation. To display more detailed information, you can use the SET MSG SUMMARY, SET MSG DETAIL, and SET NOTICE commands in a calc script. For more information, see “Specifying Global Settings for a Database Calculation” on page 31-13.

You can run a calc script from the Hyperion Essbase Application Manager desktop or from the Hyperion Essbase Application Manager menu.

Note: Before you can run a calc script in Hyperion Essbase Application Manager, you must save it as a calc script object on the Hyperion Essbase server or on your client machine. See "Saving a Calc Script" on page 31-24. 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 Hyperion 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 window or the client window, select the application and database that contains the calc script you want to run.

3. Click the Calc Scripts button,  , to display the calc scripts associated with the application and database that you selected. The following example shows the application desktop server window for a server called Aspen:

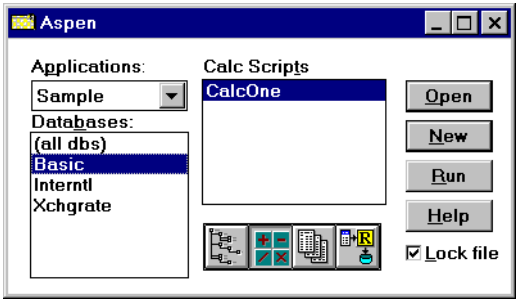


Figure 31-23: Application Desktop Server Window

4. In the Calc Scripts list, select the calc script that you want to run, and click Run.

   Hyperion Essbase displays the Select Database dialog box.

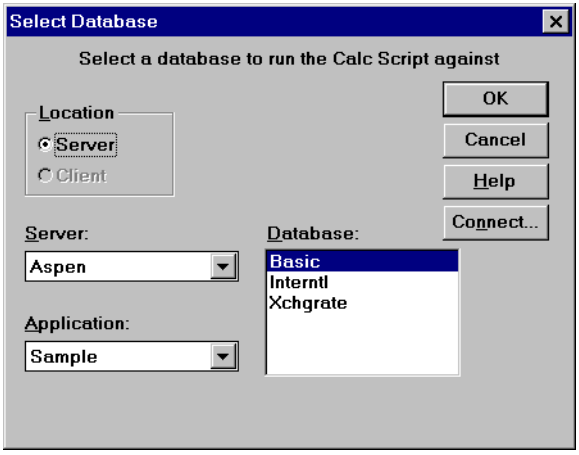


Figure 31-24: Select Database Dialog Box

5. Select the Hyperion 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.

   Hyperion Essbase runs the calc script against the database that 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 that you want to run.
2. From the Hyperion Essbase Application Manager menu, select Database > Calculate.

   Hyperion Essbase displays the Calculate Database dialog box.

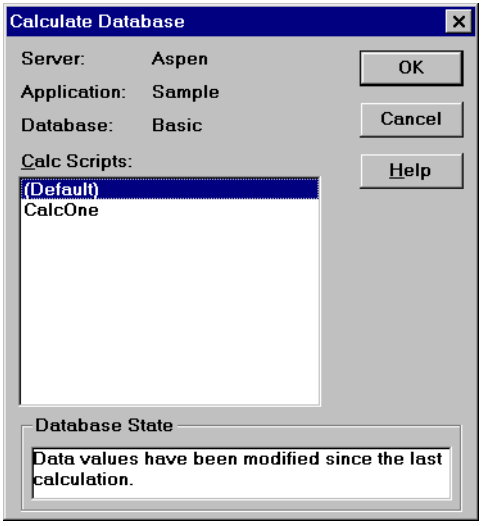


Figure 31-25: Calculate Database Dialog Box

3. In the Calc Scripts list, select the calc script that you want to run and click OK. Only scripts to which you have security access are displayed in the list.

   Hyperion 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 the `DOCS` directory for information about this command. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

For information on running a calc script from Hyperion Essbase Spreadsheet Add-in, see the *Hyperion Essbase Spreadsheet Add-in User’s Guide*.

Printing a Calc Script

You can print a calc script from Calc Script Editor.

➤ To print a calc script:

1. In Calc Script Editor, open the calc script. For more information, see “Opening Calc Script Editor” on page 31-18.
2. Select File > Print, or click the Print button,  .

   Hyperion Essbase displays the Print Calc Script dialog box.
3. If you want to print page numbers on a calc script, check Page Numbers.
4. Click Print.

Deleting a Calc Script

How you delete a calc script depends on where it is saved.

➤ To delete a calc script saved as an object on the Hyperion Essbase server or on a 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 Calc Scripts button,  .

   Hyperion Essbase displays a list of all the calc scripts associated with the application and database that you chose.

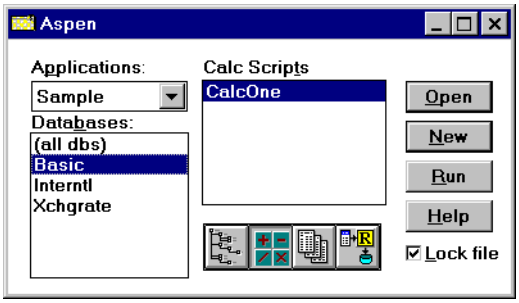


Figure 31-26: Application Desktop Server Window

3. In the Calc Scripts list, select the calc script that you want to delete.
4. From the Hyperion Essbase Application Manager menu, select File > Delete.

   Hyperion Essbase displays a Confirm Delete message box.
5. Click Yes to delete the calc script.

➤ To delete a calc script saved in the file system of a client machine:

You cannot delete the file using Hyperion Essbase Application Manager. Delete the calc script file by using the client machine's file system.

➤ To undo the last action:

In Calc Script Editor, select Edit > Undo, or click the  button.

Using Formulas in a Calc Script

You can place member formulas in a calc script. When you place formulas in a calc script, they override any conflicting formulas that are applied to members in the database outline.

In a calc script, you can do both of the following:

- Calculate a member formula on the database outline
- Define a formula

To calculate a formula that is applied to a member in the database outline, simply use the member name followed by a semicolon (;). For example:

```
Variance;
```

calculates the formula applied to the Variance member in the database outline.

To define a formula in a calc script, use Calc Script Editor. For example:

```
Expenses = Payroll + Marketing + Misc;
```

cycles through the database, adding the values in the members Payroll, Marketing, and Misc and placing the result in the Expenses member. This formula overrides any formula placed on the Expenses member in the database outline.

Note: You cannot apply formulas to shared members or label only members.

Basic Equations

You can define basic equations in a calc script as follows:

Member = mathematical_operation;

where *Member* is a member name from the database outline, and *mathematical_operation* is any valid mathematical operation.

For example, the following formula causes Hyperion Essbase to cycle through the database, subtracting the values in COGS from the values in Sales and placing the result in Margin:

```
Margin = Sales - COGS;
```

The next formula cycles through the database subtracting the values in Cost from the values in Retail, calculating the resulting values as a percentage of the values in Retail, and placing the results in Markup:

```
Markup = (Retail - Cost) % Retail;
```

For more information on the nature of multidimensional calculations, see Chapter 3, “Multidimensional Concepts.”

Conditional Equations

When you use an IF statement as part of a member formula in a calc script, you need to do both of the following:

- Associate the IF statement with a single member
- Enclose the IF statement in parentheses

For example:

```
Profit
(IF (Sales > 100)
   Profit = (Sales - COGS) * 2;
ELSE
   Profit = (Sales - COGS) * 1.5;
ENDIF;)
```

Hyperion Essbase cycles through the database and performs the following calculations:

1. The IF statement checks to see if the value of Sales for the current member combination is greater than 100.
2. If Sales is greater than 100, Hyperion Essbase subtracts the value in COGS from the value in Sales, multiplies the difference by 2, and places the result in Profit.
3. If Sales is less than or equal to 100, Hyperion Essbase subtracts the value in COGS from the value in Sales, multiplies the difference by 1.5, and places the result in Profit.

The whole of the IF … ENDIF statement is enclosed in parentheses and associated with the Profit member, `Profit(IF(...)...)`.

Interdependent Formulas

When you use an interdependent formula in a calc script, the same rules apply as for the IF statement. You need to do both of the following:

- Associate the formula with a single member
- Enclose the formula in parentheses

Consider the interdependent formula discussed earlier. If you place the formula in a 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 is associated with the Opening Inventory member, as follows:

```
"Opening Inventory"(IF(...)...)
```

Inserting Text and Operators in a Calc Script

You can type text and operators directly into the text area of Calc Script Editor, or you can use the toolbar buttons to add the text and operators. You can also cut, copy, and search for text in Calc Script Editor.

➤ To type text in Calc Script Editor:

1. Click in the text area below the toolbar.
2. Type the appropriate text.

   Text is displayed at the cursor position as you type.

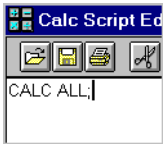


Figure 31-27: Calc Script Editor Showing Calc Script

➤ To insert an equal (=) sign in Calc Script Editor:

1. Place the cursor where you want to insert the equal sign (=).
2. Type = or click the  button.

➤ To insert a mathematical operator (+, -, X, /, %) in Calc Script Editor:

1. Place the cursor where you want to insert the mathematical operator.
2. Type the operator or click one of the following toolbar buttons:

For example, to insert an addition operator (+), place the cursor where you want to insert the addition (+) operator, and type + or click the  button.

➤ To insert the cross-dimensional operator (->) in Calc Script Editor:

1. Place the cursor where you want to insert the cross-dimensional operator.
2. Type a - (hyphen) followed by a > (greater than symbol), or click the  button.

For more information on the cross-dimensional operator, see Chapter 26, “Developing Formulas.”

➤ To insert the semicolon formula end-of-line character (;) in Calc Script Editor:

1. Place the cursor at the end of the formula.
2. Type a **;** (semicolon), or click the  button.

➤ To insert a function or operator in Calc Script Editor:

1. Place the cursor where you want to insert the function.
2. Select Formula > Paste Function, or click the  button.

   Hyperion Essbase displays the Function Templates dialog box.
3. In the Categories list, select the function category.

   For example, to insert the @VAR function, select Math.
4. In the Templates list, select the required function or operator.

   For example, scroll down the list and select @VAR. Hyperion Essbase displays the function or operator and the default arguments below the Categories list.

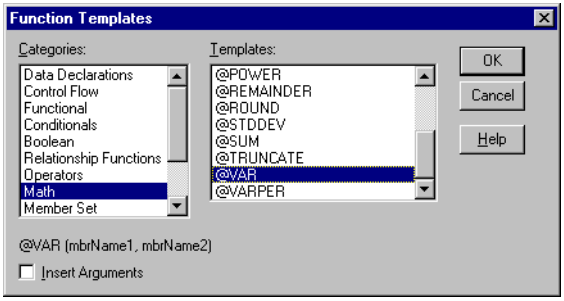


Figure 31-28: Function Templates Dialog Box

5. If required, check Insert Arguments to insert default, temporary arguments for the function.
6. Click OK.

   Hyperion Essbase inserts @VAR at the cursor position.

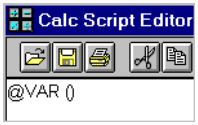


Figure 31-29: Calc Script Editor With @VAR Function Inserted

If you checked Insert Arguments, Hyperion Essbase inserts @VAR and default, temporary arguments. You can then type over the temporary arguments with the correct arguments.

Figure 31-30: Calc Script Editor With @VAR Function and Arguments Inserted

➤ To cut text in Calc Script Editor:

1. Select the text that you want to cut.
2. Select Edit > Cut, click the  button, or press Ctrl + X.

➤ To copy text in Calc Script Editor:

1. Select the text that you want to copy.
2. Select Edit > Copy, click the  button, or press Ctrl + C.

➤ To paste text in Calc Script Editor:

1. Select the text that you want to paste.
2. Select Edit > Paste, click the  button, or press Ctrl + V.

➤ To find and replace text in Calc Script Editor:

1. Select Edit > Find.

   Hyperion Essbase displays the Find dialog box.

Figure 31-31: Calc Script Editor Find Dialog Box

2. In the “Find what” text box, type the characters you want to search for, and click Find Next.

➤ To do a case-sensitive search:

1. In the Find dialog box, check Match case.

   For example, to search for Margin but not margin, type Margin in the Find what text box and check Match case.
2. Click Find Next.

Associating a Calc Script with a Database

If you want to insert member names in a calc script by selecting them within Calc Script Editor, you need to associate the calc script with the database outline that contains the members.

➤ To associate a calc script with a database outline:

1. Click the  button or, from the Calc Script Editor menu, select Options > Associate Outline.

   Hyperion Essbase displays the Associate Client Outline Object or Associate Server Outline Object dialog box.

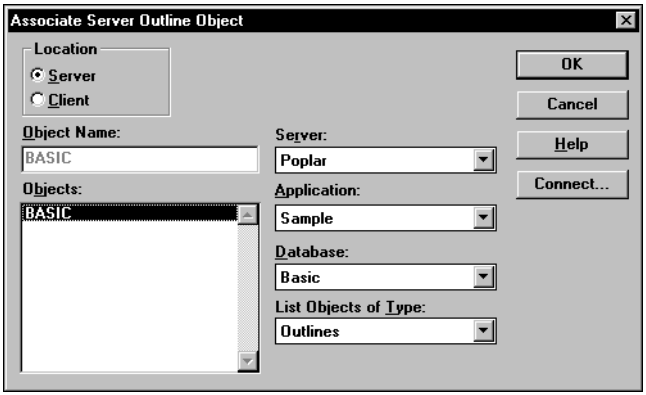


Figure 31-32: Associate Server Outline Object Dialog Box

2. Do one of the following:
   - To associate an outline saved on your client machine, under Location, select Client.
   - To associate an outline saved on the Hyperion Essbase server machine, under Location, select Server.
3. In the Server, Application, and Database lists, select the server, application, and database that contain the outline that you want to associate with a calc script.
4. In the Objects list, select the database outline.

5. Click OK.

   Hyperion Essbase displays the dimension names of the associated outline in the Dimensions list. You can now insert members from this list. See "Associating a Calc Script with a Database" on page 31-43.

Hyperion Essbase associates the calc script with the database outline only while you are editing the calc script. When you close Calc Script Editor, Hyperion Essbase cancels the association. If you want to insert members from the database outline in the future, you need to re-associate the outline with the calc script.

➤ To insert the name of a dimension in a calc script:

1. Associate the database outline that contains the dimensions you want to insert. See "Associating a Calc Script with a Database" on page 31-43.
2. In 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 a formula.

   The name of the dimension is displayed in the Members list. If a ☑ button is displayed to the left of the dimension name, the dimension has children. The following shows the Scenario dimension in the Sample Basic database.

