Oracle Essbase 11.1.2 Bootcamp

Volume I • Student Guide

D63979GC10
Edition 1.0
May 2010
D67411

ORACLE

Author

Lisa Alexander

Technical Contributors and Reviewers

Art Hetherington
Peter DeHaan
Sergiy Pecherskyy
Yulia Stolerman
Saju Phillips

Editor

Susan Moxley

Table of Contents

Preface

Lesson 1: Essbase Overview

Lesson 2: Designing Applications and Databases

Lesson 3: Designing Data Descriptor Dimensions

Lesson 4: Optimizing Data Descriptor Dimensions

Lesson 5: Developing Dimension Designs

Lesson 6: Creating Basic Dimension Build Rules Files

Lesson 7: Creating Advanced Dimension Build Rules Files

Lesson 8: Loading Data

Lesson 9: Getting Started with Smart View

Lesson 10: Creating Reports with Smart View

Lesson 11: Data Storage and Calculation

Lesson 12: Creating Calculation Scripts

Lesson 13: Controlling the Calculation Process

Lesson 14: Referencing Members in Calculations

Lesson 15: Developing and Testing Complex Calculation Scripts

Lesson 16: Normalizing Data

Lesson 17: Creating Attribute Dimensions

Lesson 18: Analyzing Varying Attributes

Lesson 19: Analyzing Text and Dates

Preface

Welcome to Oracle Essbase 11.1.2 Bootcamp.

Before you begin, please take a moment to review this section. The preface presents an overview of the following information:

- Course objectives
- Structure of the course
- Course materials used in the class
- Conventions used in the book
- Additional resources to enhance your learning
- Relevant follow-up courses that you might want to attend in the future

Course Objectives

After completing this course, you should be able to:

- Create block storage databases
- Build rules files for dimension build and data load
- Analyze data using Smart View
- Create basic and advanced calculation scripts

Course Structure

Oracle Essbase 11.1.2 Bootcamp is a 5-day, instructor-led training course consisting of lectures, demonstrations, and hands-on exercises. In this course, the instructor presents a topic, and guides you through the exercises. Demonstrations and hands-on exercises reinforce the concepts and skills introduced during lectures.

Course Materials

You use two books in class—the student guide and the activity guide. The instructor may also give you handouts.

Student Guide

The student guide contains lessons. Each lesson begins with a list of objectives followed by the presentation of slides and accompanying textThe lesson ends with a summary of the topics covered in the lesson.

Activity Guide

The activity guide has two sections—exercises and exercise solutions.

- **Exercises**—A critical part of the learning process is the challenge of completing real tasks associated with each lesson. Each exercise is an opportunity to apply your knowledge.
- **Exercise Solutions**—The exercise solutions present the detailed steps to successfully complete the exercises.

Additionally, the activity guide has two case studies for guided self-study.

Conventions

The following text conventions are used in this course book:

- Text to be entered, options to be selected, names of files and modules, and menu selections are displayed in bold type. Examples:
  - Select **Clear Profile**.
  - Click **YES** to clear the profile.
- When available, figures are used to identify an object or task. Example:

  Click .
- Keyboard shortcuts are displayed as follows: Ctrl+Enter
- Alerts are used to direct your attention to different types of information.

NOTE: A note provides related information, common mistakes, or cautions about the current topic.

TIP: A tip provides information that helps you work more efficiently.

Additional Resources

Use the following resources to enhance your learning:

- Oracle Essbase 11.1.2 Database Administrator's Guide
- Oracle Hyperion Smart View for Office, Fusion Edition 11.1.2 User's Guide
- Oracle Technology Network (http://www.oracle.com/technology/)

Oracle provides the following user assistance with Enterprise Performance Management products:

- **Context-sensitive help**—Click **Help** for context-sensitive help.
- **Help menu**—From the Help menu in the software, access screen-level help, general product help, Oracle Support Web site, Oracle Technology Network (OTN), and oracle.com.
- **MetaLink**—Access release-specific Readme files.
- **E-Delivery**—Access release-specific installation documentation before downloading software.
- **Oracle Technology Network**—Explore product documentation, get started with Java, PHP, Linux, and other industry-standard technologies,download free software, and read technical articles and notes authored by OTN members. You can also join discussion forums to request advice from Oracle engineers and other OTN members; listen to podcast interviews with Oracle engineers, customers, and partners; bookmark Technology and Developer Centers devoted to your area of interest; and subscribe to Developer e-mail newsletters.
- **OTN Documentation Library**—Download documentation for all Enterprise Performance Management products, including reference information and PDF and HTML versions of each deliverable.

Related Courses

The following courses are available:

- Oracle Essbase 11.1.2: Calculate Databases
- Oracle Essbase Studio 11.1.2: Create and Manage Data Structures

- Oracle Essbase 11.1.2: Deploy Aggregate Storage Databases
- Oracle Essbase 11.1.2 for System Administrators
- Oracle Hyperion Smart View, Fusion Edition 11.1.2 for Essbase

NOTE: Course names and learner paths may change. Visit www.oracle.com/education for the latest information.

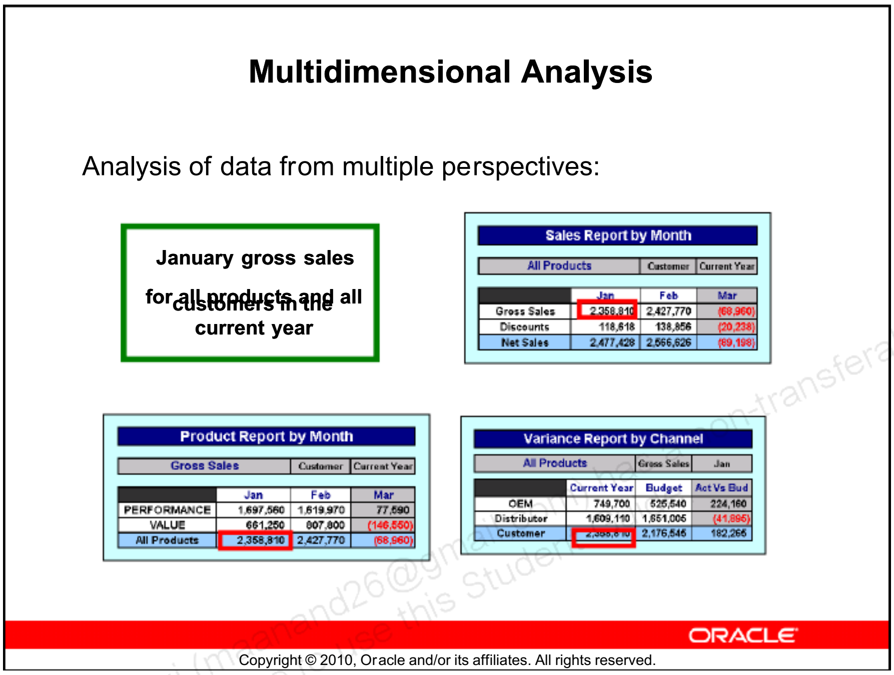
LESSON 1

Essbase Overview

Objectives

At the end of this lesson, you should be able to:

- Explain multidimensional analysis
- Discuss Oracle’s Enterprise Performance Management System
- Describe Oracle Business Intelligence Suite Enterprise Edition Plus
- Describe Essbase architecture and components
- Identify components of the Essbase production environment



Multidimensional Analysis

Information is one of the most important assets of a business. Corporate information analysis offers a road map for strategic planning and enables executive managers to make informed and timely decisions.

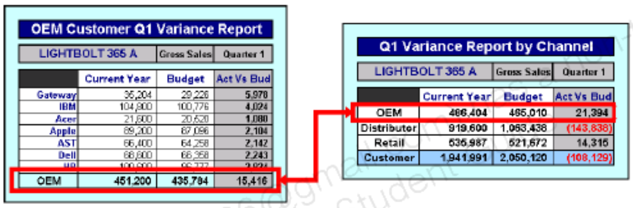
Information analysis provides insight into the past, present, and future of your business; an understanding of past performance helps you address current concerns, which in turn helps you plan for future growth. To gain business intelligence, you must analyze data from many perspectives. For example, it is not enough to know the sales totals for a month; you must know which customers bought the most and which products sold the least.

The slide shows a sales report by month, a report of product sales by month, and an actual versus budget variance report by customer channel. Although each report has a different focus, all three reports include data for January gross sales for all products and all customers in the current year. Multidimensional analysis tools such as Essbase provide the means to view the same data from any business perspective.

Spreadsheet-Based Analysis

- Data integrity suffers because of user errors or stale data.
- Validating spreadsheet reports wastes time and resources.

Two reports prepared by two users showtwo different totals. Which one is correct?



Spreadsheet-Based Analysis

Most businesses have many transactional systems that contain huge numbers of daily transactions. These fragmented systems force companies, most commonly using spreadsheet programs, to spend too much time analyzing disconnected environments. In such environments, spreadsheet programs are extremely cumbersome, if not unworkable, analysis tools.

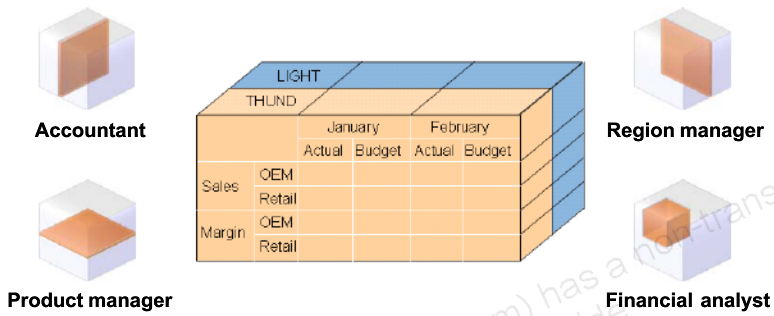
File corruption, user errors, and lack of data synchronization across disparate systems often lead to multiple versions of one truth. Businesses deplete their resources developing multiple spreadsheets to answer multidimensional questions, and then waste substantial time and manpower trying to validate reports against each other.

The slide shows two reports prepared by different users. Unfortunately, the reports do not display the same totals, so both reports must be put through a lengthy validation

process to determine which total is correct.

Legislation and other statutory reporting requirements place enormous pressure on companies to report with absolute accuracy and, at the same time, the use of spreadsheet programs for data analysis is a requirement for most businesses. Therefore, organizations require one system that not only satisfies their global reporting requirements but also provides the multidimensional spreadsheet-based analysis that executives need to make fast, accurate decisions. Essbase was created with these issues in mind.

Multidimensional View of Information



Multidimensional View of Information

In a multidimensional data model, enterprise strategic information is structured around natural business concepts and can be visually represented as a multidimensional array, instead of structuring data in a series of relational tables. You can view data from different perspectives, depending on analytic needs. A multidimensional data model provides a foundation for efficient, sophisticated business analysis.

OLAP

The term *OLAP* (online analytical processing) describes analysis tools that provide fast multidimensional analysis of information, enabling businesses to turn the wealth of data generated by transactional applications into usable business intelligence. Both Oracle OLAP and Oracle© Essbase are OLAP tools.

Benefits of OLAP

OLAP supports the natural tendency of users to view business results as facts organized by various dimensions. Implemented in a multiple-user, client-server, multitier environment, OLAP helps you synthesize enterprise strategic information through comparative, personalized viewing and analysis of historical and projected data.

Essbase and Oracle OLAP

Essbase and Oracle OLAP, the two most capable options in the OLAP market, comprise Oracle's multidimensional solution. Each product provides different OLAP solutions to meet your analytic needs. For complete coverage of the similarities and differences between Essbase and Oracle OLAP, read *Oracle Essbase and Oracle OLAP: The Guide to Oracle's Multidimensional Solution* by Michael Schrader, Dan Vlamis, Mike Nader, Chris Claterbos, Dave Collins, Mitch Campbell, and Floyd Conrad, published by Oracle Press.

Data Cubes

- Analysts prefer to view multidimensional data in cubes, rather than in relational tables.

Relational table

Product	Time	Sales
Lightbolt	Jan	166
Lightbolt	Feb	182
Thunderball	Jan	131
Thunderball	Feb	149

Data cube

	Jan	Feb
Lightbolt	166	182
Thunderball	131	149

1 measure: Sales
2 dimensions: Time, Product

- *Cube* is a metaphor for multidimensional data storage.
- Data is analyzed along only two to three dimensions at a time.

ORACLE

Data Cubes

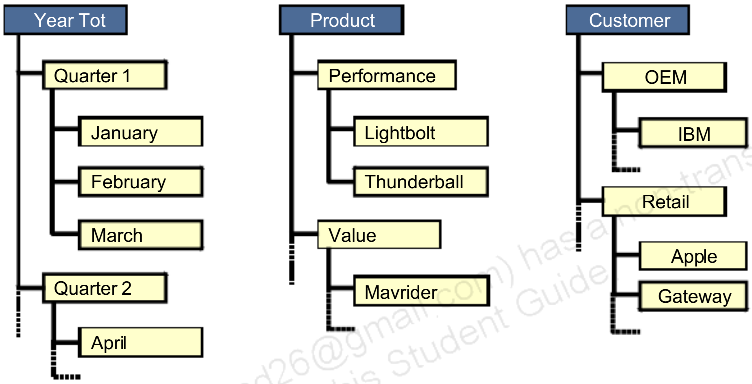
A data cube is a multidimensional extension of a two-dimensional table, just as a geometrical cube is a three-dimensional extension of a square. In OLAP tools, the term *cube* is a metaphor for multidimensional data storage.