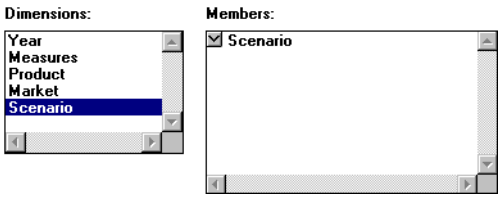


Figure 31-33: Inserting Dimensions and Members In a Calc Script

If you want to insert the name of the dimension in a formula, click the dimension name in the Members list. Hyperion Essbase inserts the dimension name at the cursor position.

➤ To expand and collapse a member branch:

1. To display a member's children, in the Members list, double-click the button next to the member name.

   The button changes to a button.

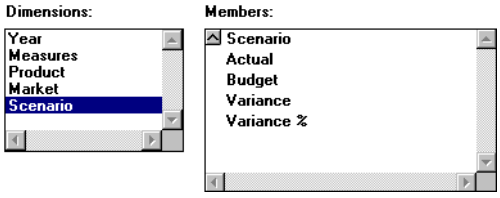


Figure 31-34: Expanding a Member Branch

2. To collapse the member branch, double-click the button.

   Hyperion Essbase does not display the member's children. The button changes to a button.

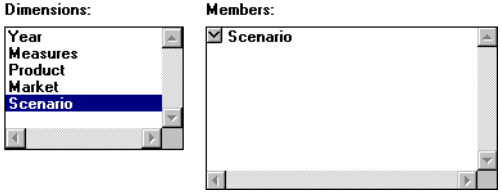


Figure 31-35: Collapsing a Member Branch

➤ To search for a member:

1. In the Dimensions list, select the dimension in which you want to search for a member.

   For example, select the Measures dimension from the Sample Basic database.

2. Click Find Member.

   Hyperion Essbase displays the Find dialog box.

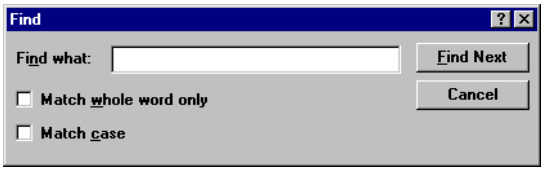


Figure 31-36: Searching For Members

3. In the Find what text box, type the characters that you want to search for.

   For example, to search for the Marketing member in the Measures dimension, enter **market**.

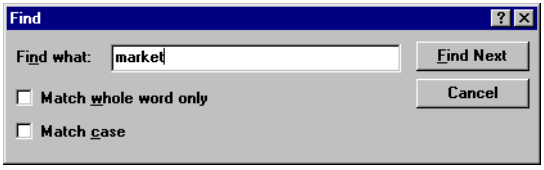


Figure 31-37: Searching For Members

   a. To make the search case-sensitive, check Match case.

   b. To search for whole words only, check Match whole word only.

      For example, to search for Margin, but not Margin % in the Sample Basic database, type **margin**, and check Match whole word only.

4. Click Find Next.

   Hyperion Essbase finds and selects the appropriate member.

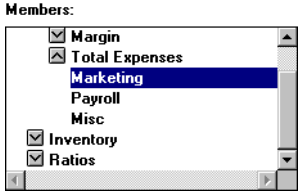


Figure 31-38: Searching For Members

➤ To expand a dimension to display all members in Calc Script Editor:

1. In the Dimensions list, select the dimension for which you want to display all the members.

   For example, select the Product dimension in the Sample Basic database.

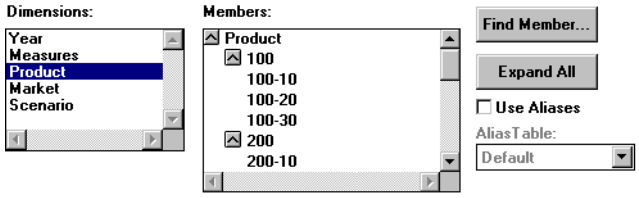


Figure 31-39: Expanding a Dimension

2. Click Expand All.

   In the Members list, Hyperion Essbase displays all members in the dimension.

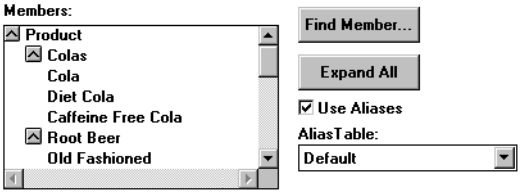


Figure 31-40: Expanding a Dimension

➤ To display and insert alias names in Calc Script Editor:

1. Check Use Aliases.

   Hyperion Essbase displays the alias names for the members. The following example shows the Product dimension from the Sample Basic database.

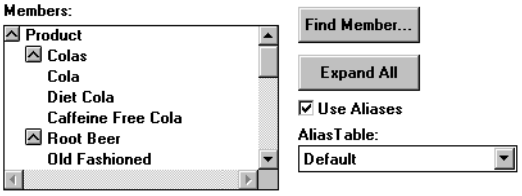


Figure 31-41: Displaying and Inserting Alias Names

2. To select a different alias table, from the Alias Table list box, select the table.

   When you select a member from the Members list, Hyperion Essbase inserts the alias name at the cursor position. If required, Hyperion Essbase automatically encloses the alias name in double quotation marks ("").

Checking Syntax

Hyperion Essbase includes a syntax checker that tells you about any syntax errors in a calc script. For example, Hyperion Essbase tells you if you have mistyped a function name.

The syntax checker cannot tell you about semantic errors in a calc script. Semantic errors occur when a calc script does not work as you expect. To find semantic errors, always run the calculation, and check the results to ensure they are as you expect.

➤ To check the syntax of a calc script in Calc Script Editor:

Select Syntax > Check or click the  button.

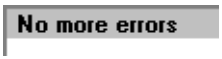
Hyperion Essbase displays the syntax checker results at the bottom of the Calc Script Editor window. If Hyperion Essbase finds no syntax errors, it displays the following message:

If Hyperion Essbase finds one or more syntax errors, it displays the number of the line that includes the error and a brief description of the error. For example, if you do not include a semicolon end-of-line character at the end of a calc script command, Hyperion Essbase displays a message similar to the following:

➤ To step through syntax errors in Calc Script Editor:

Select Syntax > Next Error or Syntax > Previous Error.

When you reach the first or last error, Hyperion Essbase displays the message:



Hyperion Essbase maintains the list of error messages until you check the syntax again.

Using a Calc Script to Control Intelligent Calculation

Assume that you have a formula on a sparse dimension member and the formula contains either of the following:

- A relationship function (for example, @PRIOR or @NEXT)
- A financial function (for example, @NPV or @INTEREST)

Hyperion Essbase always recalculates the data block that contains the formula, even if the data block is marked as clean for the purposes of Intelligent Calculation. For more information, see Chapter 34, “Using Intelligent Calculation to Optimize Calculation.”

Grouping Formulas and Calculations

You may achieve significant calculation performance improvements by carefully grouping formulas and dimensions in a calc script. For more information and examples, see “Calculating a Series of Member Formulas” on page 31-51 and “Calculating a Series of Dimensions” on page 31-51.

When you run a calc script, Hyperion Essbase automatically displays the calculation order of the dimensions for each pass through the database. Thus, you can tell how many times Hyperion Essbase has cycled through the database during the calculation.

Hyperion Essbase displays these information messages in the ESSCMD window and in the event log file. To display the event log file, select Application > View Event Log from the Hyperion Essbase Application Manager menu.

Calculating a Series of Member Formulas

When you calculate formulas, avoid using parentheses unnecessarily. The following formulas cause Hyperion Essbase to cycle through the database once, calculating both formulas in one pass:

```
Profit = (Sales - COGS) * 1.5;
Market = East + West;
```

Similarly, the following configurations cause Hyperion Essbase to cycle through the database only once, calculating the formulas on the members Qtr1, Qtr2, and Qtr3:

```
Qtr1;
Qtr2;
Qtr3;
```

or

```
(Qtr1;
Qtr2;
Qtr3;)
```

However, the inappropriately placed parentheses in the following example cause Hyperion Essbase to cycle through the database twice, once calculating the formulas on the members Qtr1 and Qtr2 and once calculating the formula on Qtr3:

```
(Qtr1;
Qtr2;)
Qtr3;
```

Calculating a Series of Dimensions

When you calculate a series of dimensions, you can optimize performance by grouping the dimensions wherever possible.

For example, the following formula causes Hyperion Essbase to cycle through the database only once:

```
CALC DIM(Year, Measures);
```

However, the following syntax causes Hyperion Essbase to cycle through the database twice. It cycles through once for each CALC DIM command:

```
CALC DIM(Year);
CALC DIM(Measures);
```

Using Substitution Variables

You can use substitution variables in calc scripts. Substitution variables are useful, for example, when you reference information or lists of members that change frequently.

When you include a substitution variable in a calc script, Hyperion Essbase replaces the substitution variable with the value you specified for the substitution variable.

You create and specify values for substitution values in Hyperion Essbase Application Manager. For more information, see Chapter 7, “Creating Applications and Databases.”

You can create variables at the server, application, and database levels. 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 ampersand (&) character prefaces a substitution variable in a calc script. Hyperion Essbase treats any string that begins with a leading ampersand as a substitution variable, replacing the variable with its value before parsing the calc script.

For example, `&CurQtr;` becomes `Qtr1;` if you have given the substitution variable `&CurQtr` the value `Qtr1`.

Consider an example in which you want to calculate Sample Basic data for the current quarter. You can use the following calc script:

```
FIX(&CurQtr)
CALC DIM(Measures, Product);
ENDFIX
```

You then define the substitution variable `CurQtr` as the current quarter; for example, Qtr3. Hyperion Essbase replaces the variable CurQtr with the value Qtr3 when it runs the calc script.

Clearing Data

You can use the CLEARDATA and CLEARBLOCK calculation commands to remove data values and data blocks from a database. You can use the CLEARBLOCK DYNAMIC command to remove blocks for Dynamic Calc And Store member combinations. For more information, see Chapter 29, “Dynamically Calculating Data Values.”

When you use the CLEARBLOCK command, Hyperion Essbase removes the entire contents of a block, including all the dense dimension members. Hyperion 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, the following example removes all the data blocks that do not contain input data values. Hyperion Essbase ignores the FIX command:

```
FIX(Actual)
CLEARBLOCK NONINPUT;
ENDFIX
```

If the Scenario dimension is sparse, the following formula removes only the blocks whose Scenario dimension member is Actual. The other blocks remain:

```
FIX(Actual)
CLEARBLOCK NONINPUT;
ENDFIX
```

When you use the CLEARDATA command, Hyperion Essbase changes the values of the cells you specify to `#MISSING`. The data blocks are not removed unless all cells in a block are cleared (that is, all cells in the block are set to `#MISSING`).

For example, the following formula clears all the Actual data values for Colas:

```
CLEARDATA Actual->Colas;
```

You can use the FIX command with the CLEARDATA command to clear a subset of a database. If you want to clear an entire database, you can select the Clear Data command from the Database menu in Hyperion Essbase Application Manager.

For more information on the CLEARBLOCK and CLEARDATA calculation commands, see the online *Technical Reference* in the `DOCS` directory.

Copying Data

You can use the DATACOPY calculation command to copy data cells from one range to another range in a database. The two ranges must be the same size.

For example, in the Sample Basic database, the following formula copies Actual values to Budget values:

```
DATACOPY Actual TO Budget;
```

You can use the FIX command to copy a subset of values.

For more information on the FIX command, see the online *Technical Reference* in the `DOCS` directory.

Calculating a Subset of a Database

You can calculate a subset of a database, which means that you can use different formulas to calculate separate sections of a database.

To calculate a subset of a database, you can use either of the following:

- Member set functions to calculate lists of members
- The FIX … ENDFIX commands to calculate a range of values

For more information, see “Calculating Lists of Members” on page 31-55 and “Using the FIX Command” on page 31-55.

Note: When you have Intelligent Calculation turned on, the newly calculated data blocks are not marked as clean after a partial calculation of a database. When you calculate a subset of a database, you can ensure that the newly calculated blocks are marked as clean using the SET CLEARUPDATESTATUS AFTER command. This ensures that Hyperion Essbase recalculates the database as efficiently as possible using Intelligent Calculation. For more information on Intelligent Calculation, see Chapter 34, “Using Intelligent Calculation to Optimize Calculation.” For more information on the SET CLEARUPDATESTATUS command, see the online *Technical Reference* in the `DOCS` directory.

Calculating Lists of Members

You can use a member set function to generate a list of members that is based on a member you specify. For example, you can use the @IDESCENDANTS function to generate a list of all the descendants of a specified member.

In the Sample Basic database, `@IDESCENDANTS("Total Expenses");` generates the following list of members: Total Expenses, Marketing, Payroll, and Misc.

When you use a member set function in a formula, Hyperion Essbase generates a list of members before calculating the formula.

For detailed information on these and other member set functions, see the online *Technical Reference* in the `DOCS` directory.

Using the FIX Command

The FIX … ENDFIX commands are particularly useful to calculate a carefully defined subset of the values in a database. For example, the following calc script calculates only the Budget values for only the descendants of East (New York, Massachusetts, Florida, Connecticut, and New Hampshire) in the Sample Basic database:

```
FIX(Budget,@DESCENDANTS(East))
CALC DIM(Year, Measures, Product);
ENDFIX
```

The next example fixes on member combinations for the children of East that have a user-defined attribute (UDA) of New Mkt. For information on defining user-defined attributes, see Chapter 8, “Creating and Changing Database Outlines.”

```
FIX(@CHILDREN(East) AND @UDA(Market,"New Mkt"))
Marketing = Marketing * 1.1;
ENDFIX
```

The next example uses a wildcard match to fix on member names that end in the characters -10. In Sample Basic, this example fixes on the members 100-10, 200-10, 300-10, and 400-10.

```
FIX(@MATCH(Product, "???-10"))
Price = Price * 1.1;
ENDFIX
```

When you use the FIX command *only* on a dense dimension, Hyperion Essbase retrieves the entire block that contains the required value or values for the member or members that you specify. Thus, I/O is not affected, and the calculation performance time is improved.

When you use the FIX command on a sparse dimension, Hyperion Essbase retrieves the block for the specified sparse dimension member or members. Thus, I/O may be greatly reduced.

Hyperion Essbase cycles through the database once for each FIX command that you use on dense dimension members. When possible, combine FIX blocks to improve calculation performance. For example, the following calc script causes Hyperion Essbase to cycle through the database only once, calculating both the Actual and the Budget values:

```
FIX(Actual,Budget)
CALC DIM(Year, Measures);
ENDFIX
```

However, this calc script causes Hyperion Essbase to cycle through the database twice, once calculating the Actual data values and once calculating the data values for Budget:

```
FIX(Actual)
CALC DIM(Year, Measures);
ENDFIX
FIX(Budget)
CALC DIM(Year, Measures);
ENDFIX
```

You cannot FIX on a subset of a dimension that you calculate within a FIX statement. For example, the following calc script returns an error message because the CALC DIM operation calculates the entire Market dimension, although the FIX above it fixes on specific members of the Market dimension.

```
FIX(@CHILDREN(East) AND @UDA(Market,"New Mkt"))
CALC DIM(Year, Measures, Product, Market);
ENDFIX
```

For detailed information on using the FIX command, see the online *Technical Reference* in the `DOCS` directory.

Writing Calc Scripts for 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 16, “Building and Maintaining Partitions.”

You can achieve significant calculation performance improvements by partitioning applications and running separate calculations on each partition.

However, when you use partitioning, you need to do both of the following:

- Consider carefully the performance impact on the overall database calculation. You might choose to do any of the following:
  - Redesign the overall calculation to avoid referencing remote values that are in a transparent partition in a remote database.
  - Dynamically calculate a value in a remote database. See Chapter 29, “Dynamically Calculating Data Values.”
  - Replicate a value in the database that contains the applicable 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 a referenced value is up-to-date when Hyperion Essbase retrieves it. Choose one of the options previously discussed (redesign, dynamically calculate, or replicate) or calculate the referenced database before calculating the formula.

Calculating Multiple Databases

You need to calculate databases in a specific order to ensure that Hyperion Essbase calculates the required results. For example, consider the following partitions in which you view information from the West, Central, and East databases transparently from the Corporate database.

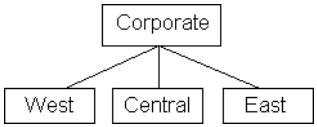


Figure 31-42: Calculating Partitions

West, Central, and East contain only actual values. Corporate contains actual and budgeted values. Although you can view the West, Central, and East data in the Corporate database, the data exists only in the West, Central, and East databases; it is not duplicated in the Corporate database.

Therefore, when Hyperion Essbase calculates Corporate, it needs to take the latest values from West, Central, and East. To obtain the required results, you need to calculate West, Central, and East before you calculate Corporate.

Chapter 32

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:

For more examples that use the Intelligent Calculation commands SET UPDATECALC and SET CLEARUPDATESTATUS in calc scripts, see Chapter 34, "Using Intelligent Calculation to Optimize Calculation."

Calculating Variance

The Sample Basic database includes a calculation of the percentage of variance between Budget and Actual values.

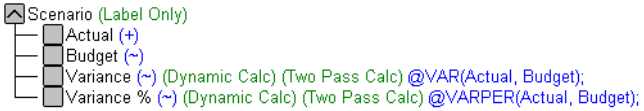


Figure 32-1: Calculating Variance and Variance %

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 Hyperion Essbase dynamically calculates Variance % values when you retrieve them. This calculation overwrites the incorrect values with the correctly calculated percentages. If you choose not to tag Variance % as a Dynamic Calc, two-pass member, you could use the following calc script to recalculate the percentages. For more information on dynamic calc members, see Chapter 29, “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 %";
```

Hyperion Essbase performs the following calculations:

1. Hyperion Essbase uses the CALC ALL command to perform a default calculation of the database. Alternatively, you could run a default calculation of the database outline without using a calc script.
2. The SET UPDATECALC OFF command turns off Intelligent Calculation.
3. The CLEARUPDATESTATUS AFTER command tells Hyperion Essbase to mark the calculated blocks as clean, even though this is a partial calculation of the database (by default, data blocks are marked as clean only after a full calculation of the database).
4. Hyperion Essbase cycles through the database calculating the formula for Variance %.

For information on calculating *statistical* variance, see the online *Technical Reference* in the `DOCS` directory.

For more information on using a calc script for two-pass calculations, see Chapter 33, “Optimizing Calculations.” For more information on developing formulas, see Chapter 26, “Developing Formulas.”

Calculating a Subset of a Database

This example is based on the Sample Basic database. The Marketing managers of each of the regions East, West, South, and Central need to calculate their corresponding areas of the database.

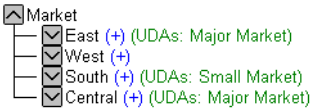


Figure 32-2: Market Dimension from the Sample Basic Database

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 for the descendants of East */
FIX(Budget, @DESCENDANTS(East))
CALC DIM(Year, Measures, Product);
ENDFIX
/* Consolidate East */
FIX(Budget)
@DESCENDANTS(East);
ENDFIX
```

Hyperion Essbase performs the following calculations:

1. Hyperion Essbase fixes on the Budget values for the descendants of East.
2. The Year, Measures, and Products dimensions are calculated in one pass of the database for each of the Budget values of the descendants of East.
3. Hyperion Essbase fixes on the Budget values for all members on the other dimensions.
4. Hyperion Essbase aggregates the descendants of East and places the result in East.

The following three calc scripts are used by the marketing managers of the other regions:

```
/* Calculate the Budget data values for the descendants of West */
FIX(Budget, @DESCENDANTS(West))
CALC DIM(Year, Measures, Product);
ENDFIX
/* Consolidate West */
FIX(Budget)
@DESCENDANTS(West);
ENDFIX

/* Calculate the Budget data values for the descendants of South
*/
FIX(Budget, @DESCENDANTS(South))
CALC DIM(Year, Measures, Product);
ENDFIX
/* Consolidate South */
FIX(Budget)
@DESCENDANTS(South);
ENDFIX

/* Calculate the Budget data values for the descendants of
Central */
FIX(Budget, @DESCENDANTS(Central))
```

```
CALC DIM(Year, Measures, Product);
ENDFIX
/* Consolidate Central */
FIX(Budget)
@DESCENDANTS(Central);
ENDFIX
```


Loading New Budget Values

The following example loads budget values into the Sample Basic database and recalculates the database:

```
/* Recalculate all Budget values */
FIX(Budget)
CALC DIM(Year, Product, Market, Measures);
ENDFIX

/* Recalculate the Variance and Variance % formulas, which
      require two passes */
Variance;
"Variance %";
```

Hyperion Essbase performs the following calculations:

1. Hyperion Essbase fixes on the Budget values.
2. Hyperion Essbase calculates all Budget values. The CALC DIM command is used to calculate all the dimensions except for the Scenario dimension, which contains Budget.
3. Hyperion Essbase calculates the formula applied to Variance in the database outline.
4. Hyperion Essbase calculates the formula applied to Variance % in the database outline.

Calculating Product and Market Share Values

The following example is based on the Sample Basic database. It calculates product share and market share values for each market and each product.

The product and market share values are calculated based on:

- Each member as a percentage of the total.
- Each member as a percentage of its parent.

Assume that you add four members to the Measures dimension: Market Share, Product Share, Market %, and Product %.

```
/* First consolidate the Sales values to ensure that they are
accurate */
FIX(Sales)
CALC DIM(Year, Market, Product);
ENDFIX

/* Calculate each market as a percentage of the total market for
each product */
"Market Share" = Sales % Sales->Market;

/* Calculate each product as a percentage of the total product
for each market */
"Product Share" = Sales % Sales->Product;

/* Calculate each market as a percentage of its parent for each
product */
"Market %" = Sales % @PARENTVAL(Market, Sales);

/* Calculate each product as a percentage its parent for each
market */
"Product %" = Sales % @PARENTVAL(Product, Sales);
```

Hyperion Essbase performs the following calculations:

1. Hyperion Essbase fixes on the Sales values and consolidates all the Sales values. The CALC DIM command is used to calculate the Year, Market, and Product dimensions. The Measures dimension contains the Sales member and therefore is not consolidated. The Scenario dimension is label only and therefore does not need to be consolidated.
2. Hyperion Essbase cycles through the database and calculates Market Share. It takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of total Sales in all markets for each product (Sales->Market).
3. Hyperion Essbase calculates Product Share. It takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of total Sales of all products in each market (Sales->Product).
4. Hyperion Essbase calculates Market %. It takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of the Sales value of the parent of the current member on the Market dimension. It uses the @PARENTVAL function to obtain the Sales value of the parent on the Market dimension.
5. Hyperion Essbase calculates Market %. It takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of the Sales value of the parent of the current member on the Product dimension. It uses the @PARENTVAL function to obtain the Sales value of the parent on the Product dimension.

Allocating Costs Across Products

The following example is based on the Sample Basic database. It allocates overhead costs to each product in each market for each month.

The overhead costs are allocated based on each product's Sales value as a percentage of the total Sales for all products.

Assume that you add two members to the Measures dimension: OH_Costs for the allocated overhead costs and OH_TotalCost for the total overhead costs.

```
/* Declare a temporary array called ALLOCQ based on the Year
dimension */
ARRAY ALLOCQ[Year];

/*Turn the Aggregate Missing Values setting off. If this is your
system default, omit this line */
SET AGGMISSG OFF;

/* Allocate the overhead costs for Actual values */
FIX(Actual)
OH_Costs (ALLOCQ=Sales/Sales->Product; OH_Costs =
OH_TotalCost->Product * ALLOCQ;);

/* Calculate and consolidate the Measures dimension */
CALC DIM(Measures);
ENDFIX
```

Hyperion Essbase performs the following calculations:

1. Hyperion Essbase creates a one-dimensional array called ALLOCQ. The size of ALLOCQ is based on the number of members in the Year dimension. Hyperion Essbase uses ALLOCQ to store the value of Sales as a percentage of total Sales temporarily for each member combination.
2. The `SET AGGMISSG OFF;` command means that `#MISSING` values are not aggregated to their parents. Data values stored at parent levels are not overwritten. If this is your system default, you can omit this line. For more information on setting the default for aggregating `#MISSING` values, see Chapter 33, “Optimizing Calculations.”
    - Hyperion Essbase fixes on the Actual values.
    - Hyperion Essbase cycles through the member combinations for Actual and calculates OH_Costs.

- It then takes the Sales value for each product in each market for each month. It calculates this Sales value as a percentage of total Sales for all products in each market (Sales->Product). It places the result in ALLOCQ.
- It then takes the total overhead costs for all products (OH_TotalCost->Product) and multiplies it by the value it has just placed in ALLOCQ. It places the result in OH_Costs.

Notice that both of the equations are enclosed in parentheses ( ) and associated with the OH_Costs member, OH_Costs(equation1; equation2;). For more information, see Chapter 31, "Developing Calc Scripts."

3. Hyperion Essbase calculates and consolidates the Measures dimension.

Allocating Values Within or Across Dimensions

Using the @ALLOCATE and @MDALLOCATE functions, you can allocate values to members in the same dimension or to members in multiple dimensions.

Allocating Within a Dimension

The following example uses the @ALLOCATE function to allocate budgeted total expenses across expense categories for two products. The budgeted total expenses are allocated based on the prior year's actual values.

Note: For more information on the @ALLOCATE function, see the online *Technical Reference* in the `DOCS` directory.

The following example is based on the Sample Basic database. Assume that you have made the following changes to Sample Basic:

- Added a child, Lease, under Total Expenses in the Measures dimension
- Added a child, PY Actual, to the Scenario dimension

- Removed the Dynamic Calc tag from the Total Expenses member

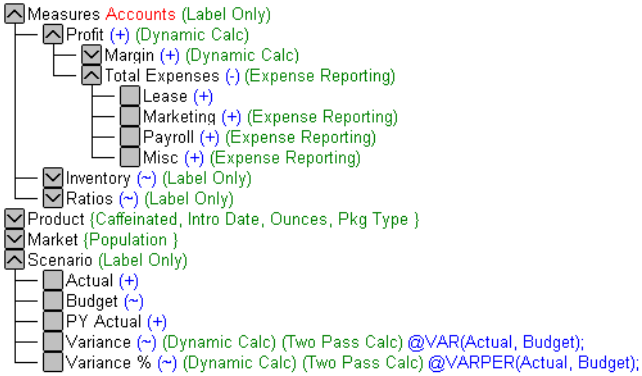


Figure 32-3: Modified Measures and Scenario Dimensions from the Sample Basic Database

For this example, assume that data values of 1000 and 2000 are loaded into Budget->Total Expenses for Colas and Root Beer, respectively. These values need to be allocated to each expense category, evenly spreading the values based on the non-missing children of Total Expenses from PY Actual. The allocated values need to be rounded to the nearest dollar.

The following calc script defines the allocation:

```
/* Allocate budgeted total expenses based on prior year */

FIX("Total Expenses")
Budget = @ALLOCATE(Budget->"Total Expenses",
  @CHILDREN("Total Expenses"),"PY Actual",,
    spread,SKIPMISSING,roundAmt,0,errorsToHigh)
ENDFIX
```

The following table shows the results:

		Budget	PY Actual
Colas	Marketing	334*	150
	Payroll	#MI	#MI
	Lease	333	200
	Misc	333	100
	Total Expenses	**1000**	**450**
Root Beer	Marketing	500	300
	Payroll	500	200
	Lease	500	200
	Misc	500	400
	Total Expenses	**2000**	**1100**

* Rounding errors are added to this value. See Step 5 for more information.

Hyperion Essbase cycles through the database, performing the following calculations:

1. Hyperion Essbase fixes on the children of Total Expenses. Using a FIX statement with @ALLOCATE may improve calculation performance.
2. For Budget->Colas->Marketing, Hyperion Essbase divides 1 by the count of non-missing values for each expense category in PY Actual->Colas for each month. In this case, 1 is divided by 3, because there are 3 non-missing expense values for Budget->Colas.
3. Hyperion Essbase takes the value from step 2 (.333), multiplies it by the value for Budget->Colas->Total Expenses (1000), and then rounds to the nearest dollar (333). This value is placed in Budget->Colas->Marketing.
4. Hyperion Essbase repeats steps 2–3 for each expense category for Budget->Colas and then for Budget->Root Beer.
5. As specified in the calc script, the allocated values are rounded to the nearest whole dollar. Hyperion Essbase makes a second pass through the block to make the sum of the rounded values equal to the allocation value (for example, 1000 for Budget->Colas->Total Expenses). In this example, there is a rounding error of 1 for Budget->Colas->Total Expenses, because the expense categories add up to 999, not 1000, which is the allocation value. Because all the allocated values are identical (333), the rounding error of 1 is added to the first value in the allocation range, Budget->Colas->Marketing (thus a value of 334).

Note: For another example of the @ALLOCATE function, see the online *Technical Reference* in the `DOCS` directory.

Allocating Across Multiple Dimensions

The following example uses the @MDALLOCATE function to allocate a loaded value for budgeted total expenses across three dimensions. The budgeted total expenses are allocated based on the prior year's actual values.

Note: For complete information on the @MDALLOCATE function, see the online *Technical Reference* in the `DOCS` directory.

The following example is based on the Sample Basic database. Assume that you have made the following modifications:

- Added a child, PY Actual, to the Scenario dimension
- Copied data from Actual into PY Actual
- Cleared data from Budget

For this example, a value of 750 (for Budget->Total Expenses->Product-> East->Jan) needs to be allocated to each expense category for the children of product 100 across the states in the East. The allocation uses values from PY Actual to determine the percentage share that each category should receive.