A data cube is a multidimensional matrix that facilitates analysis from many perspectives. You can visualize a three-dimensional data cube as a spreadsheet program workbook that includes identically structured spreadsheets (each representing two dimensions) on multiple uniquely named tabs (representing the third dimension). Each tab represents a data set that is contained within the structure of the spreadsheets.

Although data cubes are not restricted to three dimensions, visualizing dimensional cubes of more than three dimensions in spatial or geometrical terms is difficult. For analysis purposes, you usually view details for only two or three dimensions at a time, while the data cube is indexed along the remaining dimensions. Analysts typically prefer this multidimensional representation over a table format.

Dimension Hierarchies

You can organize a dimension's members in a hierarchy.



Dimension Hierarchies

in a multidimensional data model, the structural relationships between members within a dimension define the dimension hierarchy. A typical dimension consists of one or more members that, in turn, may consist of other members. For example, one possible hierarchy in a time dimension is Year, then Quarter, then Month, then Day, as shown on the slide. Hierarchies also define relationships that users see during analytic operations.

NOTE: *Parents* are members that consist of other members and define consolidation levels in the dimension hierarchy.

Operations in Multidimensional Data Models

- Selection (**slice** and **dice**)
- Aggregation (**roll up**)
- Navigation to more detailed data (**drill down**)
- Visualization operations (**pivot**)

Operations in Multidimensional Data Models

The multidimensional data model supports the following cube operations.

Slicing and Dicing

A dimension acts as an index for identifying values in the multidimensional array. Slicing and dicing are operations that select data from the data cube. In slicing, you select one member from all but two dimensions, and the remaining dimensions define a two-dimensional slice of the data cube. In dicing, you select a range of members from more than two dimensions, and the selection defines a multidimensional subcube.

Rolling Up and Drilling Down

You can roll up (summarize the data cube) by traversing upward through the dimension hierarchy. As values are combined, cardinalities shrink, and the cube view becomes smaller.

For example, if all products are displayed, you can collapse the dimension hierarchy to view the total for all products at the top of the dimension.

Drilling down is the reverse of rolling up. You traverse from summarized data to detailed data. To incorporate more detail into your analysis, you can drill down a dimension hierarchy or add another dimension to your analysis.

Pivoting

By rotating the data cube, you can view analytic data from multiple perspectives. You typically pivot to build a custom, two-dimensional analytic report.

Sparse and Dense Dimensions

Multidimensional data sets tend to be sparse.

- Data is not uniformly distributed.
- Data does not exist for the majorit y of member combinations.
- Certain aspects of multidimensional data sets tend to be predictably sparser than others.

ORACLE

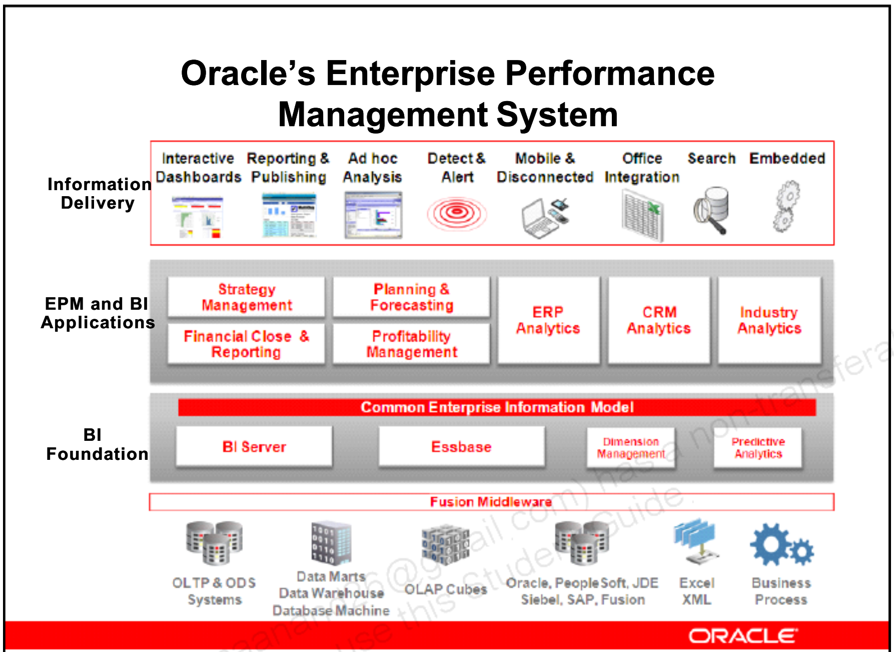
Sparse and Dense Dimensions

The data sets of most multidimensional data models have two characteristics:

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

Most multidimensional data sets 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.

Some dimensions in a multidimensional data model may be denser than others. A *dense dimension* is a dimension with a high percentage of available data positions filled.



Oracle's Enterprise Performance Management System

Oracle's Enterprise Performance Management System (EPM System) is a complete, open, and integrated system that supports a broad range of analytic requirements. It includes three layers of capabilities—information delivery, EPM and business intelligence (BI) applications, and a common BI foundation.

Information Delivery Layer

This layer provides a complete set of information delivery and access capabilities, which are designed to address the needs of different types of users in an organization. These capabilities include interactive dashboards for executives and managers, ad hoc analysis tools for power users, Microsoft Office interfaces for finance users, and pixel-perfect reports and mobile support for casual users.

EPM and BI Applications Layer

This layer includes an integrated suite of market-leading performance management applications that are based on the Hyperion product suite. The applications address key strategic and financial performance management processes, including strategy management, planning and forecasting, financial close and reporting, and profitability management.

This layer also includes integrated BI applications that can help users analyze data from enterprise resource planning (ERP) and customer relationship management (CRM) applications, as well as support industry-specific requirements.

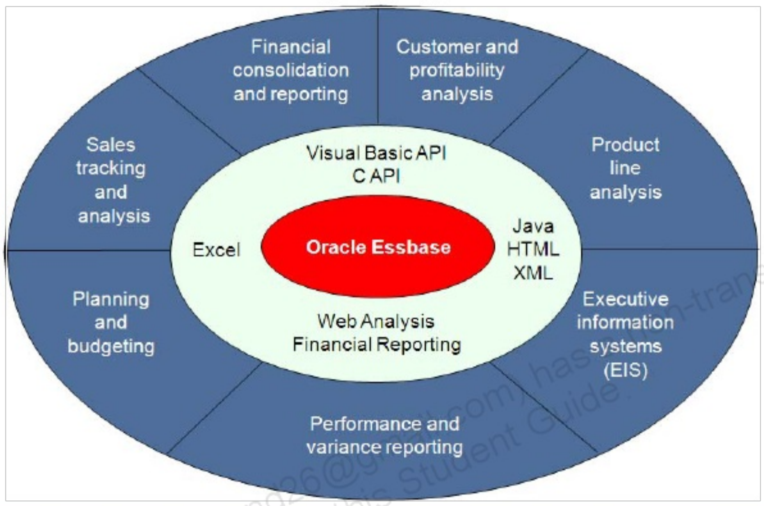
BI Foundation

The EPM and BI applications are integrated on a BI foundation that includes the following:

- Common enterprise information model
- Powerful, forward-looking analytics with Oracle© Essbase
- Ability to integrate data from Oracle and non–Oracle databases and transactional systems
- Enterprise dimension management application, which provides a single point of management for dimensions and hierarchies across EPM, BI, data warehouse, and other applications
- Predictive analytics engine, which integrates BI with business processes

The BI foundation also leverages key technologies from Oracle Fusion Middleware, such as data integration, identity management, and process management.

Essbase



Essbase

Most traditional database products are application-specific, and the proliferation of application-specific packages results in multiple support for training and development.

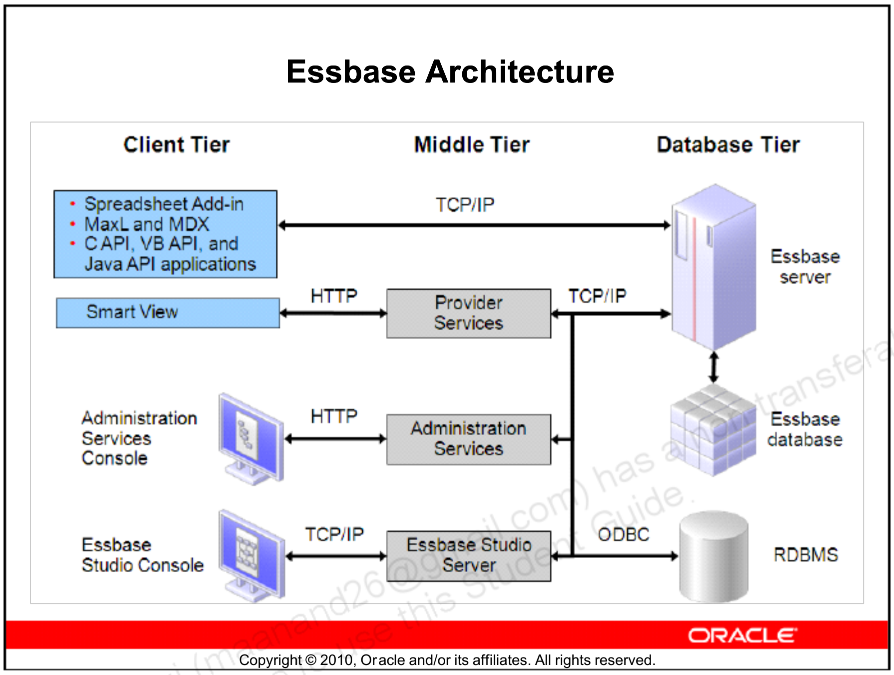
With Essbase as an application development environment, you can use one tool to build multiple databases. Individual applications are better integrated with each other. Only one environment needs to be supported for development, deployment, and training.

Essbase enables the quick and easy implementation of analytic solutions, adds value to previously inaccessible data, and transforms data into actionable information.

Here are some examples of the types of analysis that you can implement with Essbase:

- Financial consolidation and reporting
- Customer and profitability analysis
- Product line analysis

- Executive information systems (EIS)
- Performance and variance reporting
- Planning and budgeting
- Sales tracking and analysis



Essbase Architecture

The Essbase product family features a middle-tier architecture to handle a wide range of analytic applications across large, multiple-user environments:

- The database tier consists of the Essbase server (where Essbase databases are stored) and any relational databases that you are using to support the Essbase environment.
- The client tier includes locally installed client applications, such as Essbase Administration Services Console and Oracle© Hyperion Smart View for Office, Fusion Edition.
- The middle tier includes application services that facilitate communication and data transfers between the database tier and the client tier.

Essbase Components

- Essbase:
  - Essbase server
  - Essbase databases
  - Administration Services
  - Smart View
  - Spreadsheet Add-in
  - MaxL, ESSCMD, MDX
  - C API, VB API, and Java API
- Additional components:
  - Essbase Studio
  - Integration Services

Essbase Components

Essbase is a multidimensional database software optimized for planning, analysis, and management-reporting applications. Essbase offers aggregate storage databases for operational analysis and block storage databases for financial analysis.

Essbase uniquely blends an innovative technical design with an open, multitier architecture. The product enables you to extend decision support systems beyond ad hoc queries and reports on historical performance to dynamic operational systems that combine historical analysis and future planning. Essbase enables you and others in the organization to share, access, update, and analyze enterprise data from any perspective and at any level of detail without learning new tools, query languages, or programming skills.

Essbase Server This multithreaded OLAP server takes advantage of symmetrical, multiprocessing hardware platforms. The server acts as a shared resource, handling all data storage, caching, calculations, and data security.

Essbase Database This multidimensional data cube can be implemented in two storage models: block storage and aggregate storage.

Block storage databases support dense data sets, enabling users to perform complex financial analytics.

Aggregate storage databases drive operational analytics and are optimized for high dimensionality, extreme sparsity of data, and dimensions with millions of members.

Administration Services This database and system administrator interface to Essbase provides a single-point-of-access console to multiple instances of Essbase Server. Using Administration Services Console, you can design, develop, maintain, and manage multiple instances of Essbase Server and multiple applications and databases. You can preview data within the console,without having to open a client application such as Oracle Spreadsheet Add-in. You can also use custom Java plug-ins to leverage and extend key functionality.

Smart View This software program provides a common Microsoft Office interface (Excel, Word, PowerPoint, Outlook) for Essbase, Oracle© Hyperion Financial Management, Fusion Edition, Oracle© Hyperion Planning, Fusion Edition, Oracle© Enterprise Performance Management Workspace, Fusion Edition, Oracle© Business Intelligence Enterprise Edition, Oracle's Hyperion© Reporting and Analysis, and Oracle's Hyperion® Enterprise® data sources. Using Smart View, you can view, manipulate, distribute, and share data.

Spreadsheet Add-in This legacy Excel add-in enables you to access Essbase servers and perform ad hoc reporting on Essbase databases.

MaxL This practical, expressive interface for administering the Essbase system is one of the two functional domains of the multidimensional database access language for Essbase. With MaxL Data Definition Language (DDL), you use statements to make requests; MaxL DDL statements usually begin with verbs and read like English sentences. MaxL DDL improves on ESSCMD in that you can make it interact with the operating system by issuing shell commands, and you can embed it in Perl programs by implementing the Perl module.

ESSCMD This command-line language performs server operations interactively or through batch or script files. ESSCMD is a legacy language from Essbase versions prior to 7.x, and although it is forward-compatible, it does not contain any language enhancements or additions after the 6.x releases.

MDX This multidimensional query language is the second of the two MaxL functional domains. MDX provides the ability to perform advanced data extraction and querying by means of statements that typically include the verb SELECT. You use MDX to construct member formulas in aggregate storage databases.

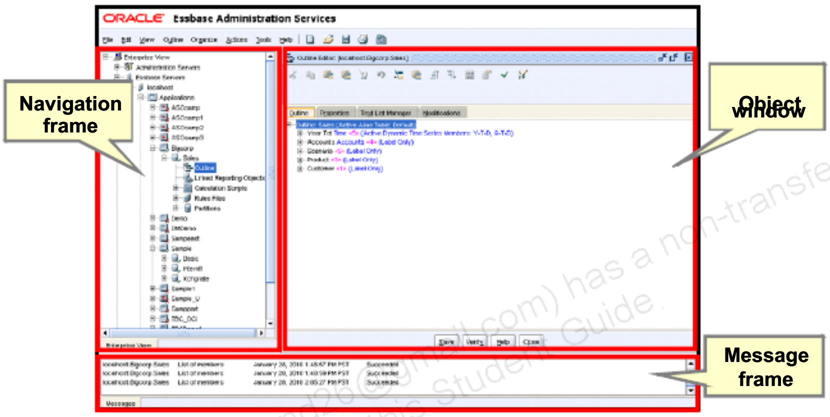
Essbase API This developers' interface to Essbase enables you to create customized applications in VB, C, or Java programming languages.

Essbase Studio This tool provides an environment for the development, deployment, and maintenance of enterprise-scale Essbase analytic applications. The Essbase Studio graphical front end enables you to build Essbase cubes from disparate data sources. It also features a built-in repository to store and reuse hierarchies in multiple cubes; lineage-tracking functionality that provides full upstream and downstream metadata visibility; and enhanced drill-through to multiple targets, including relational, Essbase, BI+ repository, URL-based, and custom Java methods.

Integration Services This legacy integration tool provides a metadata-driven environment to bridge the gap between data stored in Essbase databases and transactional data stored in relational databases. Integration Server drill-through enables business users to view linked transactional data from Essbase reports. The Hybrid Analysis feature gives business users more detail for decision making, and information technology (IT) managers more modularity in designing and maintaining large-scale analytic applications.

Administration Services Console

Interface for database and system administrators



Administration Services Console

Administration Services Console is the common administration interface for Essbase. Database and system administrators use it to manage users and user security, server options, Essbase applications and databases, and database objects. The interface relies on shortcut menus to perform most actions and is organized in three frames: navigation frame, object window frame, and message frame.

Navigation Frame

When you start Administration Services Console, the default navigation tab, Enterprise View, is displayed. Enterprise View is a graphical hierarchy view of the Essbase environment. You can expand a node of the Enterprise View hierarchy by clicking the plus sign (+) next to an object. If you double-click an object, it opens in the object window frame.

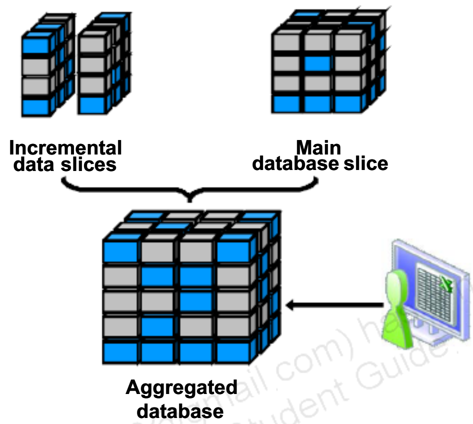
You can create custom views for specific servers, applications, or databases to reduce the number of mouse clicks required to navigate to an object in Enterprise View.

Object Window Frame
Objects are displayed in this workspace after you open them. Administration Services Console is a multiwindow environment; every object that you open stays open in the object window until you close the object. You can tile objects inside the object window or switch among objects by using the Organize menu commands. Object toolbars are displayed inside their respective objects. The main toolbar displays shortcuts, for example, for opening and saving files.

Message Frame
The message frame displays system information messages. When you check the formula syntax of an outline member formula, verification errors are displayed in the message panel.

NOTE: To disable the navigation panel or the message panel, use the View menu.

Aggregate Storage Databases



Aggregate Storage Databases

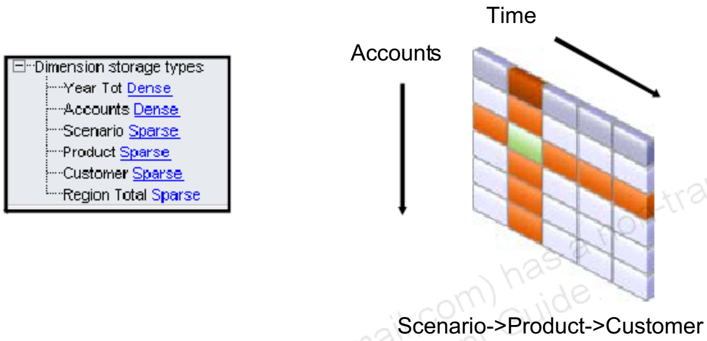
Aggregate storage databases are optimized for sparse data sets that primarily require simple aggregation. All calculations are built into the database outline and calculated on demand, as users query the database.

Additionally, incremental data loading and fast aggregations can provide near real-time analysis of transactional data.

The following examples are business models that could be well suited for analysis in aggregate storage databases:

- Customer analysis (many dimensions, millions of customers)
- Procurement analysis (many products across many customers)
- Logistics analysis (near real-time information on product shipments)

Block Storage Databases



ORACLE

Block Storage Databases

Block storage databases are optimized for data sets that are partially dense. Data is stored in dense data blocks, which are indexed along sparse dimensions for retrieval. This unique storage paradigm enables you to perform top-down budgeting and planning in addition to sophisticated preaggregation calculations.

The following examples are business models that could be well suited for analysis in block storage databases:

- Sales forecasting (top-down planning and allocations)
- Profitability analysis (cost allocations across products and customers)
- Financial consolidations (currency conversions, intercompany eliminations)

NOTE: For a comprehensive overview on selecting the best database type for your analytic needs, see“Selecting a Database Type” on page 2-9

Production Environment Components

- Database objects
  - Outline (OTL)
  - Rules files (RUL)
  - Calculation or aggregation sc ripts (CSC)
- Smart View
- Essbase Studio

ORACLE

Production Environment Components

The Essbase database production environment consists of three principal database objects for defining and managing your databases:

- Outline (OTL)
- Rules files (RUL)
- Calculation or aggregation scripts (CSC)

Essbase also provides the Smart View client interface for viewing and analyzing data in Microsoft Office and Essbase Studio for integrating your Essbase environment with relational and other data sources.

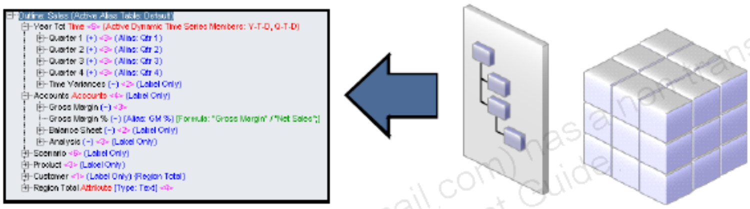
The process of creating and maintaining a database is largely the same for both database types. The process of creating a production database, at its most basic level, demonstrates the relationships among components:

1. Create a database outline, defining database dimensions and hierarchies. Outline files have the extension OTL.
2. Load data using rules files to map to the database dimensions. Rules files have the extension RUL.
3. Store calculated data using calculation scripts (block storage) or aggregation scripts (aggregate storage). Calculation and aggregation scripts have the extension CSC.
4. Analyze calculated data using Smart View.

NOTE: Alternatively, for an approach that is more integrated with your source data, use Essbase Studio to create outlines and load data, and to support transactional-level detail in your reporting and analysis.

Outlines

- Tree structure for dimension hierarchies
- Consolidations and mathematical relationships between members
- Outline Editor



ORACLE

Outlines

The database outline plays the key role in the life cycle of Essbase database design. Database development begins with creation of a database outline.

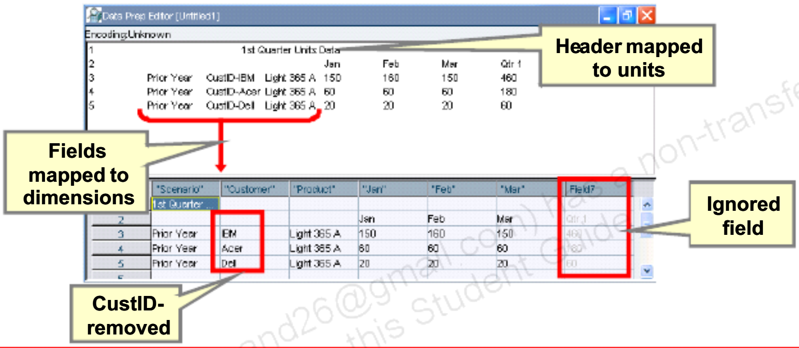
The outline defines the database dimensions—hierarchies that describe the structural relationships between members. The tree structure of the outline enables you to define consolidations between members. For example, a dimension represents the highest consolidation level in the database outline.

Additionally, you can define member formulas in the outline. In aggregate storage databases, all database calculations are defined in the outline. Block storage databases provide for defining calculations in the outline or in external calculation scripts.

To create and modify the database outline, you use Outline Editor in Administration Services Console.

Rules Files

- Map external data sources to a da tabase outline
- Can apply to many data sources
- Can load data and build outlines



Rules Files

Rules files are the native Essbase tools for mapping external data sources to Essbase databases. They provide powerful features for formatting and processing external data, both for building dimensions and loading data. Using rules files, you can load data manually or use MaxL to automate batch data loading. There are no fundamental restrictions on the size of data source files or on the number of records that can be loaded using rules files.

Rules files can process data in any of the following ways:

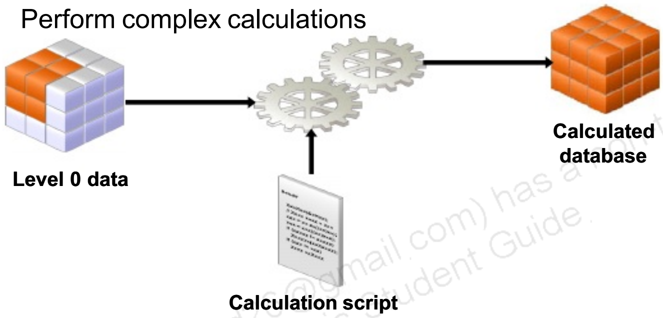
- Ignore certain fields or records in the data source
- Manage header information, such as ignoring extraneous headers or setting up special purpose headers for label identification
- Change the order of fields by moving, joining, splitting, or creating fields
- Resolve problems of member uniqueness by replacing member names with valid values

- Scale data values
- Overwrite loaded or calculated values
- Update unrecognized new members without creating error conditions
- Set header records for missing data values
- Reject invalid records and continue the data load

Rules files contain a set of operations that Essbase performs on data when it loads the associated data source into the database. Rules act on data as it is loaded, without changing the data source. You can use a rules file with any data source that requires its set of data loading rules.

Calculation Scripts

Block storage only:
- Calculate all or a portion of t he database
- Control the order in which dimensions are calculated
- Perform complex calculations



Calculation Scripts

A calculation script contains a series of calculation commands, equations, and formulas. You can use calculation scripts in block storage databases to define calculations other than those defined by the database outline.

You can use calculation scripts to specify exactly how you want Essbase to calculate databases. For example, you can calculate part of a database, copy data values between members, or define calculations other than the calculations defined by the database outline. Calculation scripts also enable you to control the order in which dimensions are calculated.

You can design and run custom database calculations quickly by separating complex calculation logic from the database outline.

The following calculation script calculates only Actual values:

```
FIX (Actual)

   CALC DIM(Year, Measures, Market, Product);

ENDFIX
```

The following calculation script is an example of a conditional calculation:

```
SET UPDATECALC OFF;

COGS
(
IF (@ISMBR(Forecast))

   COGS = Sales*.25;

ENDIF
)

CALC ALL;
```

This script calculates forecast cost of goods sold (COGS) as 25% of forecasted sales.

Aggregation Scripts

Aggregate storage only:

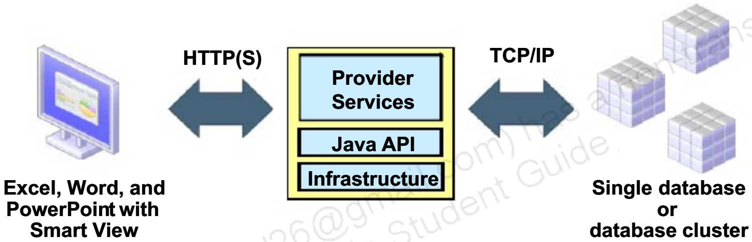
- Improved query performance
- System-generated

Aggregation Scripts

In aggregate storage databases, all database calculations are defined in the database outline and calculated on demand, as users query the database. Aggregations improve query performance by storing some of the calculated data in advance. Aggregation scripts store the options selected by an administrator using Aggregation Design wizard. Although they share the same file extension as calculation scripts (CSC), aggregation scripts are system-generated and are not editable.