The following calc script defines the allocation:

```
/* Allocate budgeted total expenses based on prior year, across
3 dimensions */

SET UPDATECALC OFF;
FIX (East, “100”, “Total Expenses”)

BUDGET = @MDALLOCATE(750,3,@CHILDREN(“100”),@CHILDREN("Total
Expenses"),@CHILDREN(East),"PY Actual",,share);
ENDFIX
```

The following table shows the values for PY Actual:

		Jan			
		PY Actual			
		Marketing	Payroll	Misc	Total Expenses
100-10	New York	94	51	0	145
	Massachusetts	23	31	1	55
	Florida	27	31	0	58
	Connecticut	40	31	0	71
	New Hampshire	15	31	1	47
100-20	New York	199	175	2	376
	Massachusetts	#MI	#MI	#MI	#MI
	Florida	#MI	#MI	#MI	#MI
	Connecticut	26	23	0	49
	New Hampshire	#MI	#MI	#MI	#MI
100-30	New York	#MI	#MI	#MI	#MI
	Massachusetts	26	23	0	49
	Florida	#MI	#MI	#MI	#MI
	Connecticut	#MI	#MI	#MI	#MI
	New Hampshire	#MI	#MI	#MI	#MI
100	New York	#MI	#MI	#MI	#MI
	Massachusetts	12	22	1	35
	Florida	12	22	1	35
	Connecticut	94	51	0	145
	New Hampshire	23	31	1	55
	East	237	220	3	460

Hyperion Essbase cycles through the database, performing the following calculations:

1. Hyperion Essbase fixes on East, the children of 100, and Total Expenses. Using a FIX statement with @MDALLOCATE may improve calculation performance.
2. Before performing the allocation, Hyperion Essbase needs to determine what share of 750 (the value to be allocated) each expense category should receive, for each product-state combination. To do this, Hyperion Essbase uses the shares of each expense category from PY Actual. Starting with PY Actual-> 100-10->New York, Hyperion Essbase divides the value for the first expense category, Marketing, by the value for PY Actual->100-10->East-> Total Expenses to calculate the percentage share of that category. For example, Hyperion Essbase divides the value for PY Actual->100-10->New York-> Marketing (94) by the value for PY Actual->100-10->East->Total Expenses (460), which yields a percentage share of approximately 20.4% for the Marketing category.
3. Hyperion Essbase repeats step 2 for each expense category, for each product-state combination.
4. During the allocation, Hyperion Essbase uses the percentage shares calculated in steps 2–3 to determine what share of 750 should be allocated to each child of Total Expenses from Budget, for each product-state combination. For example, for Marketing, Hyperion Essbase uses the 20.4% figure calculated in step 2, takes 20.4% of 750 (approximately 153), and places the allocated value in Budget->100-10->New York->Marketing (see table below).
5. Hyperion Essbase repeats step 4 for each expense category and for each product-state combination, using the percentage shares from PY Actual calculated in steps 2–3.
6. Hyperion Essbase consolidates the expense categories to yield the values for Total Expenses.

The following table shows the results of the allocation for Budget:

		Jan Budget			
		Marketing	**Payroll**	**Misc**	**Total Expenses**
100-10	New York	153.26	83.15	0	236.41
	Massachusetts	37.50	50.54	1.63	89.67
	Florida	44.02	50.54	0	94.56
	Connecticut	65.22	50.54	0	115.76
	New Hampshire	24.46	50.54	1.63	76.63
100-20	New York	#MI	#MI	#MI	#MI
	Massachusetts	#MI	#MI	#MI	#MI
	Florida	42.39	37.50	0	79.89
	Connecticut	#MI	#MI	#MI	#MI
	New Hampshire	#MI	#MI	#MI	#MI
100-30	New York	#MI	#MI	#MI	#MI
	Massachusetts	#MI	#MI	#MI	#MI
	Florida	#MI	#MI	#MI	#MI
	Connecticut	#MI	#MI	#MI	#MI
	New Hampshire	19.57	35.87	1.63	57.07
100	New York	153.26	83.15	0	236.41
	Massachusetts	37.50	50.54	1.63	89.67
	Florida	86.41	88.04	0	174.46
	Connecticut	65.22	50.54	0	115.76
	New Hampshire	44.02	86.41	3.26	133.70
	East	386.41	358.70	4.89	750

Note: For another example of the @MDALLOCATE function, see the online *Technical Reference* in the `DOCS` directory.

Goal Seeking Using the LOOP Command

The following example is based on the Sample Basic database. However, the example assumes that no members are tagged as Dynamic Calc, and that the Profit per Ounce member (under Ratios in the Scenario dimension) is not included in the calculation. For more information on Dynamic Calc members, see Chapter 29, “Dynamically Calculating Data Values.”

You want to know what sales value you have to reach in order to obtain a certain profit on a specific product.

This example adjusts the Budget value of Sales to reach a goal of 15,000 Profit for Jan. The results are shown for product 100-10.

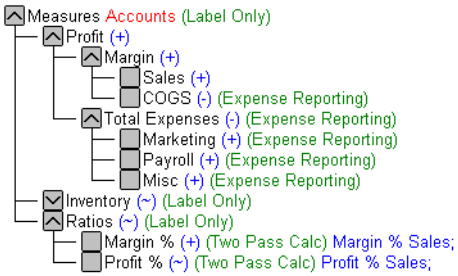


Figure 32-4: Measures Dimension from the Sample Basic Database

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
```

Hyperion Essbase performs the following calculations:

1. It declares the required temporary variables using the VAR command. Where appropriate, the initial values are set.
2. Hyperion Essbase declares a one-dimensional array called Rollback. The size of Rollback is based on the number of members in the Measures dimension. Hyperion Essbase uses Rollback to store the Budget values.
3. Hyperion Essbase fixes on the Jan->Budget values for all Product and Market members.
4. The LOOP command ensures that the commands between LOOP and ENDLOOP are cycled through 35 times *for each member combination*. However, if the Quit variable is set to 1, then the LOOP is broken and the calculation continues after the ENDLOOP command.

5. Hyperion Essbase cycles through the member combinations, performing the following calculations:

   a. Hyperion Essbase places the Budget->Sales value in the Rollback temporary array variable.

   b. It calculates the acceptable error. It multiplies the Target value (15000) by the AcceptableErrorPercent value (0.001) and places the result in the AcceptableError variable.

   c. It retains the current Sales value. It places the Sales value for the current member combination in the PriorVar temporary variable.

   d. It retains the current Profit value. It places the Profit value for the current member combination in the PriorTar temporary variable.

   e. It calculates a new Sales value. It multiplies the PctNewVarChange value (0.1) by the current Sales value, adds the current Sales value, and places the result in Sales.

   f. Hyperion Essbase calculates and consolidates the Measures dimension.

   g. It subtracts the PriorTar value from the PriorVar value and places the result in the DependencyCheck temporary variable.

   h. The IF command checks that DependencyCheck is not 0 (zero).

   i. If DependencyCheck is not 0, then Hyperion Essbase subtracts the Target value (15000) from the current Profit and places the result in the CurTarDiff temporary variable.

      The IF command checks to see if the absolute value (irrespective of the + or – sign) of CurTarDiff is greater than the absolute value of the acceptable error (AcceptableError). If it is, Hyperion Essbase calculates the Slope, NxtVar, and PctNewVarChange temporary variables.

      If it is not greater than AcceptableError, Hyperion Essbase breaks the LOOP command by setting the value of Quit to 1. The calculation continues after the ENDLOOP command.

      If DependencyCheck is 0, Hyperion Essbase places the value in the Rollback array into Budget. Hyperion Essbase breaks the LOOP command by setting the value of Quit to 1. The calculation continues after the ENDLOOP command.

6. Hyperion Essbase calculates and consolidates the Measures dimension.

The results are shown in 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

Forecasting Future Values

The following example uses the @TREND function to forecast sales data for June through December, assuming that data currently exists only up to May. Using the linear regression forecasting method, this example produces a trend, or line, that starts with the known data values from selected previous months and continues with forecasted values based on the known values. In addition, this example demonstrates how to check the results of the trend for "goodness of fit" to the known data values.

Note: For more information on the @TREND function, see the online *Technical Reference* in the `DOCS` directory.

The following example is based on the Sample Basic database. Assume that the Measures dimension contains an additional child, ErrorLR. The goodness-of-fit results are placed in this member.

The following calc script defines the forecasting:

```
Sales
(@TREND(@LIST(Jan,Mar,Apr),@LIST(1,3,4),,
  @RANGE(ErrorLR,@LIST(Jan,Mar,Apr)),
    @LIST(6,7,8,9,10,11,12),
      Jun:Dec,LR););
```

The following table explains each parameter:

@LIST(Jan,Mar,Apr)	Represents the *Ylist*, or the members that contain the known data values. The @LIST function is needed to group the three members as a comma-delimited list and to keep the list separate from other parameters.
@LIST(1,3,4)	Represents the *Xlist*, or the underlying variable values. Since Feb and May are skipped, we need to number the *Ylist* values accordingly (1,3,4).
,	The extra comma after the *Xlist* parameter indicates that a parameter has been skipped, in this case, the *weightList* parameter. The default weight of 1 is used for this example.
@RANGE(ErrorLR, @LIST(Jan,Mar,Apr)	Represents the *errorList*, or the member list where results of the goodness of fit of the trend line to *Ylist* are placed. The values placed in *errorList* are the differences between the data points in *Ylist* and the data points on the trend line produced. The @RANGE function combines the ErrorLR member with *Ylist* (Jan, Mar, Apr) to produce a member list.
@LIST(6,7,8,9,10,11,12)	Represents the *XforecastList*, or the underlying variable values for which the forecast is sought. This example forecasts values consecutively for Jun through Dec, so the values are simply 6,7,8,9,10,11,12.

Jun:Dec	Represents the *YforecastList*, or the member list into which the forecast values are placed. In this example, values are forecast for Jun through Dec based on the values for Jan, Mar, and Apr.
LR	Specifies the Linear Regression method.

Note: For more information on the @LIST and @RANGE functions, see the online *Technical Reference* in the `DOCS` directory.

The following table shows the results of the calc script:

	100 West Actual	
	Sales	**ErrorLR**
Jan	2339	4.57
Feb	2298	#MI
Mar	2313	-13.71
Apr	2332	9.14
May	2351	#MI
Jun	2315.14	#MI
Jul	2311.29	#MI
Aug	2307.49	#MI
Sep	2303.57	#MI
Oct	2299.71	#MI
Nov	2295.86	#MI
Dec	2292	#MI

Hyperion Essbase cycles through the database, performing the following calculations:

1. Hyperion Essbase finds the known data values on which to base the trend (Sales for Jan, Mar, Apr), as specified for the *Ylist* and *Xlist* parameters in the calc script.
2. Hyperion Essbase calculates the trend line using Linear Regression and places the results in Sales for Jun through Dec, as specified for the *YforecastList* parameter in the calc script.
3. Hyperion Essbase calculates the goodness of fit of the trend line to the data values for Jan, Mar, and Apr and places the results in ErrorLR for those months. For example, the value in ErrorLR for Jan (4.57) means that after Hyperion Essbase calculates the trend line, the difference between the Sales value for Jan (2339) and the Jan value on the trend line is 4.57. The ErrorLR values for Feb and May are `#MISSING` since these months were not part of *Ylist*.

Note: For another example of the @TREND function, see the online *Technical Reference* in the `DOCS` directory.

Chapter 33

Optimizing Calculations

This chapter provides information on how to optimize the performance of Hyperion Essbase calculations. The following three features are not discussed in this chapter. However, you can use any or all of these features to optimize overall database calculations:

- Intelligent Calculation, as discussed in Chapter 34, “Using Intelligent Calculation to Optimize Calculation”
- Dynamic calculations, as discussed in Chapter 29, “Dynamically Calculating Data Values”
- Partitioning, as discussed in Chapter 6, “Designing Partitioned Applications” and Chapter 31, “Developing Calc Scripts”

This chapter includes the following sections:

Designing for Calculation Performance

You can configure a database to optimize calculation performance. The optimal configuration is highly dependent on the nature and size of the database. The following sections provide guidelines only.

Block Size and Block Density

Generally speaking, you do not want data block size to be too small. A data block size of 8K to 64K provides optimal performance in most cases.

If data blocks are very small, the index is likely to be very large. This configuration causes a performance overhead as Hyperion Essbase writes and retrieves the index from disk. However, if data blocks are too large, Intelligent Calculation does not work effectively. For more information on Intelligent Calculation, see Chapter 34, “Using Intelligent Calculation to Optimize Calculation.”

To optimize calculation performance and data storage, you need to balance data block density and data block size. You can create balance by rearranging the dense and sparse dimension configuration of the database. Therefore, keep these suggestions in mind:

- 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 a database, including the potential and actual number of data blocks and the data block size. In Hyperion Essbase Application Manager, select Database > Information to display the Database Information dialog box. Select the Statistics tab to display data block statistics.

You can also use the GETDBINFO command in ESSCMD to view database information. For information about the GETDBINFO command, see the online *Technical Reference* in the `DOCS` directory. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Order of Sparse Dimensions

You may achieve improvement in calculation performance by changing the order of the sparse dimensions in the database outline. Order the sparse dimensions by their 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 the database outline. This arrangement 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 Memory Cache Sizes" on page 33-14.

Incremental Data Loading Considerations

Many people load data incrementally. For example, they load data on a month-by-month basis. Each month they load data for that month.

To optimize calculation performance when you load data incrementally, make the dimension tagged as time a sparse dimension. If the time dimension is sparse, the data for each time period is contained in a separate data block. When you load data, you are loading it into only the required data blocks. Thus, if you have Intelligent Calculation enabled, only the data blocks marked as dirty are recalculated.

For example, if you load data for March, only the data blocks for March are updated. The data blocks for January and February do not change. With Intelligent Calculation enabled, Hyperion Essbase recalculates only the data blocks for March and its dependent parents.

Note: Making the time dimension sparse when it is naturally dense may significantly increase the size of the index.

If the dimension tagged as time is dense, you still receive some benefit from Intelligent Calculation when you do a partial data load for a sparse dimension. For example, if Product is sparse and you load data for one product, Hyperion Essbase recalculates only the blocks affected by the partial load, even though time is dense and Intelligent Calculation is enabled.

For more information on Intelligent Calculation, see Chapter 34, "Using Intelligent Calculation to Optimize Calculation."

Performance for Database Outlines with Two or More Flat Dimensions

If a database outline has two or more flat dimensions, calculation performance may be affected. A *flat dimension* is a dimension in which there are very few parents, and in which each parent has many (thousands) children.

If a database has two or more flat dimensions, you can achieve a performance improvement by doing one of the following:

- Adding intermediate levels to the database outline.
- Using the SET CALCHASHTBL command in a calc script to optimize calculation of the outline.

  You can specify the default setting for optimizing outline calculation in the configuration (`ESSBASE.CFG`) file by 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 the `DOCS` directory.

Monitoring and Tracing Calculations

You can display information about how Hyperion Essbase is calculating the database by using the following commands and settings in a calc script:

- SET MSG SUMMARY
- SET MSG DETAIL
- SET NOTICE

SET MSG SUMMARY and SET MSG DETAIL

You can use the SET MSG SUMMARY and SET MSG DETAIL calculation commands in a calc script to do the following:

- Display calculation settings (for example, whether completion notice messages are enabled)
- Provide statistics on the number of data blocks created, read, and written
- Provide statistics on the number of data cells calculated

In addition, the SET MSG DETAIL command provides a detailed information message every time Hyperion Essbase calculates a data block. SET MSG DETAIL is useful for seeing the calculation order of data blocks and for testing intelligent recalculations. Keep in mind that the SET MSG DETAIL command causes a high processing overhead.

SET MSG SUMMARY causes a processing overhead of approximately 1% to 5%, depending on database size.

For more information on SET MSG SUMMARY and SET MSG DETAIL, see the online *Technical Reference* in the `DOCS` directory.

SET NOTICE

You can use the SET NOTICE calculation command in a calc script to display calculation completion notices that tell you what percentage of the database has been calculated. You can use the SET MSG SUMMARY command with the SET NOTICE command to show calculation progress between completion notices. Completion notices do not significantly reduce calculation performance, except when used with a very small database.

For more information on this calculation command, see the online *Technical Reference* in the `DOCS` directory.

Using Formulas

You may achieve significant improvements in calculation performance by careful use of formulas in the database outline. For example, you may achieve improved calculation performance by placing formulas on members in the database outline instead of placing the formulas in a calc script. For more information, see Chapter 26, “Developing Formulas.”

Consider the questions posed in the following sections to ensure that you use formulas in a way that optimizes calculation performance.

Can 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, consider the following consolidation on the Sample Basic database outline:

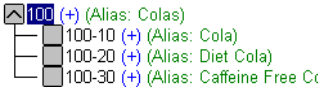


Figure 33-1: Consolidation on Sample Basic Outline

Using outline consolidation 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* do any of the following:

- Reference values from a different dimension (sparse or dense); for example, Product->Jan.
- Use range functions, for example, @AVGRANGE, @MAXRANGE, @MINRANGE, or @SUMRANGE.
- Use relationship or financial functions; for example, @ANCESTVAL, @NEXT, @PARENTVAL, @SHIFT, @ACCUM, or @GROWTH. For a complete list of relationship and financial functions, see the online *Technical Reference* in the `DOCS` directory.

If you use 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 the block size, the more impact simple formulas have on calculation performance. For more information, see “Block Size and Block Density” on page 33-2. For information on how formulas affect calculation performance, see “Understanding Bottom-Up Versus Top-Down Calculation” on page 33-12.

Are you using a complex formula?

A complex formula does any of the following:

- References a member or members in a different dimension (sparse or dense); for example, Product->Jan.
- Uses one or more range functions, for example, @AVGRANGE, @MAXRANGE, @MINRANGE, or @SUMRANGE.
- Uses relationship or financial functions; for example, @ANCESTVAL, @NEXT, @PARENTVAL, @SHIFT, @ACCUM, or @GROWTH. For a complete list of relationship and financial functions, see the online *Technical Reference* in the `DOCS` directory.

If you apply a complex formula to a member in a sparse dimension, Hyperion Essbase checks each possible sparse member combination (possible data block) for that member to see if the block exists. This evaluation causes a significant calculation overhead. If the complex formula uses relationship or financial functions, Hyperion Essbase evaluates each possible sparse member combination, which causes an even greater calculation overhead. For more information about how complex formulas affect calculation performance, see “Understanding Bottom-Up Versus Top-Down Calculation” on page 33-12.

The lower the ratio of existing data blocks to possible data blocks, the higher the calculation performance overhead.

If you use a complex formula, consider the following:

- If possible, apply the formula to a member in a *dense* dimension.
- Use the FIX command in a calc script to calculate only the minimum, required data blocks. See Chapter 31, “Developing Calc Scripts.”
- Increase the density of the database (ratio of existing data blocks to possible data blocks). See “Block Size and Block Density” on page 33-2.

Optimizing Formulas on Sparse Dimensions in Large Database Outlines

You can use the SET FRMLBOTTOMUP calculation command to optimize the calculation of formulas in sparse dimensions in large database outlines. With this command, you can force a bottom-up calculation on sparse member formulas that would otherwise be calculated top-down. For more information, see "Understanding Bottom-Up Versus Top-Down Calculation" on page 33-12.

Forcing a bottom-up calculation on a top-down formula enables efficient use of the CALC ALL and CALC DIM commands to calculate the database. For more information, see the SET FRMLBOTTOMUP calculation command and the CALCOPTFRMLBOTTOMUP configuration setting in the online *Technical Reference* in the `DOCS` directory.

Assigning Constants to Members in a Sparse Dimension

When you assign a constant to a member in a sparse dimension, Hyperion Essbase automatically creates a data block for every combination of sparse dimension members that contains the member.

For example, assume that a member or a calc script formula contains the following expression:

```
California = 120;
```

In this formula, California is a member in a sparse dimension and 120 is a constant value. Hyperion Essbase automatically creates all combinations of data blocks for California and assigns the value 120 to all data cells. Many thousands of data blocks may be created.

When assigning constants to sparse dimension members, use the FIX command to ensure that Hyperion Essbase creates only the required data blocks. Consider the following example:

```
FIX(Colas,Misc,Actual)
California = 120;
ENDFIX
```

The example script assigns the value 120 to California (in the Market dimension), Actual (in the Scenario dimension), Misc (miscellaneous expenses in the Measures dimension), Colas (in the Product dimension), and all members in the Year dimension (since a specific member of Year is not specified in the script).

In the Sample Basic database, Colas is a member of the sparse Product dimension, Actual is a member of the dense Scenario dimension, and Misc is a member of the dense Measures dimension. Hyperion Essbase creates new data blocks for all combinations of the sparse members, California and Colas. It leaves other Measures and Scenario values set to `#MISSING` within the new blocks.

For more information on the FIX command, see the online *Technical Reference* in the `DOCS` directory.

When you assign a constant to a member in a sparse dimension, you do not need to enable Create Blocks on Equations. However, if you assign anything other than a constant to a member in a sparse dimension, and a data block does not already exist for that member, you still need to enable Create Blocks on Equations in Hyperion Essbase Application Manager.

➤ To enable Create Blocks on Equations:

1. In Hyperion Essbase Application Manager, select Database > Settings.

   Hyperion Essbase displays the Database Settings dialog box.

2. Check Create Blocks on Equations.
3. Click OK.

For example, you need to enable Create Blocks on Equations for the following formula:

```
West = California + 120;
```

You also can use the SETDBSTATE command in ESSCMD to create blocks on equations. For information about the SETDBSTATE command, see the online *Technical Reference* in the `DOCS` directory. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

Using a Cross-Dimensional Operator (->)

Use caution when using a cross-dimensional operator (->) in the following situations:

- On the left side of an equation
- In equations in a dense dimension
- When doing a two-pass formula calculation on a dimension tagged as accounts. For more information, see “Using Two-Pass Calculation” on page 33-24.

On the Left Side of an Equation

In a member formula in the database outline, you can use a cross-dimensional operator on the left side of the equation. In a calc script, you can use a cross-dimensional operator on the left side of an equation if you associate the formula with a member. 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 the database outline:

```
Sales(Sales->Jan = Sales->Jan * .05;)
```

As Hyperion Essbase cycles through the database, it calculates the formula for every combination of members. Thus, Hyperion Essbase recalculates the formula for every member in the dimension tagged as time, causing redundant calculations.

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, Hyperion Essbase calculates the formula only for member combinations for Jan.

For more information on calc scripts and the FIX command, see Chapter 31, “Developing Calc Scripts” and the online *Technical Reference* in the `DOCS` directory.

In Equations in a Dense Dimension

When you use a cross-dimensional operator in an equation in a dense dimension, Hyperion 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.

Consider the following equation, in which *result* is from a dense dimension and *operand* is from a sparse dimension:

```
result = member->operand
```

Now consider a specific example from Sample Basic, in which you want to create budget sales and expense data from existing actual data. Sales and Expenses are members in the dense Measures dimension. Budget and Actual are members in the sparse Scenario dimension.

```
FIX(Budget)
      (Sales = Sales->Actual * 1.1;
       Expenses = Expenses->Actual * .95;)
ENDFIX
```

The calc script above does *not* create the required data blocks. Budget data values are not calculated for blocks that do not already exist. The resultant values are dense dimension members (Sales and Expenses). The operand is a sparse dimension member (Actual).

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
```

Hyperion Essbase then copies the data and creates the required blocks. Hyperion Essbase creates blocks that contain 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 that contains the dense member equations:

```
FIX(Sales, Expenses)
Budget (Sales = Sales->Actual * 1.1;
      Expenses = Expenses->Actual * .95;)
ENDFIX
```

Understanding Bottom-Up Versus Top-Down Calculation

Hyperion Essbase uses one of two calculation methods to do a full calculation of a database outline: bottom-up calculation or top-down calculation. By default, Hyperion Essbase does a bottom-up calculation of a database. However, if the database outline contains a complex member formula, Hyperion Essbase performs a top-down calculation for that member.

For a bottom-up calculation, Hyperion Essbase determines which data blocks need to be calculated before it calculates the database. Hyperion Essbase then calculates only the blocks that need to be calculated during the full database calculation. The calculation begins with the lowest existing block number and works up through each subsequent block until the last existing block is reached. For more information on block calculation order, see Chapter 28, “Defining the Calculation Order.”

Before starting a calculation, Hyperion Essbase searches the database outline and marks complex formulas that require a top-down calculation, for example, a member formula that contains a cross-dimensional reference. When Hyperion Essbase reaches a member with a top-down formula, it does a top-down calculation only for that member.

When a formula on a member is complex, all possible blocks for that member must be examined to see if an existing block needs to be changed or a new block needs to be created; it is difficult to determine the dependency on other blocks prior to

the start of the calculation. The top-down method slows down calculation performance because Hyperion Essbase searches for the appropriate blocks to calculate in order to execute the formula. For more information on complex formulas, see “Are you using a complex formula?” on page 33-7.

For simple formulas, Hyperion Essbase does a bottom-up calculation to determine which blocks need to be calculated prior to running the calculation. For example, for a simple formula on a member (such as `A = B + C`), A is calculated only if B or C exists in the database. That is, the dependency of the formula on B and C is known before the calculation is started.

For a complex formula (such as `A = B->D + C->D`), Hyperion Essbase must examine every possible combination of A to see whether B->D or C->D exists. Unlike a simple formula, the dependencies for a complex formula cannot be determined easily prior to the calculation.

A top-down calculation is less efficient than a bottom-up calculation because more blocks are calculated than is necessary. Although a top-down calculation is less efficient than a bottom-up calculation, top-down calculations are necessary in some cases to ensure that calculation results are correct.

You can force a bottom-up calculation on a top-down formula by using the SET FRMLBOTTOMUP command in a calc script or by using the CALCOPTFRMLBOTTOMUP configuration setting. For important information about checking calculation results when using these commands, see the online *Technical Reference* in the `DOCS` directory.

Implementing Calc Script Techniques

You may achieve significant improvements in calculation performance by carefully grouping formulas and dimensions in a calc script. In this way, you can ensure that Hyperion Essbase cycles through the data blocks in the database as few times as possible during a calculation. For more information, see Chapter 31, “Developing Calc Scripts.” For more information on calculation passes, see Chapter 28, “Defining the Calculation Order.”

Aim to make the database calculation as simple as possible. If possible, consider applying all formulas to the database outline and using a default calculation (CALC ALL). This method may improve calculation performance.

Setting Memory Cache Sizes

When calculating the database, Hyperion Essbase uses approximately 30 bytes of memory per member in the database outline. So if the database has 5,000 members, Hyperion Essbase needs approximately 150K of memory to calculate the database.

When you run concurrent calculations, each calculation uses separate memory space. For example, if you are running two calc scripts concurrently, Hyperion Essbase requires 60 bytes per member. Hyperion Essbase needs 30 bytes per member for each calculation.

Hyperion Essbase uses memory to optimize calculation performance, especially for large calculations. The amount of memory used is not controllable, except by altering the size of the database outline. However, you can ensure that the memory cache sizes enable Hyperion Essbase to optimize the calculation.

Hyperion Essbase uses four memory caches to coordinate memory usage:

- The calculator cache
- The index cache
- The data cache
- The data file cache

Ensure that the calculator cache is large enough to optimize calculation performance. For more information, see the following sections.

The index cache needs to be large enough to reduce the need to keep writing the index pages to the disk. If the database is large, the default index cache is not large enough to provide optimum calculation performance.

For information on sizing the index, data file, and data caches, see Chapter 15, “Estimating Disk and Memory Requirements for a Database.”

Using the Calculator Cache

The calculator cache creates and tracks data blocks during a calculation in order to avoid excessive disk activity. Hyperion Essbase uses the calculator cache's bitmap to determine which blocks exist rather than making a call to the Hyperion Essbase Kernel to obtain this information.

Using the calculator cache's bitmap significantly improves calculation performance, particularly if you are calculating the database for the first time or if the data in the database is very sparse. The size of the performance improvement depends on the configuration of the database. For more information on the bitmap, see "Understanding Calculator Cache Options" on page 33-16.

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 setting in the online *Technical Reference* in the `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 calculator cache size that you choose depends on the amount of memory the system has available and the configuration of the database. For information on estimating the size of the calculator cache, see "Calculating the Required Size of the Calculator Cache" on page 33-18.

Hyperion Essbase uses the calculator cache, provided that both of the following conditions are met:

- The database has at least two sparse dimensions.
- You calculate at least one, full sparse dimension (unless you specify the SET CACHE ALL command in a calc script, in which case Hyperion Essbase uses the calculator cache, even when you do not have at least one full sparse dimension).

Understanding Calculator Cache Options

For the calculator cache, Hyperion Essbase separates the sparse dimensions in the database into two groups:

- Bitmap dimensions

  The sparse dimensions in the database outline that Hyperion Essbase can fit into the bitmap. Each member combination of these sparse dimensions occupies 1 bit of memory.

- "Anchoring" dimensions

  The remaining one or more sparse dimensions in the database outline (that is, the dimensions that do not fit into the bitmap).

To determine how many dimensions can fit into the bitmap, Hyperion Essbase starts with the first sparse dimension in the database outline and attempts to fit as many sparse dimensions as possible, based on their size, into the bitmap. The dimensions that fit are the bitmap dimensions. Only full dimensions can be placed in the bitmap. Hyperion Essbase stops the process when it cannot fit any more sparse dimensions into the bitmap, based on the size of the calculator cache. The remaining sparse dimensions are the anchoring dimensions. For anchoring dimensions, Hyperion Essbase cannot use the bitmap and must make calls to the Hyperion Essbase Kernel to determine whether or not blocks exist.