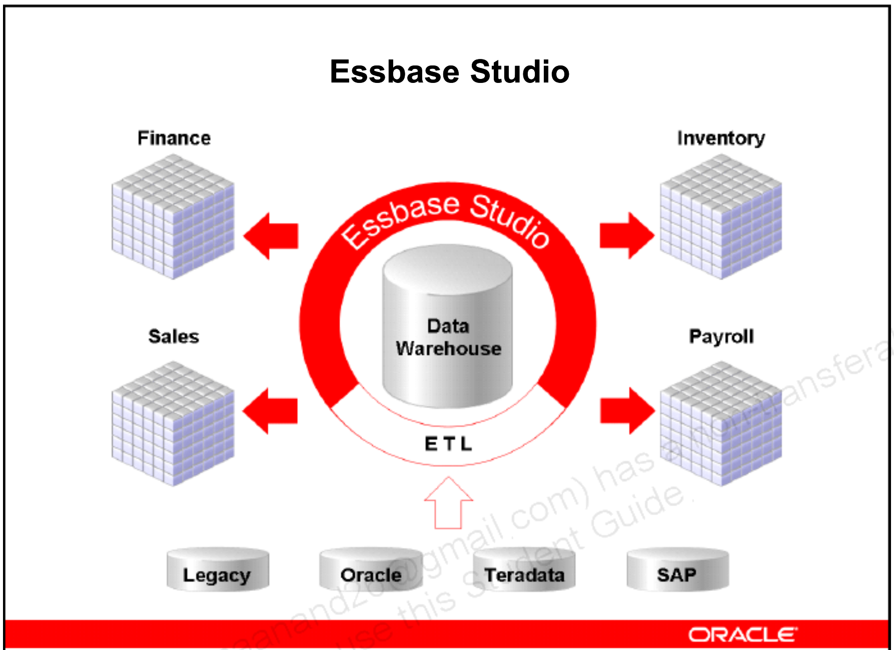
Smart View

- Enables you to retrieve data and create ad hoc reports
- Provides integration with Microsoft Office products
- Provides a single Excel interface to analyze data



Smart View

Smart View is a Web-deployed, thin-client software program that is embedded in Microsoft Office applications. Additionally, Smart View offers an intuitive user interface, and provides a common Microsoft Office interface for EPM System applications. Using Smart View, you can view, import, manipulate, distribute and share data from EPM System data sources in Microsoft Excel, Word, Outlook, and PowerPoint - data entry, analysis, and reporting.



Essbase Studio

Prior to release 11.1.1, users could create links to transactional data in relational data warehouses by using a graphical suite of tools called Integration Services.

Release 11.1.1 introduced Essbase Studio. Combining much of the functionality of Integration Services and Administration Services into a single workbench, Essbase Studio is the next-generation environment for the development, deployment, and maintenance of enterprise-scale Essbase analytic applications. Essbase Studio graphical front-end features include:

- The ability to build Essbase cubes from snowflake and star schema relational data sources, as well as from traditional data sources.
- A built-in repository that enables you to store and reuse data elements and hierarchies in multiple cubes.
- Lineage tracking functionality, providing full upstream and downstream metadata visibility. enhanced drill-through to multiple targets, including relational, Essbase, reporting and analysis tools repositories, URL-based, and custom Java methods—with reports based on any member selection defined in the data source.

Summary

In this lesson, you should have learned to:

- Explain multidimensional analysis
- Discuss Oracle's Enterprise Performance Management System
- Describe Oracle Business Intelligence Suite Enterprise Edition Plus
- Describe Essbase architecture and components
- Identify components of the Essbase production environment

LESSON 2

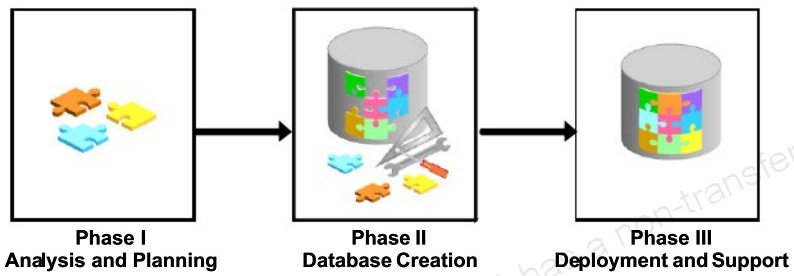
Designing Applications and Databases

Objectives

At the end of this lesson, you should be able to:

- Describe the Essbase implementation process
- Analyze and plan implementations
- Create block storage applications and databases
- Create block storage outlines
- Modify member properties

Essbase Implementation Process



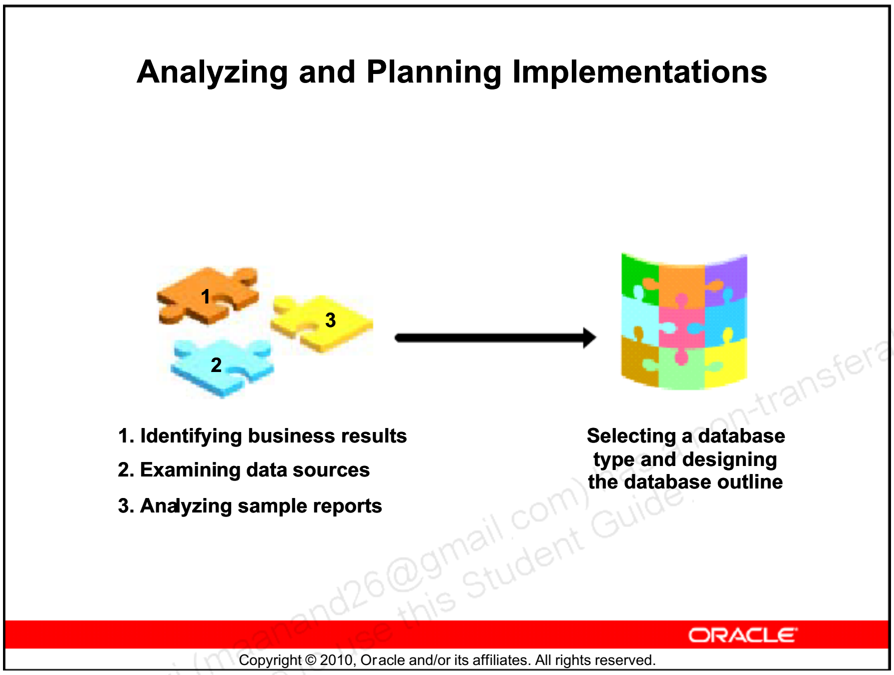
Essbase Implementation Process

An Essbase database implementation includes many steps, which can be roughly broken down into three phases: analysis and planning, database creation, and deployment and support.

The implementation process is iterative. Analysis of the results of one cycle may stimulate new questions, leading to newly defined business information requirements. New requirements may lead to new designs and implementations of the process.

The following table contains a set of guidelines and suggested steps for planning, creating, and deploying Essbase databases:

Implementation Phase	Implementation Steps
Analysis and Planning	1. Identify business results. 2. Examine data sources. 3. Analyze sample reports. 4. Select a database type and design the database outline.
Database Creation	5. Create the database outline. 6. Create rules files. 7. Create calculation scripts or design aggregations.
Deployment and Support	8. Maintain outlines. 9. Manage data flow. 10. Analyze data. 11. Provide management and user support.



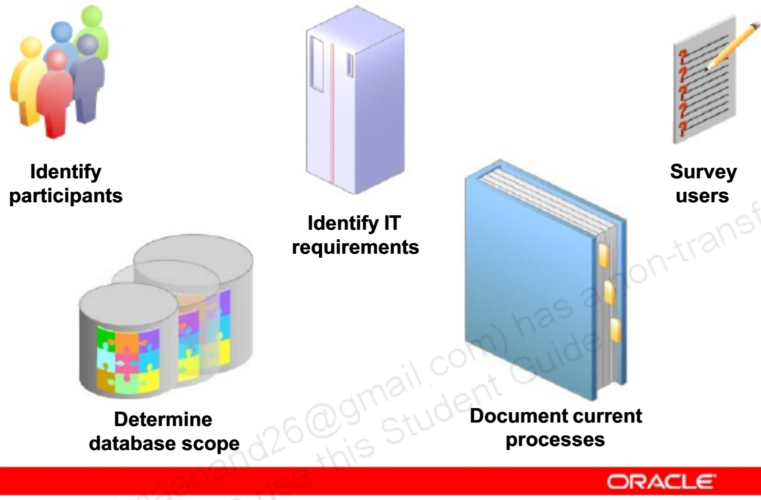
Analyzing and Planning Implementations

The design and operation of an Essbase multidimensional database plays a key role in creating a well-tuned system that enables you to analyze business information efficiently. A detailed plan that outlines data sources, user needs, and prospective database elements can save you development and implementation time.

As shown in the example on the slide, during the analysis and planning phase of implementation, you start by gathering information—you identify business results, examine data sources, and analyze sample reports.

During the fourth step, you synthesize the collected information into a working design by selecting a database type and designing the database outline.

Identifying Business Results



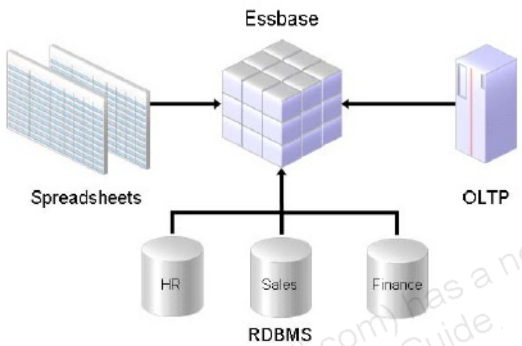
Identifying Business Results

This step of the implementation process sets the groundwork for everything that follows. The following tasks are related to this step of the process:

- **Identify participants**—Identify all participants in the implementation process and establish the responsibilities, deadlines, and project scope of each group of participants.
- **Identify IT requirements**—Conduct hardware surveys to determine your available IT infrastructure and identify where hardware upgrades may be necessary.
- **Survey users**—A user analysis survey determines who will use the completed cube, how users currently analyze data, what types of analysis are required from the completed cube, and where users and administrators are located.
- **Document current processes**—Document how data is currently being processed, from data collection to final information analysis. Determine which databases, software, and client tools are currently involved in the process.

- **Determine the database scope**—Decide how much or how little detail is required for final analysis. For example, if your organization has thousands of product families containing hundreds of thousands of products, you may want to store only aggregated data values for product families, if data analysis at the product level is not required.

Examining Data Sources



ORACLE

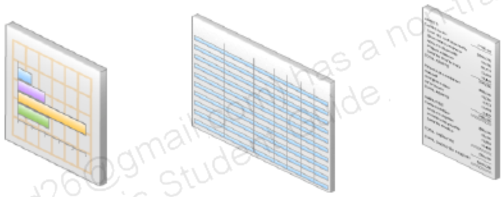
Examining Data Sources

An IT task force is typically responsible for the majority of the data examination step, in which all source data systems must be identified and data extract strategies must be developed. The following tasks are related to this step of the process:

- **Determine data sources for Essbase**—Discover where production data for the Essbase database will come from.
- **Document data sources**—Document the types of data sources involved and where they are physically located.
- **Evaluate data transformation needs**—Determine if data is in a form that Essbase can use, and if not, how data will be transformed to a usable format.
- **Document planned data flow**—For each data source, document user responsibilities for transferring data into Essbase. For example, current actuals are transferred programmatically from the Oracle G/L database through Essbase Studio into Essbase, but end users load forecast data directly to Essbase using Smart View.

Analyzing Sample Reports

Sample reports:
- Represent both ad hoc and production-style reports
- Provide insight into the number, type, and granularity of dimensions
- Ensure a database design with usable results
- Provide the best perspective for creating hierarchy relationships



Analyzing Sample Reports

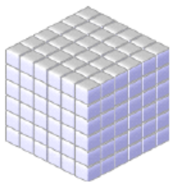
Data source analysis provides you with information about data in its raw form, but analysis of sample reports (that is, reports that users want to create using Essbase data) provides a crucial look at the final requirements and architecture of the database.

Given a set of sample reports, representing both ad hoc and production-style reports, a design team can determine how many database dimensions are required, how much detail should be included, and what kind of hierarchy relationships are required to provide users with the details that they need.

During this step of implementation, a design team makes decisions about which features and attributes are necessary and which features and attributes are desirable. Finalizing a feature-attribute list helps set expectations for the final design and for the users. This step ensures a database design with usable results.

Selecting a Database Type

- Select aggregate storage by default
- Select block storage for certain analysis needs



Aggregate storage

Block storage

Selecting a Database Type

In general, aggregate storage should be your default database type. Aggregate storage databases are simpler to maintain and tune, require less disk space, and provide better aggregation performance than block storage databases. Most multidimensional data sets are very sparse and require a great deal of simple aggregation, making them good candidates for aggregate storage.