To see which dimensions are anchoring dimensions and which are bitmap dimensions, use the SET MSG DETAIL calculation command to display this information in the application event log. For more information on this command, see the online *Technical Reference* in the `DOCS` directory.

So that as many dimensions as possible can be placed into the bitmap, sparse dimensions should be carefully ordered in the outline. Order the sparse dimensions by the number of stored members the dimension contains, starting with the dimension that contains the fewest members. This order allows more dimensions to fit into the bitmap and results in improved calculation performance.

Note: The order of sparse dimensions in the outline also affects query performance. To optimize the outline for query performance, see "Positioning Dimensions and Members" on page 8-20.

After the bitmap dimensions are determined, Hyperion Essbase chooses whether to have one bitmap (single) or two bitmaps (multiple):

With a single bitmap:

- A single bitmap is used to track child blocks.
- A single bitmap uses the least memory but is less efficient than multiple bitmaps.
- The calculator cache uses a single bitmap if there is more than one anchoring dimension or if the calculator cache is not large enough to support multiple bitmaps.

With multiple bitmaps:

- Two or more bitmaps are used, one to track child blocks and one to track parent blocks.
- The use of multiple bitmaps is the optimal method. Multiple bitmaps use more memory but are more efficient than a single bitmap. The performance improvement is particularly high when you are calculating the database for the first time.
- The calculator cache can use multiple bitmaps only if there is a single anchoring dimension.
- The number of bitmaps used is determined by the maximum number of dependent parents for any of the members in the anchoring dimension.
- A member has one dependent parent, unless it has a shared member. For example, consider the Product dimension of the Sample Basic database. The member Cola (100-10) has one parent, which is Colas (100). However, Diet Cola (100-20) has two parents, which are Diet Drinks (Diet) and Colas (100). No members of Product have more than two dependent parents. Therefore, if Product is the anchoring dimension, the maximum number of dependent parents is 2.

Depending on the calculator cache size that you specify, Hyperion Essbase chooses one of three options for the calculation:

Option	Method	Performance Rating
1	Single anchoring dimension, multiple bitmaps	1
2	Single anchoring dimension, single bitmap	2
3	Multiple anchoring dimensions, single bitmap	3

Hyperion Essbase chooses the optimal performance method for the database calculation, based on the size of the calculator cache. If the calculator cache size is too small for any of the above options, Hyperion Essbase does not use a calculator cache. Calculation performance may be significantly impaired.

Calculating the Required Size of the Calculator Cache

The size of the calculator cache depends on the amount of memory the system has available. It also depends on the nature and configuration of the database. You can calculate the calculator cache size that is required for each of the 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: Single Anchoring Dimension, Multiple Bitmaps

Bitmap dimensions	S1, S2, S3, S4
Anchoring dimension	S5
Dependent parents in 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 in the anchoring dimension
+
2 constant bitmaps
= 3 + 2
= 5

Calculator cache = Bitmap size * Number of bitmaps
= 125,000 * 5
= **625,000 bytes**

Option 2: Single Anchoring Dimension, Single Bitmap

Bitmap dimensions	S1, S2, S3, S4
Anchoring dimension	S5
Dependent parents in 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
= 1
Calculator cache = Bitmap size * Number of bitmaps
= 125,000 * 1
= **125,000 bytes**

Option 3: Multiple Anchoring Dimensions, Single Bitmap

Bitmap dimensions	S1, S2, S3
Anchoring dimensions	S4, S5
Dependent parents in anchoring dimensions	Not applicable

Bitmap size in bytes = (S1 * S2 * S3)/8
= (20 * 20 * 50)/8
= 2,500 bytes
Number of bitmaps = Single bitmap
= 1
Calculator cache = Bitmap size * Number of bitmaps
= 2,500 * 1
= **2,500 bytes**

Conclusion

The following table shows which calculator cache option Hyperion Essbase uses, depending on the calculator cache size specified:

If you specify at least...	Hyperion Essbase uses...
625,000 bytes	Option 1 (provides optimal performance)
125,000 bytes	Option 2
2,500 bytes	Option 3

If you specify a calculator cache size of less than 2,500 bytes, Hyperion Essbase does not use a calculator cache during the calculation. Calculation performance may be significantly impaired.

You can check which calculator cache option Hyperion Essbase is able to use on a database by using the SET MSG SUMMARY command in a calc script. Run the following calc script on the empty database:

```
SET MSG SUMMARY;
CALC ALL;
```

Hyperion 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 33-4.

The size of the calculator, index, data file, and data caches generally has a greater effect on performance if the database calculation is based mainly on aggregations rather than on formula calculations.

Calculating the Database for the First Time

If you are calculating the database for the first time, the size of the calculator cache is particularly significant for calculation performance. If possible, ensure that the calculator cache is large enough for Hyperion Essbase to use the optimal calculator cache option. For more information, see “Understanding Calculator Cache Options” on page 33-16.

Using the Calculator Cache for Large, Flat Database Outlines

You can use the SET CALCHASHTBL command to optimize how Hyperion Essbase uses the calculator cache when calculating large, flat database outlines (for example, where one member has more than 5000 children). To use SET CALCHASHTBL, you must enable the calculator cache.

When you enable SET CALCHASHTBL, Hyperion Essbase uses a hash table to optimize use of the calculator cache for large, flat databases. Using this feature may significantly improve the performance of a CALC ALL of the database or a CALC DIM of the dimension containing the member with over 5000 children.

For more information, see the SET CALCHASHTBL command and the CALCOPTCALCHASHTBL and CALCHASHTBLMEMORY configuration settings in the online *Technical Reference* in the `DOCS` directory.

Locking Blocks During Calculation

When a block is calculated, Hyperion Essbase gets addressability to the block and to the blocks that contain the children of the block being calculated. Hyperion Essbase calculates the block and then releases both the block and the blocks containing its children.

By default, Hyperion Essbase gets addressability to up to 100 blocks concurrently when calculating a block. This number of blocks is sufficient for most database calculations. If you are calculating a formula in a sparse dimension, Hyperion Essbase works most efficiently if it can get addressability to all required children concurrently. Therefore, when calculating a formula in a sparse dimension, you may want to set a number higher than 100 if you are consolidating very large numbers of children (for example, more than 100 children). By increasing the number, you ensure that Hyperion Essbase can get addressability to all required blocks 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 Hyperion Essbase can get addressability to concurrently when calculating a block. For more information, see the online *Technical Reference* in the `DOCS` directory.

Note: For aggregations in a sparse dimension, block locking is not a consideration because Hyperion Essbase does not need to get addressability to all the children concurrently.

Hyperion Essbase locking behavior depends on the Hyperion Essbase Kernel Isolation Level setting. For more information, see Chapter 42, “Ensuring Data Integrity.”

Considering Multiple Users

Hyperion Essbase uses a block locking system to provide concurrent access to users. This system ensures that only one user at a time can update or calculate a particular data block. How Hyperion Essbase handles locking blocks and committing data depends on the Hyperion Essbase Kernel Isolation Level setting.

When Hyperion Essbase calculates a data block, it gets addressability to the block with an exclusive lock. Thus, no other user can update or calculate the data block. However, other users can have read-only access to the block. When Hyperion Essbase finishes the calculation, it releases the block. Other users can then update the block if they have the appropriate security access.

When a user is updating a data block, the block is locked. If a database calculation requires a data block that is being updated by another user, the calculation waits for one of the following, depending on the Hyperion Essbase Kernel 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)

Hyperion Essbase does not provide a message to say that the calculation is waiting for the data block to be released.

You can prevent calculation delays caused by waiting for locked blocks by using Hyperion Essbase security options to do either of the following:

- Deny access to other users
- Disconnect users from Hyperion Essbase

For more information on these security options, see Part III, “Designing and Building a Security System.” For information on how Hyperion Essbase handles locks and transactions, see Chapter 42, “Ensuring Data Integrity.”

Note: When Hyperion Essbase gets addressability to a data block for calculation, it does not put an exclusive lock on the dependent child blocks. Thus, another user can update values in the child blocks. If necessary, you can use the above security options to prevent such updates.

Using Two-Pass Calculation

In some cases, you can achieve a significant performance improvement by tagging an accounts dimension member as two-pass in the database outline rather than by repeating the formula in a calc script. In other cases, you may need to use a calc script to calculate a formula twice.

Note: You can achieve performance improvement by tagging two-pass members as Dynamic Calc. You can tag Dynamic Calc members as two-pass, even if the members are *not* in the accounts dimension. For more information, see Chapter 29, “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 table shows a subset of a data block with Measures and Year as dense dimensions. Measures is tagged as accounts, and Year is tagged as time. The AGGMISSG setting is turned off (the default).

Data values have been loaded into the input cells. Hyperion Essbase calculates the shaded cells. The numbers in bold show the calculation order for the cells. Cells with multiple consolidation paths are darkly shaded.

Measures/Year	Jan	Feb	Mar	Qtr1
Profit	75	50	120	**5**
Sales	150	200	340	**6**
Profit%	**1**	**2**	**3**	**4/7**

Note: For detailed information on how cell calculation order depends on database configuration, see Chapter 28, "Defining the Calculation Order."

The calculation order is as follows:

1. Hyperion Essbase calculates the formula `Profit % Sales` for Profit%->Jan, Profit%->Feb, Profit%->Mar, and Profit%->Qtr1.
2. Hyperion Essbase calculates Profit->Qtr1 and Sales->Qtr1 by adding the values for Jan, Feb, and Mar.
3. Hyperion 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, Hyperion Essbase uses the `Profit % Sales` formula to recalculate the Profit% values and produce the correct results.

Measures/Year	Jan	Feb	Mar	Qtr1
Profit	75	50	120	**245**
Sales	150	200	340	**690**
Profit%	**50%**	**25%**	**50%**	**36%**

Note: You can tag a member as two-pass only if it is in 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* in the accounts dimension. For more information, see Chapter 29, "Dynamically Calculating Data Values."

When you perform a default calculation on the database, Hyperion Essbase automatically recalculates any formulas tagged as two-pass in the dimension tagged as accounts in the database outline.

In some situations, you may need to use a calc script to calculate a two-pass formula. For more information, see "Using a Calc Script for Two-Pass Calculations" on page 33-31.

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 33-27.

When you use a calc script, Hyperion 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 33-31.

Using Two-Pass on a Default Calculation

When you perform a default calculation on a database with two-pass calculation enabled (the default), Hyperion Essbase automatically attempts to calculate any formulas tagged as two-pass in the dimension tagged as accounts in the database outline. This is true even if you have customized the default calc script.

You can perform a default calculation by using either of the following:

- ESSCMD

  Use the CALCDEFAULT command in ESSCMD to run the default calculation. For information about the CALCDEFAULT command, see the online *Technical Reference* in the `DOCS` directory. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

- Hyperion Essbase Application Manager (by selecting Database > Calculate and selecting Default)

➤ To enable two-pass calculation in Hyperion Essbase Application Manager:

1. Select Database > Settings.

   Hyperion Essbase displays the Database Settings dialog box.

2. On the General tab, check Two-Pass Calculation.
3. Click OK.

You can also use the SETDBSTATE command in ESSCMD to enable two-pass calculation. For information about the SETDBSTATE command, see the online *Technical Reference* in the `DOCS` directory. See Chapter 44, “Performing Interactive and Batch Operations Using ESSCMD” for information about ESSCMD.

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 33-31.

Hyperion Essbase recognizes formulas on members tagged as two-pass only in 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* in the accounts dimension. For more information, see Chapter 29, “Dynamically Calculating Data Values.”

When you are doing a default calculation of the database, you need to ensure that the two-pass calculation option is selected in the Database Settings dialog box in Hyperion Essbase Application Manager. Hyperion Essbase then does the two-pass calculation as part of the default calculation.

Whenever possible, Hyperion Essbase calculates two-pass formulas at the data block level, calculating the two-pass formulas at the same time as the main calculation. Thus, Hyperion Essbase does not need to do an extra calculation pass through the database. However, in some situations, Hyperion Essbase needs to calculate the two-pass formulas on an extra calculation pass through the database. For information on calculation passes, see Chapter 28, “Defining the Calculation Order.”

How Hyperion Essbase calculates the two-pass formulas depends on whether there is a dimension tagged as time as well as a dimension tagged as accounts. It also depends on the dense-sparse configuration of these dimensions.

Two scenarios are described in detail in the following sections. If you are using Intelligent Calculation, consider the scenario that matches the configuration of your database; each scenario tells you how to ensure that Hyperion Essbase calculates two-pass formulas accurately.

The following information assumes that you understand the concepts of Intelligent Calculation. For more information, see Chapter 34, “Using Intelligent Calculation to Optimize Calculation.”

Scenario A

Scenario A may be summarized as follows:

- No extra calculation pass for two-pass formulas
- All data blocks marked as clean

In Scenario A, you place formulas in the outline and then, as appropriate, tag specific formulas as two-pass. Scenario A produces an efficient calculation process.

No extra calculation pass for two-pass formulas

Hyperion Essbase calculates the two-pass formulas while it is calculating the data block. Thus, Hyperion Essbase does not need to do an extra calculation pass through the database.

All data blocks marked as clean

After the calculation, all data blocks are marked as clean for the purposes of Intelligent Calculation. For more information, see Chapter 34, “Using Intelligent Calculation to Optimize Calculation.”

If you have changed the 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 34, “Using Intelligent Calculation to Optimize Calculation.” You can check the default calculation setting by selecting Database > Set Default Calc in Hyperion Essbase Application Manager.

If the database configuration allows Hyperion Essbase to use Scenario A, you can achieve a performance improvement by tagging a member formula as two-pass in the outline rather than by repeating the formula in a calc script. When you tag a member formula as two-pass in the outline, Hyperion Essbase does the two-pass calculation while each data block is being calculated. However, when you repeat a formula in a calc script, Hyperion Essbase has to read and write the data blocks to memory in order to recalculate the formula.

Scenario B

Scenario B may be summarized as follows:

- Extra calculation pass for two-pass formulas
- Data blocks representing two-pass formulas are *not* marked as clean

Extra calculation pass for two-pass formulas

Hyperion Essbase calculates the database and then does an extra calculation pass to calculate the two-pass formulas. Even though all data blocks are marked as clean after the first database calculation, Hyperion Essbase ignores the clean status on the blocks that represent the two-pass formulas and recalculates these blocks.

Data blocks representing two-pass formulas are not marked as clean

After the first calculation, Hyperion Essbase has marked all the data blocks as clean for the purposes of Intelligent Calculation. In a second calculation pass through the database, Hyperion Essbase recalculates the required data blocks for the two-pass formulas. However, because the second calculation is a partial calculation of the database, Hyperion Essbase does not mark the recalculated blocks as clean. When you recalculate the database with Intelligent Calculation turned on, these data blocks may be recalculated unnecessarily.

If you have changed the 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 34, “Using Intelligent Calculation to Optimize Calculation.” You can check the default calculation setting by selecting Database > Set Default Calc in Hyperion Essbase Application Manager.

If the database configuration allows Hyperion Essbase to use Scenario B, consider using a calc script to perform two-pass formula calculations. If you use a calc script, Hyperion Essbase still does an extra calculation pass through the database; however, you can ensure that Hyperion 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 33-31.

Using a Calc Script for Two-Pass Calculations

You need to use a calc script to calculate a formula twice if the database configuration means that Hyperion Essbase uses Scenario A, as described in “Using Two-Pass on a Default Calculation” on page 33-27, and if the formula references values from another data block.

You may want to use a calc script to calculate two-pass formulas if the database configuration means that Hyperion Essbase uses Scenario B, as described in “Using Two-Pass on a Default Calculation” on page 33-27.

Consider an example in which the 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 Hyperion Essbase calculates the data block for each product, it has not yet calculated the value Sales->Product, so the results for the sales of each product as a percentage of total sales are incorrect.

If you have Intelligent Calculation turned on (the default), Hyperion Essbase calculates only those data blocks that are not marked as clean. When you perform a default calculation of the database with Intelligent Calculation turned on, all the data blocks are marked as clean. Therefore, before you recalculate a two-pass 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 the database.

Option 1: Using Intelligent Calculation with a Large Index

If the index is quite large and you want the benefit of using Intelligent Calculation, you can use any of the following options:

- **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 calculation marks all the data blocks as clean. Then, in the calc script, turn off Intelligent Calculation, tell Hyperion Essbase to mark the recalculated 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 a member as two-pass, and use a calc script to calculate the two-pass member:** Place a formula in the database outline and tag it as two-pass. Place the formula on the Share of Sales member in the dimension tagged as accounts. In this case, the required calc script is as follows:

  ```
  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 a full calculation in Hyperion Essbase Application Manager. Select Database > Calculate from the main menu and select Default (providing that you have not customized the default calc script), or use 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;
  ```

Hyperion Essbase performs the following calculations during each of the three options:

1. The SET UPDATECALC ON command turns on Intelligent Calculation.
2. The CALC ALL command 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 Hyperion Essbase to mark the recalculated blocks as clean, even though this calculation is a partial calculation of the database. (If you do not use the SET CLEARUPDATESTATUS AFTER command, Hyperion Essbase marks data blocks as clean only after a full calculation of the database.)
5. Hyperion Essbase cycles through the database calculating only the formula for Share of Sales or calculating all formulas tagged as two-pass in the database outline.

Note: For more information on Intelligent Calculation, see Chapter 34, "Using Intelligent Calculation to Optimize Calculation." For more information on developing formulas and calc scripts, see Chapter 26, "Developing Formulas," and Chapter 31, "Developing Calc Scripts."

Option 2: Using Intelligent Calculation with a Small Index

If the index is small and you want the benefit of using Intelligent Calculation, you can run a calc script to calculate the database, but tell Hyperion Essbase not to mark the calculated data blocks as clean. Then mark all the data blocks as clean, but do *not* recalculate the data blocks.

```
SET CLEARUPDATESTATUS OFF;
CALC ALL;
CALC TWOPASS;
SET CLEARUPDATESTATUS ONLY;
CALC ALL;
```

Hyperion Essbase performs the following operations:

1. The SET CLEARUPDATESTATUS OFF command tells Hyperion Essbase not to mark the calculated data blocks as clean.
2. The CALC ALL command causes Hyperion Essbase to cycle through the database calculating all dirty data blocks. Hyperion Essbase does not mark the calculated data blocks as clean. Hyperion Essbase does *not* automatically recalculate the formulas tagged as two-pass in the database outline.
3. Hyperion Essbase cycles through the database recalculating the formulas that are tagged as two-pass in the dimension tagged as accounts in the database outline. Hyperion Essbase recalculates the formulas because the required data blocks are *not* marked as clean by the previous CALC ALL. Hyperion Essbase does *not* mark these recalculated data blocks as clean.
4. The SET CLEARUPDATESTATUS ONLY command tells Hyperion 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 Hyperion Essbase to cycle through the database and mark all the data blocks as clean. Hyperion Essbase searches through the index and marks the data blocks as clean. It does not calculate the data blocks.

Option 3: Not Using Intelligent Calculation

Use a calc script to turn off Intelligent Calculation, perform a full calculation, and repeat the two-pass 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, Hyperion Essbase gives the combination a value of #MISSING. Hyperion Essbase treats #MISSING values and zero (0) values differently.

The following table shows how Hyperion Essbase calculates #MISSING values. In this table, X represents any number:

Table 33-1: How Hyperion Essbase Treats #MISSING Values

Calculation/Operation	Result
X + #MISSING	X
X – #MISSING #MISSING – X	X -X
X * #MISSING	#MISSING
X / #MISSING #MISSING / X X / 0	#MISSING #MISSING #MISSING
X % #MISSING #MISSING % X X % 0	#MISSING #MISSING #MISSING
X == #MISSING	FALSE, unless X is #MISSING
X != #MISSING X <> #MISSING	TRUE, unless X is #MISSING TRUE, unless X is #MISSING
(X <= #MISSING)	(X <= 0)
(X >= #MISSING)	(X >= 0) or (X == #MISSING)
(X > #MISSING)	(X > 0)
(X < #MISSING)	(X < 0)
X AND #MISSING: 1 AND #MISSING, where 1 represents any nonzero value 0 AND #MISSING #MISSING AND #MISSING	 #MISSING 0 #MISSING

Table 33-1: How Hyperion Essbase Treats #MISSING Values (Continued)

Calculation/Operation	Result
X OR #MISSING: 1 OR #MISSING, where 1 represents any nonzero value 0 OR #MISSING #MISSING OR #MISSING	 1 #MISSING #MISSING
IF (#MISSING)	IF (0)
f (#MISSING)	#MISSING for any Hyperion Essbase function of one variable
f (X)	#MISSING for any X not in the domain of *f* and any Hyperion Essbase function of more than one variable (except where specifically noted)

By default, Hyperion Essbase does not aggregate #MISSING values. This default is important to consider when you need to load data at parent, rather than child, levels. For more information, see "Loading Data at Parent Levels" on page 33-38.

You can change the way Hyperion Essbase aggregates #MISSING values by doing any of the following:

- Changing the default setting in Hyperion Essbase Application Manager
- Using the SET AGGMISSG command in a calc script
- Using the SETDBSTATEITEM command in ESSCMD

For information about the SETDBSTATEITEM command, see the online *Technical Reference* in the DOCS directory. See Chapter 44, "Performing Interactive and Batch Operations Using ESSCMD" for information about ESSCMD.

➤ To change how #MISSING values are aggregated at the database level, use Hyperion Essbase Application Manager:

1. Select Database > Settings.

   Hyperion Essbase displays the Database Settings dialog box.

2. Check Aggregate Missing Values.

   By default, this setting is turned off.

3. Click OK.

To change how #MISSING values are aggregated on a per-calculation basis, use the SET AGGMISSG calculation command in a calc script. For more information on this command, see the online *Technical Reference* in the DOCS directory.

If you never load data at parent levels, you can achieve performance improvements by aggregating #MISSING values. To aggregate, enable the setting for aggregating #MISSING values by using one of the methods described above. The degree of performance improvement you achieve depends on the ratio between upper level blocks and input blocks in the database. For more information, see Chapter 28, "Defining the Calculation Order."

Note: If you enable the setting for aggregating #MISSING values, the cell calculation order within a data block changes. For more information, see Chapter 28, "Defining the Calculation Order."

When the setting for aggregating #MISSING values is disabled, note that the performance overhead is particularly high in the following two situations:

- When you have a low ratio of calculated data blocks to input data blocks
- When you load many data values at parent levels on sparse dimensions; for example, in the Sample Basic database, if you load many data values into East in a sparse Market dimension

In these situations, the performance overhead is between 10% and 30%. If calculation performance is critical, you may want to reconsider the database configuration or reconsider how you load data.

Loading Data at Parent Levels

Sometimes you may need to load data into cells that represent parent members of one or more dimensions. For example, in the Sample Basic database, you may want to load Miscellaneous Expenses (Misc) for each product at the consolidated product level. You 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 a parent load, you do not want the loaded data to be overwritten when Hyperion Essbase aggregates the `#MISSING` values in the children. By default, Hyperion 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 enable the setting for aggregating `#MISSING` values. Use of this setting provides a calculation performance improvement of between 1% and 30%. The performance improvement varies, depending on database size and configuration. For more information on setting the behavior for aggregating `#MISSING` values, see “Calculating #MISSING Values” on page 33-35, Chapter 28, “Defining the Calculation Order,” and the online *Technical Reference* in the `DOCS` directory.

CAUTION: You need to ensure that the setting for aggregating `#MISSING` values is off (the default) to protect 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 Hyperion Essbase aggregates the child values and overwrites the parent values.

Chapter 34

Using Intelligent Calculation to Optimize Calculation

This chapter provides information on how to use Intelligent Calculation to optimize the performance of Hyperion Essbase calculations. For additional information on optimizing overall database calculations, see the following:

- Chapter 33, “Optimizing Calculations”
- Chapter 29, “Dynamically Calculating Data Values”
- Chapter 6, “Designing Partitioned Applications”

This chapter includes the following sections:

Introducing Intelligent Calculation

When you do a full calculation of a database, Hyperion Essbase tracks which data blocks it has calculated. If you then load a small subset of data, on subsequent calculations, you can choose to calculate *only* those data blocks that Hyperion Essbase has *not* calculated or that require recalculating. In Hyperion Essbase, this process is called Intelligent Calculation.

Benefits of Intelligent Calculation

Intelligent Calculation is designed to provide significant calculation performance benefits for a full calculation of a database (CALC ALL), or when you use a calc script that calculates all members in *one* CALC DIM command. For more information, see “Intelligent Calculation and Data Block Status” on page 34-2. If you cannot use Intelligent Calculation for all of a database calculation, you may be able to use it for part of the calculation.

For example, consider a case in which you calculate a database by doing a default consolidation and then an allocation of data. To significantly improve your calculation performance in this case, turn on Intelligent Calculation for the default consolidation and then turn off Intelligent Calculation for the allocation.

Assuming that Intelligent Calculation is turned on (the default), use a calc script to accomplish the following:

1. Do a CALC ALL of a database with Intelligent Calculation turned on.
2. Use the SET UPDATECALC command to turn off Intelligent Calculation for the duration of the calc script.
3. Allocate Headquarters Costs across all products.

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 34-5. For more information on the `ESSBASE.CFG` file, see the online *Technical Reference* in the `DOCS` directory.

Intelligent Calculation and Data Block Status

To provide Intelligent Calculation, Hyperion Essbase uses the status of the data blocks in a database. Data blocks have a calculation status of either *clean* or *dirty*. Hyperion Essbase marks a data block as clean after certain calculations.

When Intelligent Calculation is turned on, Hyperion Essbase calculates only dirty blocks and their dependent parents. Turning off Intelligent Calculation means that Hyperion Essbase calculates all data blocks, regardless of whether they are marked as clean or dirty.

Hyperion Essbase marks data blocks as clean in the following calculations:

- A full calculation (CALC ALL) of a database. Unless you have changed it, CALC ALL is the default calculation for a database. You can check the default calculation setting by selecting 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 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 Hyperion Essbase to do at least two calculation passes through the database. In this calculation, Hyperion 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 Hyperion Essbase to do two or more calculation passes through a database, you need to consider carefully the effects of Intelligent Calculation. For more information, see “Using the SET CLEARUPDATESTATUS Command” on page 34-7. For information on calculation passes, see Chapter 28, “Defining the Calculation Order.”

Hyperion 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 a calc script. For more information, see “Using the SET CLEARUPDATESTATUS Command” on page 34-7.

Hyperion Essbase marks a data block as dirty in the following situations:

- It calculates the data block. If the data block is part of a full calculation or if you are using the SET CLEARUPDATESTATUS AFTER command, Hyperion Essbase marks the data block as clean.
- You load data into a data block.
- You restructure the 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.

Maintaining Clean and Dirty Status

If you want to use Intelligent Calculation when calculating a subset of a database or when performing multiple calculation passes through a database, consider carefully the implications of how Hyperion Essbase marks data blocks as clean. When using Intelligent Calculation, carefully maintain the clean and dirty status of the data blocks to ensure that Hyperion Essbase recalculates the database as efficiently as possible.

For example, when you calculate a subset of a database, the newly calculated data blocks are not marked as clean by default. You can ensure that the newly calculated blocks are marked as clean by using the SET CLEARUPDATESTATUS AFTER command in a calc script. To ensure accurate calculation results, you need to consider the information in "Using the SET CLEARUPDATESTATUS Command" on page 34-7 and the online *Technical Reference* in the `DOCS` directory.

Limitations of Intelligent Calculation

Consider the following limitations when using Intelligent Calculation:

- Intelligent Calculation works on a data block level and not on a cell level. For example, if you load a data value into one cell in a data block, the whole data block is marked as dirty.
- Changing a formula on the database outline or changing an accounts property on the database outline does not cause Hyperion Essbase to restructure the database. Therefore, Hyperion Essbase does not mark the affected blocks as dirty. You must recalculate the appropriate data blocks. For more information, see "Changing a Formula or Accounts Property" on page 34-18.
- Whenever possible, Hyperion Essbase calculates formulas tagged as Two-Pass on the dimension tagged as accounts as part of the main calculation of a database. However, you may need to use a calc script to calculate some formulas twice. When you use a calc script, you need to turn off Intelligent Calculation before recalculating formulas.

Using Intelligent Calculation

This section provides information on turning Intelligent Calculation on and off and on using Intelligent Calculation with different types of calculations.

Turning Intelligent Calculation On and Off

By default, Intelligent Calculation is turned on. You can change the default by using the UPDATECALC setting in the `ESSBASE.CFG` file.

You can turn Intelligent Calculation on and off for the duration of a calc script by using the SET UPDATECALC command in a calc script. Turning on Intelligent Calculation means that Hyperion Essbase calculates only dirty blocks and their dependent parents. Turning off Intelligent Calculation means that Hyperion 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 the `DOCS` directory.

Using Intelligent Calculation for a Default, Full Calculation

Intelligent Calculation provides significant performance benefits when you do a full calculation (CALC ALL) of a database. If you do a full calculation of a database, leave Intelligent Calculation turned on (the default) to take advantage of the performance benefits it provides.

Unless you have changed the default, a full calculation (CALC ALL) is the default calculation for a database. You can check the default calculation setting by selecting Database > Set Default Calc in Hyperion Essbase Application Manager.

CAUTION: When using Intelligent Calculation, note the information in "Limitations of Intelligent Calculation" on page 34-4.

Calculating for the First Time

When you do a full calculation of a database for the first time, Hyperion Essbase calculates every existing block. The performance is the same whether you have Intelligent Calculation turned on or off.

Recalculating

When you do a full recalculation of a database with Intelligent Calculation turned on, Hyperion 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 "Intelligent Calculation and Data Block Status" on page 34-2.