Block storage databases typically perform best when they contain fewer than 10 dimensions, and when measures and time dimensions are densely populated with data. Block storage models are typically implemented for financial models, because they fulfill certain analysis needs that are not possible in the aggregate storage architecture:

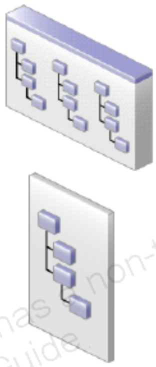
- Top-down planning (requiring data to be loaded to upper-level members)
- Preaggregation calculations, such as intercompany eliminations or certain types of allocations
- Currency conversion

- Calculations based on input drivers, such as *Units x Price = Sales*, where *Price* is an input value
- Planning implementations (Planning requires block storage databases for back-end data storage.)

If your analysis needs match any of the preceding scenarios, consider implementing a block storage database. For example, Bigcorp data analysis involves only five dimensions, of which the measures and time dimensions are densely populated with data. Additionally, Bigcorp requires top-down loading and allocation of budget data (involving external calculation scripts for allocation) and calculations based on input drivers (Units * List Price = Gross Sales). For this type of analysis, block storage is the best solution.

Designing Outlines

- Number of dimensions
- Dimension hierarchies
- Metadata granularity
- Database time span
- Functional scenarios
- Mathematical requirements



Design document

Designing Outlines

Outline design is a synthesis of all previously gathered research. During the outline design process, you make the final decisions about the number of dimensions, the depth and nature of outline hierarchies, and the database calculation requirements.

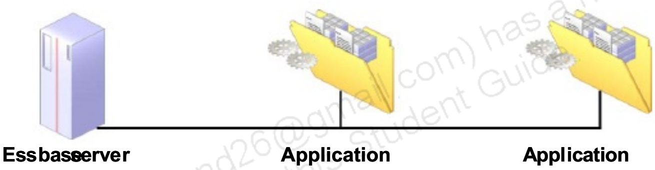
The design team documents the preliminary outline design on paper, clearly defining the following information:

- Number of dimensions
- Dimension hierarchies
- Metadata granularity
- Database time span
- Functional scenarios
- Mathematical requirements for measures data

A design document also usually includes the process and frequency for updates to both data and metadata (outline structures), as well as the process for automating and backing up the database.

Creating Applications and Databases

- Applications
  - Contain databases
  - Reside on the Essbase server
  - Run application server processes
- Databases
  - Are repositories for multidimensional analytic data
  - Contain database objects



Creating Applications and Databases

Applications and databases provide Essbase with a structure for organizing processes and related files.

Applications

An Essbase application is a management structure that contains one or more databases and related files. Applications reside on the Essbase server, and one server can store multiple applications.

The main Essbase server process (ESSBASE), also called the Essbase server agent, handles user logins and security and acts as a traffic controller for all other server requests.

Essbase applications run as an ESSSVR process controlled by the Essbase server agent. When you start Essbase applications, the Essbase server agent loads it and all associated databases into memory on the server. All client requests for data, such as data loads, calculations, reports, and spreadsheet queries, are then handled through the application process. Multiple application processes can be run concurrently by one Essbase Server agent. If a block storage application contains multiple databases, all requests to the databases are managed by the same ESSSVR process.

NOTE: Aggregate storage applications can contain only one database.

When you stop the application process, the Essbase Server agent unloads all application information and databases from its memory, and closes the application ESSSVR process.

TIP: You may want to start an application process before users connect to the application's databases; if you do so, users may experience improved performance when they connect to databases because the application and all associated databases are in memory. Try to manage your server resources by starting only the application processes that receive heavy user traffic.

Unicode Applications

The Unicode standard was developed to enable computers with different locales to share character data. Unicode provides encoding forms with thousands of bit combinations, enough to support the character sets of multiple languages simultaneously. An application is either a Unicode-mode application or a non-Unicode-mode application. When choosing between Unicode and non-Unicode, keep the following in mind:

- Non-Unicode-mode applications support only one character set, which is defined by a locale value. The locale value for the Essbase server must match the locale value for all clients that work with non-Unicode-mode applications.
- Unicode-mode applications support multiple character sets. The locale value for the Essbase server does not have to match the locale value for the clients that work with Unicode-mode applications.
- By default, Essbase creates applications in non-Unicode mode.
- Prior to release 11.1.1, aggregate storage applications did not support the Unicode-mode option.

- Smart View supports both Unicode-mode and non-Unicode-mode applications.

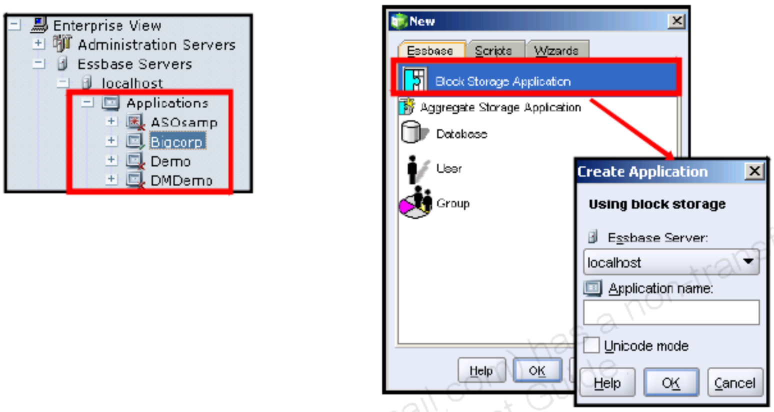
NOTE: For additional information about the Unicode standard, see `www.unicode.org`. For additional information about Unicode-mode applications, see the *Oracle Essbase Database Administrator's Guide*.

Databases

An Essbase database is a data repository that contains a multidimensional data storage array.

Essbase databases consist of database objects that define or perform actions against the database, such as outlines or rules files, and a proprietary file space to store physical data. By default, database objects are stored in their associated database folder on the Essbase server. You can also save objects to a client computer or another available network directory. However, you cannot store, load, or calculate data on a client computer.

Creating Block Storage Applications



Creating Block Storage Applications

Before you create a block storage database, you need to create a block storage application to contain the database and other objects.

To create block storage applications:

1. In Administration Services Console, select **File**, and then **New**.

   The New dialog box is displayed.

2. On the Essbase tab, select **Block Storage Application**, and then click **OK**.

   The Create Application dialog box is displayed.

3. Perform the following tasks:

   - From the Essbase Server drop-down list, select a server name.
   - In the Application name text box, enter the name for the new application.

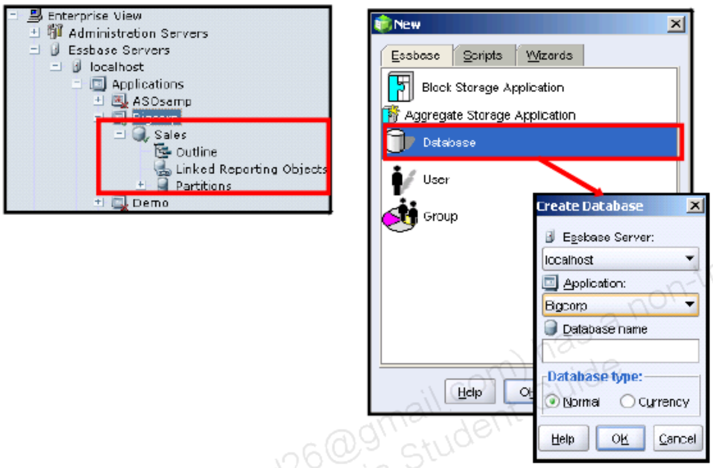
- **Optional:** if you want to support Unicode characters in your application, select **Unicode mode** to create the application as a Unicode-mode application.

NOTE: You cannot undo the Unicode-mode option.

- Click **OK**.

  Essbase creates the application and updates the Enterprise View tab.

Creating Block Storage Databases



ORACLE

Creating Block Storage Databases

You must create an application before you create databases.

To create block storage databases:

1. In Administration Services Console, select **File**, and then **New**.

   The New dialog box is displayed.

2. On the Essbase tab, select **Database**, and then click **OK**.

   The Create Database dialog box is displayed.

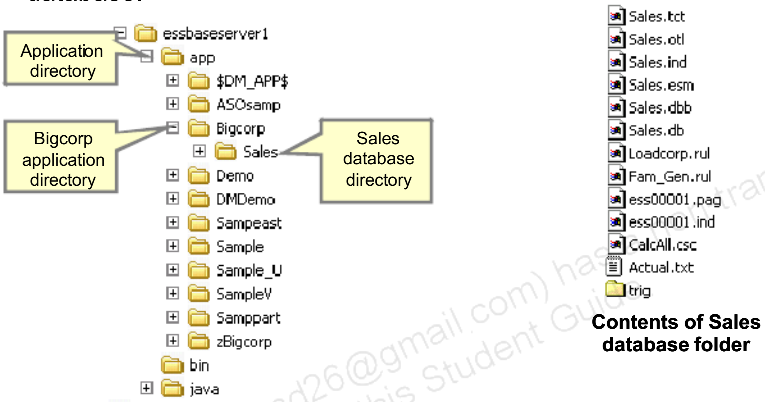
3. Perform the following tasks:
   - From the Essbase Server drop-down list, select a server name.
   - From the Application drop-down list, select an application name.

- In the “Database name” text box, enter the name for the new database.
- Select a database type: **Normal** or **Currency**.
- Click **OK**.

  Essbase creates the database and updates the Enterprise View tab.

Server File Structure

Essbase creates a directory f or each application and database.



Server File Structure

When you create a block storage application, Essbase creates a subdirectory for the application on the Essbase server in the `%ORACLEHOME%\user_projects\epmsystem1\EssbaseServer\essbaseserver1\app` directory. The new subdirectory includes the name of the application; for example, `C:\Oracle\MiddleWare\user_projects\epmsystem1\EssbaseServer\essbaseserver1\app\Bigcorp`.

When you create a block storage database, Essbase creates a subdirectory for the database within the application directory. The new subdirectory includes the name of the database; for example, `C:\Oracle\MiddleWare\user_projects\ epmsystem1\EssbaseServer\essbaseserver1\app\Bigcorp\Sales`.

Design Recommendations

One database per application is recommended in a production environment.

- Server resources are more balanced.
- Messages are logged at the application leve l, not at the database level.
- If an application becomes u navailable, all d atabases of the application become unavailable.
- There are exceptions to the rec ommendation:
  - Planning applications
  - Currency databases

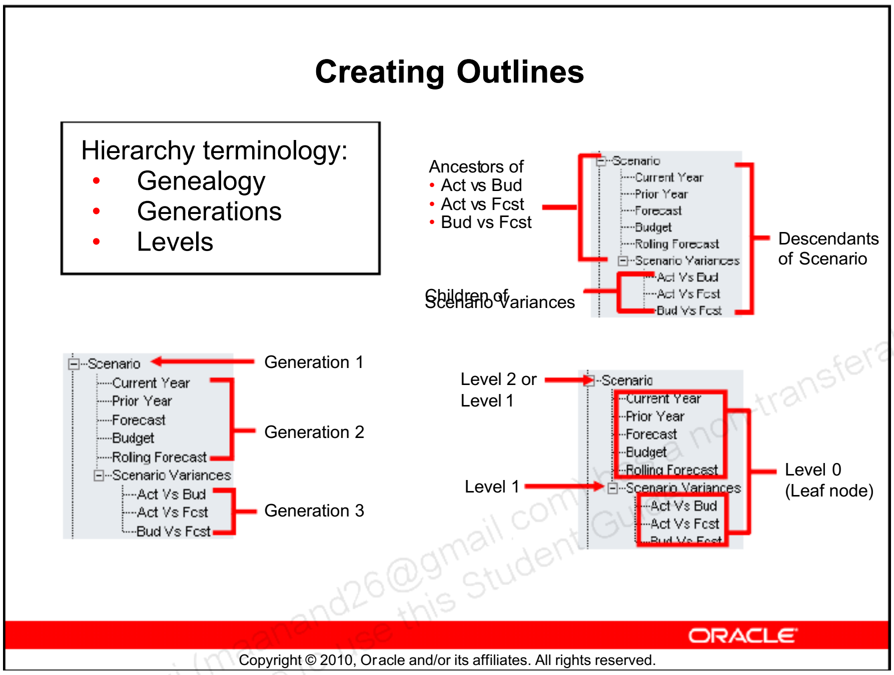
Design Recommendations

Although block storage applications accept multiple databases per application, Oracle recommends that you create only one database for each application for the following reasons:

- To manage server resources effectively, you can balance the database load across multiple, independent application processes, rather than using one application process to manage requests from multiple databases.
- During processes, Essbase logs messages only at the application level. There are no database-level process logs. If multiple databases exist in one application, Essbase logs all process messages for all databases in an application to one log file. Multiple database logging makes it difficult to effectively interpret the log.
- If the administrator terminates an application process or if an application process becomes unavailable, all databases of the unavailable application are inaccessible to users until the application process is restored.

There are two common exceptions to the rule of one database per application:

- Planning applications with multiple plan types automatically create a single block storage application with multiple databases.
- If you are using the currency conversion option for your block storage databases, you must create a primary database (where both input and converted data is stored) and a secondary currency database (where exchange rates are stored) inside one application to leverage the currency conversion calculation scripts.



Creating Outlines

Every dimension in an outline contains a hierarchy of members, with the dimension name at the top of the hierarchy. The composition of a dimension hierarchy is decided during the analysis and planning phase of the implementation; the number of levels, number of members, consolidation rules, and properties and attributes of the members vary with every database design. However, certain naming conventions—which apply to all hierarchies—facilitate reporting, calculating, security assignment, and other processes in which you need to use relationships, rather than names, to define a set of members.

Genealogy

Genealogy terms, based on relationships in a hierarchy, are defined in the following table:

Term	Definition
Member	A name at any level in the hierarchy, including dimension names
Parent	A member that has a branch below it (For example, Scenario Variances is a parent member for Act Vs Bud, Act Vs Fcst, and Bud Vs Fcst.)
Child	A member that has a parent above it (For example, Act Vs Bud, Act Vs Fcst, and Bud Vs Fcst are children of Scenario Variances.)
Siblings	Child members of the same parent and of the same generation (For example, Act Vs Bud, Act Vs Fcst, and Bud Vs Fcst are siblings.)
Descendants	All members in branches below a parent (For example, Current Year, Budget, and Scenario Variances, and the children of Scenario Variances are descendants of Scenario.)
Ancestors	All members in branches above a member (For example, Scenario Variances and Scenario are ancestors of Act Vs Bud.)
Root	The top member in a branch (For example, Scenario is the root for Current Year and its siblings and for the children of Scenario Variances.)
Leaf	Members with no children; also referred to as *detail* members, *level 0* members, and *leaf nodes* (For example, Act Vs Bud, Current Year, and Budget are leaf members.)

Generations

The term *generation* refers to a consolidation level in a dimension. The root is generation 1. Generation numbers increase as you count from the root toward the leaf.

For the example on the slide, Scenario is generation 1, Current Year and its siblings are generation 2, and the children of Scenario Variances are generation 3. All siblings of a given parent belong to the same generation.

Levels

The term *level* also refers to a consolidation level in a dimension; however, levels reverse the numerical ordering used for generations. Levels count up from the leaf member toward the root. The root level number varies, depending on the depth of the branch from which you count.

For the example on the slide, Current Year, Budget, and Act Vs Bud are level 0. All other leaf members are also level 0. Scenario Variances is level 1. Notice that the level number of Scenario varies, depending on the branch. For the Scenario Variances branch, Scenario is level 2. For the Current Year branch, Scenario is level 1.

Creating Dimensions and Members

- Add child
- Add sibling

Creating Dimensions and Members

Because implementations differ, a new database outline contains no dimensions or members. Because you decide the number and names of database dimensions during the analysis and planning phase of your implementation, the first step in creating an outline is usually to add all dimension names to the outline. After dimension names exist, you can add members to dimensions and define dimension hierarchies.

Outline Editor enables you to control how you build your hierarchies. You use the following options to determine the level and placement of members in the hierarchy:

- **Add child** adds a member one level below the selected member.
- **Add sibling** adds a member at the same level and in the same branch as the selected member.

NOTE: If the selected member is a dimension, "Add sibling" adds another dimension.

To add dimensions to an outline:

1. In Outline Editor, select the top outline node **Outline:*dbName***.
2. From the Edit menu, select **Add child**.

   A text box is displayed under the Outline:*dbName* node.
3. Enter a dimension name, and then press **Enter**.

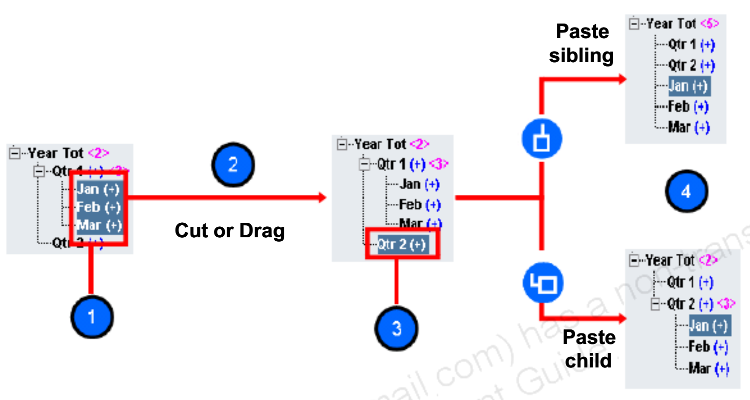
   A text box is displayed. Entering a name in the box defines another dimension.
4. Repeat step 3 to add more dimensions.
5. Press **Esc** or **Enter** to close the text box.

To add members to a dimension:

1. Select a member for the insertion point of the new member.
2. Select one of the following options:
   - From the Edit menu, select **Add child** to add a child to the selected member.
   - From the Edit menu, select **Add sibling** to add a sibling to the selected member.

   A text box is displayed.
3. Enter a member name, and press **Enter**.
4. Repeat step 3 to add more members as siblings of the first member that you added.
5. Press **Esc** or **Enter** to close the text box.

Moving Dimensions and Members



Moving Dimensions and Members

As you continue with the second implementation phase and start to build your outline, rules files, and calculation scripts, you may find it necessary to move members and dimensions. Changes may be needed because the srcinal design does not meet your requirements or because a mistake needs to be corrected.

Before you move the members and dimensions of an outline, consider the following:

- Moving dimensions and members can affect the performance of calculations and retrievals.
- If you add, delete, or move dimensions or members, Essbase restructures the database, and you must recalculate the data.
- If you move or copy a parent member, it retains all of its descendants.
- Pasting a member as a child makes it the first child of the selected member.
- Pasting a member as a sibling makes it the next sibling after the selected member.

To move members in the outline:

1. In Outline Editor, select the members that you want to move.

TIP: Press Shift to select multiple consecutive members.

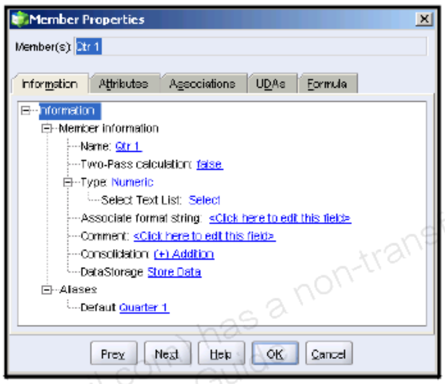
---

2. From the Edit menu, select **Cut**.
3. In the outline, select the member to which you want to relate the cut members.
4. Depending on the relationship that you want to create, perform one of the following actions:
   - From the Edit menu, select **Paste sibling**.
   - From the Edit menu, select **Paste child**.

NOTE: You can also select members and use the mouse to drag them to the preferred location in the outline. As you drag, the icons shown in the example on the slide are displayed under the cursor to indicate “Paste child” or “Paste sibling.”

Modifying Member Properties

- Data storage options
- Consolidation options
- Aliases
- Attributes
- User-defined attributes
- Member formulas



Modifying Member Properties

As you create your outline, you can assign properties to members and dimensions:

- **Data storage options**—These options control whether data is stored for the member and what data is indexed or calculated if data is not stored. The default setting is Store Data.
- **Consolidation options**—These options control hierarchy calculations. The default setting is (+) Addition.
- **Aliases**—Aliases are alternate names for the member.
- **Attributes**—Both the Attributes and Associations tabs are for assigning members of an attribute dimension to members of an associated base dimension. For a detailed discussion of attribute dimensions, see "Creating Attribute Dimensions" on

  page 17-1.

- **User-defined attributes (UDAs)**—UDAs are words or phrases about members, and are used for grouping members for calculation, security, or reporting purposes.
- **Member formulas**—These formulas are used to define special calculation requirements.

To access member properties:

1. In Outline Editor, select the member whose properties you want to change.
2. From the Edit menu, select**Edit member properties**.

Summary

In this lesson, you should have learned to:

- Describe the Essbase implementation process
- Analyze and plan implementations
- Create block storage applications and databases
- Create block storage outlines
- Modify member properties

LESSON 3

Designing Data Descriptor Dimensions

Objectives

At the end of this lesson, you should be able to:

- Describe data descriptor dimensions
- Design time, scenario, and accounts dimensions
- Explain outline calculations
- Test outline calculations

Data Descriptor Dimensions Overview

- Define data in its essential form
- Are manually developed and maintained
- Incorporate variances (both value and percentage differences)
- Are calculation-intensive
- Are densely populated with data

ORACLE

Data Descriptor Dimensions Overview

Although each Essbase database implementation is inherently unique, you can use data-measuring structures, called data descriptor dimensions, that are common to nearly all block storage data models.

From a block storage design perspective, data descriptor dimensions share the following characteristics:

- **They define data essentials.** Types of dimensions that are common to almost every block storage database outline are time, scenario, and measures (commonly called *accounts*). Almost all data fits into some kind of time scheme; most types of analysis require the comparison of data sets; and all data—whether financial account data or statistical measures—measures something.

- **They are usually developed and maintained manually**Because one time structure differs little from another time structure, administrators can copy a time dimension from one block storage outline to another with a minimum of changes. Because scenario dimensions tend to be small, there are few scenario members to create or maintain. Because the structure of a typical measures hierarchy is relatively complex, manual development is recommended.
- **They incorporate variances.** Value and percentage variance analysis most commonly occurs in scenario dimensions; however, it is not uncommon to see variances in time and accounts dimensions.
- **They are calculation-intensive.** The most complex calculation rules tend to be in measures dimensions. Calculations in other dimensions are typically simple aggregations; however, complex calculations can be incorporated into scenario and time dimensions.
- **They are usually densely populated with data.** Because data descriptor dimensions tend to have a high percentage of occupied data points, they are usually configured as dense dimensions.

Designing Time Dimensions

Decisions to make:

- Lowest level of data granularity
  - Depends on the type of analysis required
  - Can be very summarized (quarters) or very detailed (15-minute intervals)
- Number of years to represent and representation method
  - Generic time designs
  - Fiscal year crossover designs

ORACLE

Designing Time Dimensions

With few exceptions, block storage databases contain time frames for analyzing data. Variations on time dimensions from one implementation to another are largely due to differing answers to two design decisions:

- **How granular is the time dimension?** Decisions about the granularity (level of detail) of the time dimension are made according to the type of analysis required from the data set. Although most data is collected transaction by transaction, a point of consolidation usually provides the most analytic value. For financial applications, this point is usually at the monthly level; thus, most time designs for financial applications use months as the lowest level of detail. Other applications may require much more detail; for example, a restaurant may want to have a time dimension in 15-minute intervals to track sales during and around mealtimes.

NOTE: The more granular and transactional your data, the more likely your analysis is best suited to an aggregate storage model, rather than block storage. Sparsity of data typically increases as dimensions become more granular.

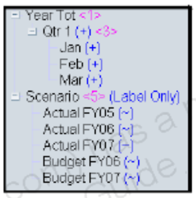
- **How many years of data are represented?** In most applications, companies want to track data year over year, so the second major design decision about how to track time concerns multiple-year analysis. Although there are many variations on time dimension design, the two most common design approaches are multidimensional time design and single-dimensional time design.

Tracking Time in Multiple Dimensions

A multidimensional time design requires at least two dimensions:

- A time dimension with generic time periods
- Discrete years represented in a scenario dimension or in a separate years dimension

Three-dimension design



Two-dimension design

ORACLE

Tracking Time in Multiple Dimensions

A multidimensional time design is the most common time model in block storage databases. This design makes the most efficient use of disk space and resources and, in most cases, provides better performance than other designs. Its primary time dimension contains time periods (days, months, quarters, and so on) with no fiscal year identification. A secondary dimension holds information about fiscal years. The two dimensions in combination provide complete information about the year and time period. Within multidimensional time designs, there are two common variations:

- Three-dimension design
- Two-dimension design

Three-Dimension Design

Time periods are represented in a dimension, fiscal years are represented in a separate year dimension, and scenarios are represented in a third dimension. This is the default time design for Planning applications.

The first example on the slide shows a three-dimension design, with generic time periods such as Jan, Feb, and Mar in the Year Tot dimension, fiscal years such as FY05, FY06, and FY07 in the Years dimension, and scenarios Actual and Budget in the Scenario dimension.

Two-Dimension Design

Time periods are represented in a dimension, and fiscal years are represented in combination with another dimension, usually a scenario dimension.

The second example on the slide shows a two-dimension design, with generic time periods such as Jan, Feb, and Mar in the Year Tot dimension, and scenario-year combinations such as Actual FY06, Actual FY07, Budget FY06, and Budget FY07 in the Scenario dimension.

Selecting a Multidimensional Design

Determining which multidimensional design to implement in your outline requires the following considerations:

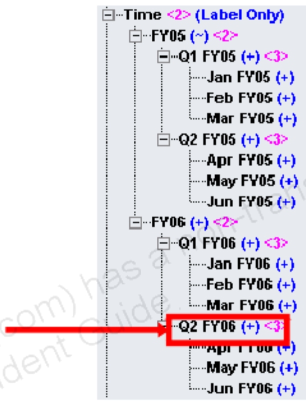
- **Reporting display and ease of use**—Do you want users to be able to reference years alone in reports? The three-dimension time design offers complete independence of each dimension and enables users to easily create crosstab reports (for example, years in rows and scenarios in columns) that may not be possible with the two-dimension design.
- **Database size and performance**—For block storage databases with more dimensions, you risk a larger database size and possible performance degradation. A two-dimension design may provide better performance than a three-dimension design.