During most recalculations, this small performance overhead is insignificant when compared to the performance gained by turning on Intelligent Calculation.

However, if you recalculate a database in which more than approximately 80% of the values have changed, the overhead of Intelligent Calculation may outweigh the benefits. In this case, you can turn off Intelligent Calculation.

Using Intelligent Calculation for a Partial, Calc Script Calculation

Hyperion Essbase marks a data block as clean when it calculates the data block on a full calculation (CALC ALL) or when it calculates all dimensions in one CALC DIM command. For more information, see "Intelligent Calculation and Data Block Status" on page 34-2.

In any other calculations, Hyperion Essbase does *not* mark calculated data blocks as clean, unless you use the SET CLEARUPDATESTATUS command in a calc script. For example, if you calculate a subset of a database or calculate a database in two calculation passes, Hyperion Essbase does not mark the calculated blocks as clean, unless you use the SET CLEARUPDATESTATUS command.

The following calc scripts do *not* cause Hyperion 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

In some cases, Hyperion Essbase does not mark calculated blocks as clean; for example, if you calculate a subset of a database or calculate a database in two calculation passes. To manually mark data blocks as clean for purposes of Intelligent Calculation, use the SET CLEARUPDATESTATUS command in a calc script. For more information, see “Intelligent Calculation and Data Block Status” on page 34-2.

The SET CLEARUPDATESTATUS command has three parameters: AFTER > ONLY > OFF.

- SET CLEARUPDATESTATUS AFTER;

  Hyperion Essbase marks calculated data blocks as clean, even if it is calculating a subset of a database.

- SET CLEARUPDATESTATUS ONLY;

  Hyperion Essbase marks the specified data blocks as clean, but does not actually calculate the data blocks. This parameter provides the same result as AFTER, but disables calculation.

- SET CLEARUPDATESTATUS OFF;

  Hyperion Essbase calculates the data blocks but does not mark the calculated data blocks as clean. Data blocks are not marked as clean, even on a full calculation (CALC ALL) of a database. The existing clean or dirty status of the calculated data blocks remains unchanged.

Considerations for Using SET CLEARUPDATESTATUS

When you use the SET CLEARUPDATESTATUS command to mark calculated data blocks as clean, it is imperative that you 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 34-10.

Do concurrent calculations affect the same data blocks?

Do not use the SET CLEARUPDATESTATUS AFTER command with concurrent calculations unless you are certain that the concurrent calculations do not need to calculate the same data block or blocks. If concurrent calculations attempt to calculate the same data blocks, with Intelligent Calculation turned on, Hyperion Essbase may not recalculate the data blocks because the blocks are already marked as clean. For more information, see “Handling Concurrent Calculations” on page 34-12.

Does Hyperion Essbase recalculate the same data blocks on a second calculation pass through a database?

When Hyperion Essbase calculates data blocks on a first calculation pass through a database, it marks the data blocks as clean. If you then attempt to calculate the same data blocks on a subsequent pass with Intelligent Calculation turned on, Hyperion Essbase does not recalculate the data blocks because they are already marked as clean.

Examples Using SET CLEARUPDATESTATUS

The following examples are based on the Sample Basic database. Assume the following:

- The sparse dimensions are Market and Product.
- New York is a member on the sparse Market dimension.
- Intelligent Calculation is turned on (the default).

Example 1

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Product);
ENDFIX
```

In this example, Hyperion Essbase searches for dirty parent data blocks for New York (for example New York->Colas, in which Colas is a parent member). It calculates these dirty blocks and marks them as clean. (The calculation is based on the Product dimension.) Hyperion Essbase does not mark the level 0 data blocks as clean because they are not calculated. For information on level 0 blocks, see Chapter 28, “Defining the Calculation Order.”

Example 2

```
SET CLEARUPDATESTATUS ONLY;
FIX("New York")
CALC DIM(Product);
ENDFIX
```

This example differs from the first example in that the SET CLEARUPDATESTATUS ONLY command is used instead of SET CLEARUPDATESTATUS AFTER. Hyperion 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.

Example 3

```
SET CLEARUPDATESTATUS OFF;
CALC ALL;
CALC TWOPASS;
SET CLEARUPDATESTATUS ONLY;
CALC ALL;
```

In this example, Hyperion Essbase first calculates all the dirty data blocks in the database. The calculated data blocks remain dirty. Hyperion Essbase does *not* mark them as clean. Hyperion Essbase then calculates those members tagged as Two-Pass on the dimension tagged as accounts. Because the data blocks are still marked as dirty, Hyperion Essbase recalculates them. Again, it does not mark the calculated data blocks as clean. Hyperion 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

Hyperion 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, level 0, and input data blocks. For more information on how Hyperion Essbase creates data blocks, see Chapter 28, “Defining the Calculation Order.”

Calculating a Dense Dimension

When you calculate a dense dimension and do not use a FIX command, Hyperion Essbase calculates at least some of the data values in every data block in a database. For example, the following calc script is based on the Sample Basic database:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Year);
```

This script calculates the Year dimension, which is a dense dimension. Because Year is dense, every data block in the database includes members of the Year dimension. Therefore, Hyperion Essbase calculates data values in every data block. Because the script uses the SET CLEARUPDATESTATUS AFTER command, Hyperion Essbase marks all the data blocks as clean.

Calculating a Sparse Dimension

When you calculate a sparse dimension, Hyperion Essbase may not need to calculate every data block in a database. For example, the following calc script is based on the Sample Basic database:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product);
```


This script calculates the Product dimension, which is a sparse dimension. Because Product 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, Hyperion Essbase does not need to calculate this data block. It does not mark it as clean, even though the script uses 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. Hyperion Essbase needs to calculate values for Colas, so Hyperion Essbase calculates this data block. Because the script uses the SET CLEARUPDATESTATUS AFTER command, Hyperion Essbase marks this data block as clean.

When Hyperion 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 use the SET CLEARUPDATESTATUS AFTER command.

When Hyperion 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 a database, the level 0 data blocks into which you load data are marked as dirty. If you subsequently calculate only a sparse dimension or dimensions, these level 0 blocks remain dirty, because Hyperion Essbase does

not calculate them. Therefore, when you recalculate only a sparse dimension or dimensions, Hyperion 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 unnecessary calculation by ensuring that you calculate at least one dense dimension. When you calculate a dense dimension and do not use the FIX command, data values are calculated in every data block, including the level 0 blocks. So the level 0 blocks are marked as clean.

Handling Concurrent Calculations

If concurrent calculations attempt to calculate the same data blocks and Intelligent Calculation is turned on, Hyperion Essbase may not recalculate the data blocks because they are already marked as clean.

Do not use the SET CLEARUPDATESTATUS AFTER command with concurrent calculations unless you are certain that the concurrent calculations do not calculate the same data block or blocks.

Consider the following example, which is based on the Sample Basic database. Actual and Budget are members of the dense Scenario dimension. Because Scenario is dense, each data block in the database contains both Actual and Budget values.

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York", Actual)
CALC DIM(Product, Year);
ENDFIX
```

If User One runs the above calc script, Hyperion Essbase calculates the Actual values for all data blocks that represent New York. Hyperion Essbase marks the calculated data blocks as clean, even though not all the data values in each calculated block have been calculated. For example, the Budget values have not yet been calculated.

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York", Budget)
CALC DIM(Product, Year);
ENDFIX
```

If User Two runs the previous calc script to calculate the Budget values for New York, Hyperion 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 are in different data blocks, for example, New York->Colas->Actual and New York->Colas->Budget. In this case, the second calc script correctly calculates the data blocks.

Handling Multiple-Pass Calculations

Whenever possible, Hyperion Essbase calculates a database in one calculation pass through the database. For information on calculation passes, see Chapter 28, "Defining the Calculation Order."

When you use a calc script to calculate a database, the number of calculation passes Hyperion Essbase performs depends upon the calc script. For more information, see "Intelligent Calculation and Data Block Status" on page 34-2, and for more information on grouping formulas and calculations, see Chapter 31, "Developing Calc Scripts."

Consider a situation in which you calculate data blocks on a first calculation pass through a database and Hyperion Essbase marks them as clean. If you then attempt to calculate the same data blocks on a subsequent pass with Intelligent Calculation turned on, Hyperion Essbase does not recalculate the data blocks because they are already marked as clean.

The following examples describe situations in which you obtain incorrect calculation results. The examples provide solutions for these situations. The 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;
```

Hyperion Essbase calculates the dirty data blocks in the database and marks all the data blocks as clean. Hyperion Essbase then needs to recalculate those members tagged as Two-Pass in the dimension tagged as accounts. However, Hyperion 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. The calculation is based on the Product dimension:

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Product);
ENDFIX
CALC TWOPASS;
```

Hyperion Essbase performs the following processes:

1. Because the SET CLEARUPDATESTATUS AFTER command is used, Hyperion Essbase marks the calculated blocks as clean, even though only part of the database is being calculated.
2. Hyperion Essbase cycles through the database calculating the dirty data blocks that represent New York. The calculation is based on the Product dimension. Thus, Hyperion Essbase calculates only those blocks that represent a parent member on the Product dimension (for example, New York->Colas, New York->Root Beer, and New York->Fruit Soda). Hyperion Essbase calculates only the aggregations and formulas for the Product dimension. Hyperion Essbase marks the calculated data blocks as clean, even though all the data values in each calculated block have not been calculated.

3. Hyperion Essbase needs to recalculate those members tagged as Two-Pass in the dimension tagged as accounts. However, some of these data blocks are already marked as clean from the calculation in step 2. Hyperion 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 28, "Defining the Calculation Order."

Example 3

This calc script bases the database calculation on the Product and Year dimensions. Because two CALC DIM commands are used, Hyperion Essbase does two calculation passes through the database:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product);
CALC DIM(Year);
```

Hyperion Essbase performs the following processes:

1. Because the SET CLEARUPDATESTATUS AFTER command is used, Hyperion Essbase marks the calculated blocks as clean, even though only part of the database is being calculated.
2. Hyperion Essbase cycles through the database calculating the dirty data blocks. The calculation is based on the Product dimension, as in Example 2. Hyperion Essbase marks the calculated data blocks as clean, even though all the data values in each calculated block have not been calculated.
3. Hyperion Essbase needs to recalculate the data blocks. The recalculation is based on the Year dimension. However, as a result of the calculation in step 2, some of the data blocks are already marked as clean. Hyperion 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 both the Product and Year dimensions. Hyperion Essbase then calculates both dimensions in one calculation pass through the database. The following calc script calculates the correct results:

```
SET CLEARUPDATESTATUS AFTER;
CALC DIM(Product, Year);
```

Note: When you calculate several dimensions in one CALC DIM command, Hyperion 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 28, "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
```

Hyperion Essbase calculates the data blocks that include New York. The calculation is based on the Product dimension. Thus, Hyperion Essbase calculates only the dirty blocks that include a parent member on the Product dimension (for example, New York->Colas, New York->Root Beer, and New York->Fruit Soda). It calculates only the aggregations and formulas for the Product dimension. Hyperion 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 calculates the Year dimension:

```
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Year);
ENDFIX
```

Hyperion Essbase calculates the data blocks that represent New York. The calculation is based on the Year dimension. Year is a dense dimension, which means that Hyperion Essbase needs to calculate all data blocks that include New York. Hyperion Essbase calculates only the aggregations and formulas for the Year dimension.

However, as a result of the previous calculation, some of the data blocks for New York are already marked as clean. Hyperion Essbase does not recalculate these data blocks because they are already marked as clean. The calculation results are not correct.

You can calculate the correct results by telling Hyperion Essbase *not* to mark the calculated data blocks as clean. The following calc script calculates the correct results:

```
SET CLEARUPDATESTATUS OFF;
FIX("New York")
CALC DIM(Product);
ENDFIX
SET CLEARUPDATESTATUS AFTER;
FIX("New York")
CALC DIM(Year);
ENDFIX
```


The SET CLEARUPDATESTATUS OFF command tells Hyperion Essbase to calculate the dirty data blocks, but *not* to mark them as clean. The SET CLEARUPDATESTATUS AFTER command tells Hyperion Essbase to mark the data blocks as clean.

This solution assumes that the data blocks are *not* already marked as clean from a previous partial calculation of the database.

You can ensure that all data blocks are calculated, irrespective of their clean or dirty status, by 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, Hyperion Essbase does not mark the calculated data blocks as clean.

Considering the Effects of Intelligent Calculation

Using Intelligent Calculation may have implications for the way you administer a database. This section discusses the implications of each of the following:

- Changing a formula or accounts property
- Using relationship and financial functions
- Restructuring
- Copying and clearing data
- Converting currencies

Changing a Formula or Accounts Property

Changing a formula in the database outline, or changing an accounts property in the database outline, does not cause Hyperion Essbase to restructure the database. Thus, the data blocks affected by the change are not marked as dirty, and when you subsequently run a default calculation with Intelligent Calculation turned on, your changes are *not* calculated. For example, if you change a time balance tag in the dimension tagged as accounts, Hyperion Essbase does not restructure the database and does not mark the affected blocks as dirty.

You must recalculate the appropriate data blocks. You can accomplish this by using a calc script to do any of the following:

- 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 a database.
- Turn off Intelligent Calculation and perform a default CALC ALL on a database.

For more detailed information, see Chapter 31, “Developing Calc Scripts.”

Using Relationship and Financial Functions

If you use relationship functions (for example, @PRIOR or @NEXT) or financial functions (for example, @NPV or @INTEREST) in a formula on a sparse dimension, Hyperion Essbase always recalculates the data block that contains the formula, even if the data block is marked as clean.

For more information on relationship functions and financial functions, see the online *Technical Reference* in the `DOCS` directory.

Restructuring a Database

When you restructure a database (for example, by adding a member to a dense dimension), all data blocks potentially need recalculating. Therefore, Hyperion Essbase marks all data blocks as dirty. When you calculate the restructured database, all blocks are calculated.

Note: Changing a formula in the database outline or changing an accounts property in the database outline does not cause Hyperion Essbase to restructure the database. You must recalculate the appropriate data blocks. For more information, see "Changing a Formula or Accounts Property" on page 34-18.

Copying and Clearing Data

When you copy values to a data block by using the DATACOPY command, the resulting data block is marked as dirty. Hyperion Essbase calculates the block when you recalculate a database.

When you clear data values by using the CLEARDATA and CLEARBLOCK commands, Hyperion 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 the `DOCS` directory.

Converting Currencies

When you convert currencies using the CCONV command, the resulting data blocks are marked as dirty. Hyperion Essbase calculates all the converted blocks when you recalculate a database.

For more information on this command, see the online *Technical Reference* in the `DOCS` directory.

Index

This index spans two volumes. Chapters 1–34 are in Volume I; Chapters 35–48 and an Appendix are in Volume II. Examples of entries: 33-10 means Chapter 33, p. 10; A-2 means Appendix A, p. 2.

Symbols

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z