Tracking Time in a Single Dimension

A single-dimension time design requires one of the following:

- Year-specific time periods
- Duplicate names

Year-specific time period



ORACLE

Tracking Time in a Single Dimension

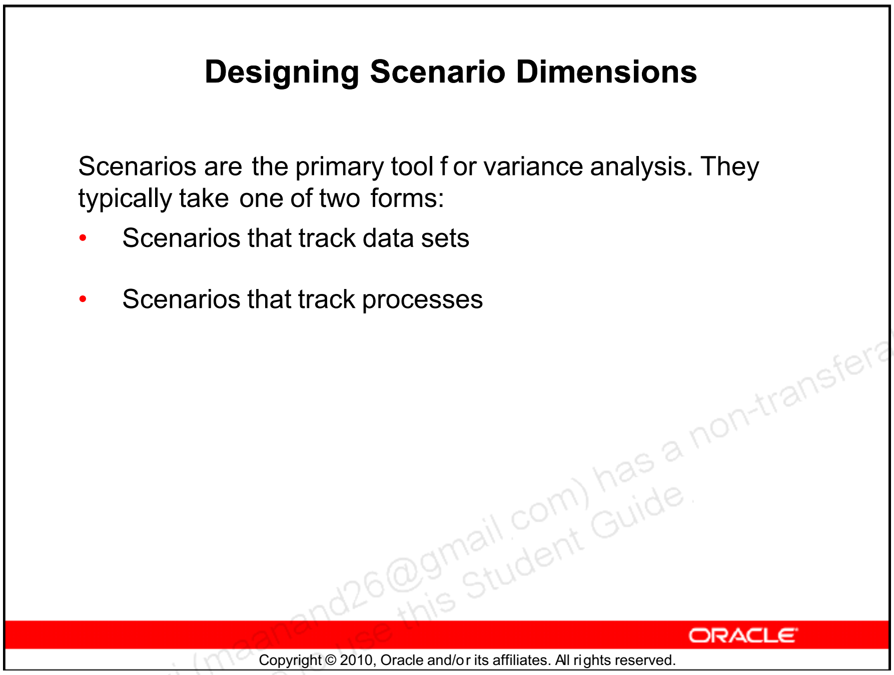
In the single-dimension time design, time periods are arranged in a continuous, single time dimension, with years as a generation and year-specific time periods under each year. You can implement this model either by creating unique time period names or by enabling duplicate names in the time dimension.

For example, the slide shows Jan FY05 as a child of Q1 FY05, which consolidates to FY05; and Jan FY06 as a child of Q1 FY06, which consolidates to FY06. FY05 and FY06 are siblings, children of the dimension Time.

While the single-dimension time design provides definite calculation advantages, because of the number of members required—in a monthly time dimension, every new year adds 17 more members—it can be less efficient for data storage than a multidimensional time design.

As a rule, you implement a single-dimension time design in a block storage outline only if your analysis or calculations require a continuum for the time frame, as in the following examples:

- You need to forecast prices or costs for a rolling forecast, where the forecast time horizon crosses over the end of the fiscal year.
- You need to create rolling lead or lag calculations that require a single time continuum (for example, an advertising drop in any given month that has a continuing lead generation impact over the six months after the drop).
- You need to calculate cumulative period-to-date values over a complete range from the first time period forward.
- You need both calendar year and fiscal year hierarchies in a single database.



Designing Scenario Dimensions

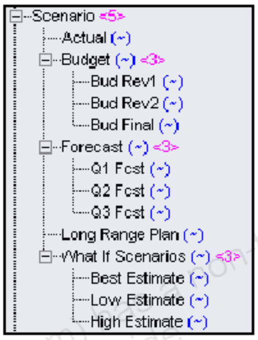
Although scenario dimensions usually have few members and a minimal hierarchy, their impact on design and calculation issues is substantial. You typically compare and compute the most important variances between data sets in the scenario dimension. With multiple data sets incorporated into one dimension, analysis of differences between sets becomes extremely fast and efficient.

The data sets vary based on the type of database that you are building, but they usually fall into one of two types: scenarios that track physical data sets and scenarios that track underlying processes.

Tracking Data Sets

Scenarios that track data sets:

- Are typical in financial applications
- Define the foundation of the planning and control process



ORACLE

Tracking Data Sets

The most common type of scenario dimension, used often in financial applications, tracks physical data sets and acts as the foundation for the variance planning and control process. Data sets in this type of dimension can include the following types of members:

- **Actual**—Monthly downloads from the general ledger of actual financial results
- **Budget**—Data for setting standards derived from the annual planning process
- **Forecast**—Estimates of financial performance updated monthly or quarterly
- **Plan**—Long-range estimates of financial performance
- **What if**—Estimates for target setting and complex modeling

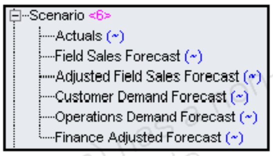
This type of scenario dimension often has sequential versions of data sets; the example on the slide shows three versions of the Budget data (Budget Rev1, Budget Rev2, and Budget Rev3). In production, each new version begins with a copy of the previous version, which you create using the DATACOPY calculation script command.

Tracking Processes

Scenarios that track processes:

- Are typical in forecasting applications
- Define a single data set
- Track adjustments by distinct functional groups

Tracking month-end close



Tracking sales forecasting

Tracking Processes

A slightly less common design, used primarily in forecasting applications, tracks the modification of one data set during the steps of internal processes. This scenario dimension design facilitates tracking of the internal build of a data set as various functional areas of the company contribute data.

By using a DATACOPY calculation script command, you can begin each successive scenario with the contents of the previous scenario, incorporating the input of a new functional group while preserving not only the srcinal data but also every adjustment to the srcinal data.

The names of the scenarios in the first example on the slide represent the transformation of data during a month-end close process:

1. Actuals GL Dump
2. Actuals Post Adjustments

3. Actuals Post Allocations
4. Actuals Mgmt Reporting
5. Actuals Shareholder Reporting

The second example on the slide represents the transformation of data during a sales forecasting process:

1. Actuals
2. Field Sales Forecast
3. Adjusted Field Sales Forecast
4. Customer Demand Forecast
5. Operations Demand Forecast
6. Finance Adjusted Forecast

Outline Calculations

The database outline fac ilitates calculations two ways:

- Hierarchy structures
- Member formulas

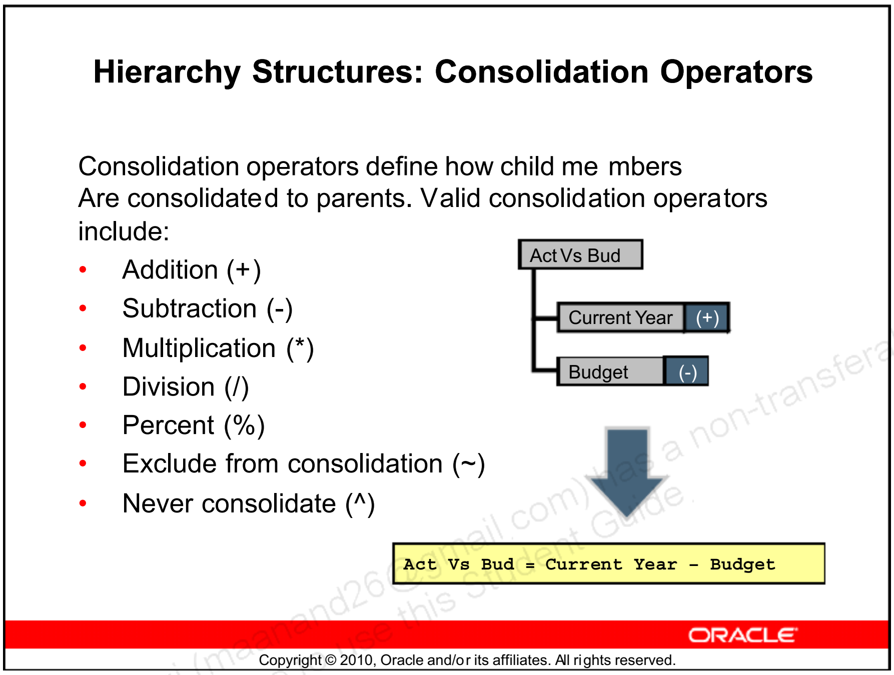
Outline Calculations

Because the primary purpose of a scenario dimension is to store distinct data sets for variance analysis, setting up variance calculations is an integral step in creating a scenario dimension. However, before you create variances, to facilitate calculations, you must be familiar with the calculation tools available in block storage outlines.

You can build calculations into your outline in two ways:

- **Hierarchy structures** are calculations determined by parent-child relationships and member consolidation operators.
- **Member formulas** are script calculations assigned to specific members using the member formula editor.

NOTE: Although both aggregate storage and block storage databases use hierarchy structures for calculation purposes, aggregate storage member formulas are written in MDX, a standards-based multidimensional query language.



Hierarchy Structures: Consolidation Operators

Database hierarchies, in addition to defining structural relationships between members, enable you to define calculations between members by setting consolidation operators. By default, when you add a member to the outline, the member consolidation operator is set to addition (+). You create custom calculations by changing the default consolidation operator to one of the following:

- Subtraction (-)
- Multiplication (*)
- Division (/)
- Percent (%)
- Exclude from consolidation (~)
- Never consolidate (^)

The consolidation operator defines how the member rolls up to its parent. The example on the slide shows the parent Act Vs Bud with children Current Year (+) and Budget (-). When the scenario hierarchy is calculated, the effective calculation for Act Vs Bud is Current Year - Budget.

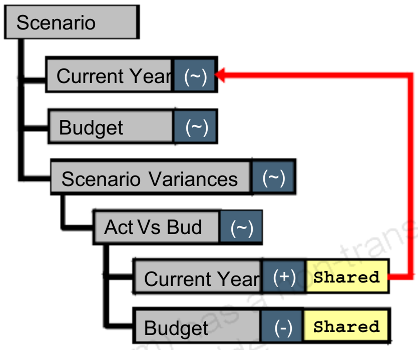
The exclude from consolidation (~) operator identifies the member as exempt from consolidation to the parent. For example, the children of Scenario are tagged as exclude from consolidation (~) so that they do not roll up into Scenario.

The percent (%) operator is the mathematical equivalent of using the divide operator and then multiplying by 100, returning a whole number instead of a decimal result.

The never consolidate (^) operator (available in Release 9.3 and later) prevents members from being consolidated across any dimension. For a detailed description about when to use this operator, see "Design Considerations for Rates" in Lesson 12, "Creating Calculation Scripts."

Hierarchy Structures: Shared Members

- Do not store data
- Create a pointer to a stored member
- Are always level 0 members
- Are positioned after (below) the stored member in the outline



Hierarchy Structures: Shared Members

The shared member storage property provides a way to reuse data in hierarchy calculations that Essbase has already indexed or calculated. Instead of storing data in multiple places, shared members create a pointer to a stored member. When Essbase encounters a shared member during calculations or retrievals, it references the data in the associated stored member.

Using shared members enables you to use members repeatedly throughout a dimension. Essbase stores the data value only once, but the value is displayed in multiple locations. Storing the data value only once saves space and increases processing efficiency.

While the shared member assumes most attributes of the stored member, it has its own consolidation operator and alias, both of which are independent of the stored member.

This independence enables you to build complex models with calculation dependencies between members.

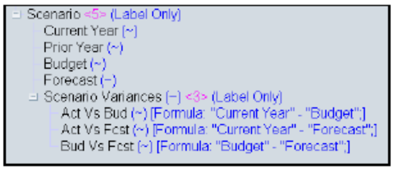
For the example on the slide, Current Year is a child of Scenario, with a consolidation operator of (~). Current Year is also a child of Act Vs Bud, with a consolidation operator of (+). The second occurrence of Current Year is a shared member.

When creating shared members, keep the following in mind:

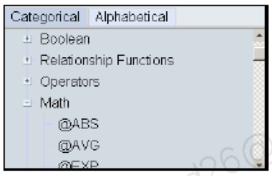
- Shared members and the associated stored member must be in the same dimension.
- In the outline, shared members must be positioned below their associated stored members.
- Stored members can be at any level in the hierarchy.
- Shared members must be level 0, even if their stored member is not.
- Shared members do not increase database size.
- You can have an unlimited number of shared members with the same name.
- You cannot assign UDAs or formulas to shared members.
- You cannot associate attributes with shared members.

Member Formulas

- **Member formulas** define calculations in th e outline:



- Calculation script **functions** return sets of members or data values:



Member Formulas

Whenever possible, use consolidation operators in the outline to define natural formulas. This method is more intuitive to the user, who can drill down to see how a member is derived. If you do not wish to provide hierarchy structures for users to drill down, you can construct calculations as member formulas, which are stored in the properties of specific members. The first example on the slide shows the member formula `"Current Year" - "Forecast";` for the scenario variance Act Vs Fcst, instead of a shared-member hierarchy structure. As a level 0 member, Act Vs Fcst provides no drill down for users.

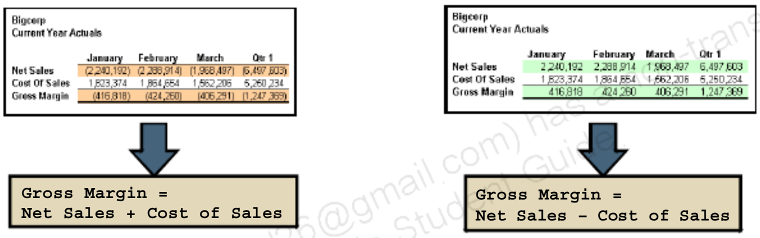
Some member calculations, such as averages, cannot be accomplished with hierarchy structures alone. For such calculations, member formulas are required. In block storage databases, Essbase provides a comprehensive set of calculation script functions that return sets of members or data values, which you can use to construct member formula calculations.

The second example on the slide shows a list of function categories. Calculation functions include Boolean functions to provide conditional tests and mathematical, statistical, and financial functions to perform specialized calculations.

Designing Accounts Dimensions

Design choices are based on input data and s ample reports:

- Profit and loss accounts, balance sheet accounts
- Units, rates, prices, and other assumptions
- Analysis metrics



Designing Accounts Dimensions

The accounts dimension (often called the *measures* dimension) is generally the most complex dimension in the outline. Your business model resides in the accounts dimension, and the most complicated and potentially time-consuming calculations occur there.

You can set up your accounts dimension to contain any of the following typical structures:

- Natural class accounts that define the profit and loss structure or subsets of the structure, as, for example, Bigcorp Sales, which includes accounts that comprise the Gross Margin total
- Balance and cash flow accounts and associated metrics, such as inventory and average inventory
- Unit, rate, and dollar calculations, especially where such calculations involve activity driver relationships between members in the dimension
- Metrics and analysis calculations of all types, including members such as Profit%, Margin%, Sales Per Employee, Cost Per Transaction, and Commission%

The calculation structures that you create for your accounts dimensio depend largely on the input data and the final reporting requirements that you defined during the design phase of your implementation. A common choice in building financial applications is whether to retain the srcinal signs of natural account balances or to change them.

In the first example on the slide, Net Sales and Goss Margin retain their natural negative (credit) balance, while Cost of Sales is positive (debit). In this case, Cost of Sales inherently offsets Net Sales, so the formula that you create for Gross Margin in your outline should be:

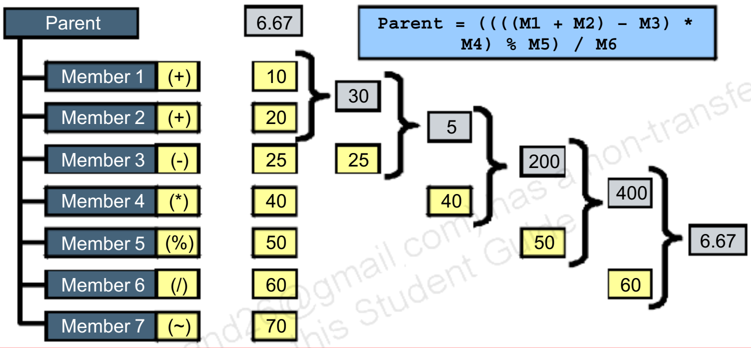
```
Gross Margin = Net Sales + Cost of Sales
```

However, in the second example, although all three accounts have a positive sign, the total for Gross Margin implies that Cost of Sales data still offsets Net Sales. Because the data in Cost of Sales and Net Sales is positive, you create an explicit offset in the formula for Gross Margin:

```
Gross Margin = Net Sales - Cost of Sales
```

Consolidation Order

- Consolidation order is defined by the order of members in the outline.
- Essbase calculates data in top-down order.



Consolidation Order

Before creating your accounts dimension, you must understand how Essbase calculates members with different consolidation operators. When you use only addition and subtraction operators, the order of members in the outline is irrelevant. However, when you use any other operator, you need to consider member order and its impact on consolidation.

When siblings have different operators, Essbase calculates data in top-down order. The example on the slide illustrates a top-down calculation. A parent in the outline has children in the following order:

1. Member 1 (+)
2. Member 2 (+)
3. Member 3 (-)
4. Member 4 (*)

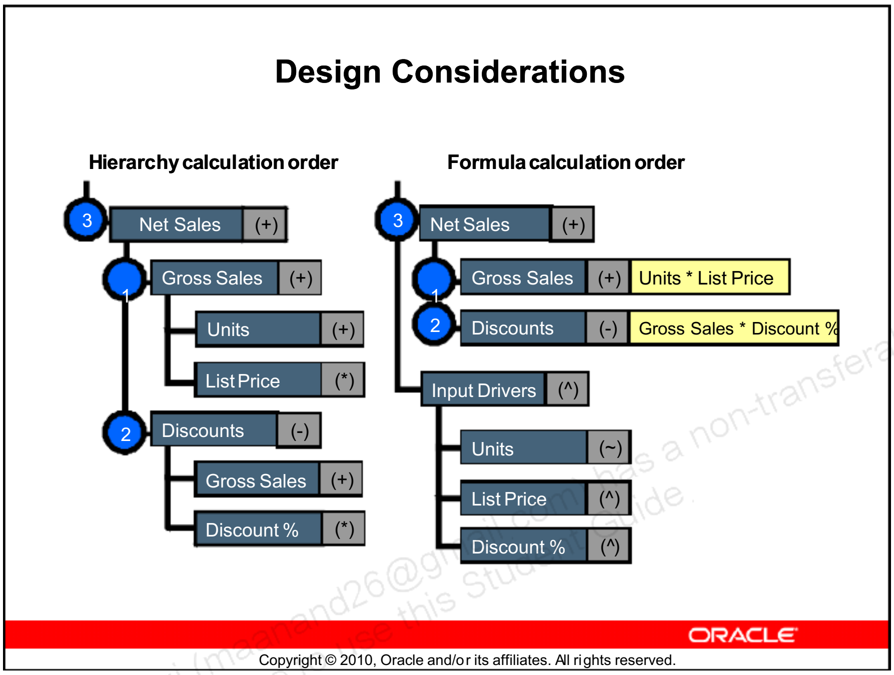
5. Member 5 (%)
6. Member 6 (/)
7. Member 7 (~)

Written left to right, the formula for the parent is:

```
Parent = ((((Member 1 + Member 2) - Member 3) * Member 4) % Member 5) /
Member 6
```

As you can see from this example, the consolidation operator of a member identifies the mathematical operation that Essbase performs when applying the data of the member to the ongoing calculation.

NOTE: Using top-down calculation order, Essbase ignores the first child of a parent if it does not have either the (+) addition or (-) subtraction consolidation operator.



Design Considerations

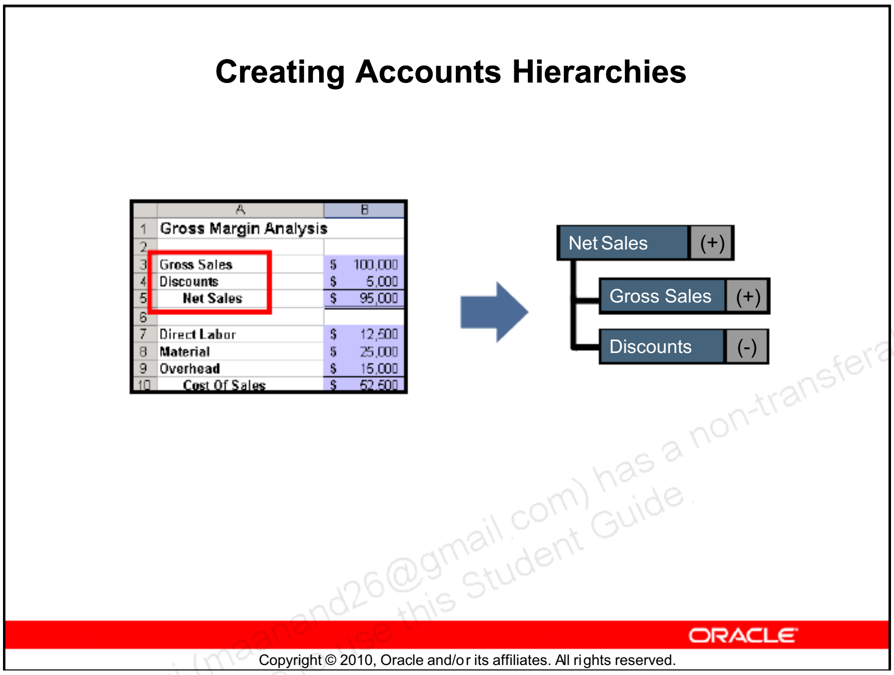
Dimension calculation order is also top-down; dimension branches are calculated one by one in top-down order. However, within each branch, Essbase calculates the lowest-level calculations first, followed by the related parent calculations. The calculation continues in this manner until all levels in the first branch are calculated. Essbase then moves to the second branch and repeats the process.

Although, in many cases, consolidation operators and member formulas produce the same calculated data, you should maximize the use of hierarchy-based calculations and consolidation operators when building your accounts dimension. The main reasons for this choice are performance and visibility.

- **Performance**—When an outline is consolidated, hierarchy-based calculations are much faster than member formulas. For the examples on the slide, the hierarchy calculation design—Units and List Price as children of Gross Sales with consolidation operators of (+) and (*), respectively—provides faster performance than the formula calculation design, where Gross Sales is assigned the member formula `"Units" * "List Price";`.
- **Visibility**— Hierarchy-based calculations provide visibility for users as the users drill down the accounts hierarchy to see how a value is derived. Formulas obscure visibility of the calculations and are less intuitive for users. What detail you want your users to see should be a primary design consideration.

The following is the best-practice design philosophy for building an accounts dimension:

1. Whenever possible, use hierarchy relationships and consolidation operators for outline calculations.
2. When you cannot achieve your desired calculation with consolidation operators, use an outline member formula or create a calculation script. For example, you may need to use a member formula for an account that calculates commissions based on several distinct conditions.
3. When you want to eliminate confusion in reports by simplifying hierarchy paths, do not apply guidelines 1 and 2. Even though the calculation for Gross Margin % involves Gross Margin and Net Sales, your users may not need to see the detail beneath the Gross Margin % value.



Creating Accounts Hierarchies

A typical challenge when creating your accounts dimension is converting a report in spreadsheet format into a meaningful outline structure. In this situation, keep the following points in mind:

- A total or subtotal in the spreadsheet becomes a parent member in your outline; the children are the elements that produce the total or subtotal.
- A parent member is displayed *above* its children in the outline editor, but *below* its children in the spreadsheet.
- The top member of a hierarchy branch in the outline is often what is called the *bottom line* in a spreadsheet report. For the example on the slide, Gross Margin is the final total of all data on the report; in the outline, Gross Margin is the top member of a hierarchy branch that includes all other members represented in the report.

Testing Outline Calculations

- Disable intelligent calculation
- Set default calculation script
- Follow standard test procedure

Calculation Testing Procedure	1. Clear all data from the database. 2. Load calculation test data. 3. Execute your calculation. 4. Audit your calculation results.

Testing Outline Calculations

After creating a complex accounts hierarchy, it is a best practice to input simple test data and test your calculation results.

Considerations for testing:

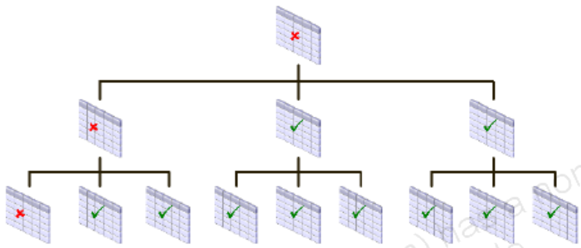
- **Intelligent calculation**—By default, intelligent calculation is enabled and Essbase calculates only data that changed since the last calculation. If your outline hierarchy has errors, you may need to perform multiple iterations of your calculation test. To ensure correct test results for each iteration, you must disable intelligent calculation.
- **Default calculation script**—The default calculation script for every database is a CALC ALL command, which executes every calculation in the outline. It is common practice during database and calculation development to disable intelligent calculation by modifying the default calculation script.

- **Testing procedure**—Using the following testing procedure ensures results that are based on your input data and the subsequent calculation, not on other factors:

1. Clear all data from the database.
2. Load calculation test data.
3. Execute your calculation.
4. Audit your calculation results.

Intelligent Calculation

Only data blocks marked as changed are recalculated.



Intelligent Calculation

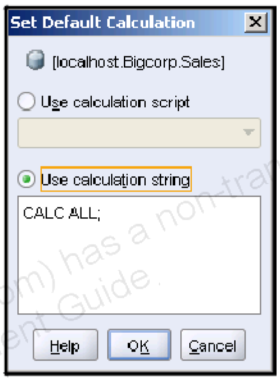
When you perform a full block storage database calculation, Essbase marks what was calculated. If you then load a subset of data, you can calculate only the changed data and related ancestors. This selective calculation process is called *intelligent calculation*.

By default, intelligent calculation is enabled at the server level. During database and calculation development, you must disable intelligent calculation. You can change the default setting in the `essbase.cfg` file or on a script-by-script basis with the SET UPDATECALC OFF command. For information about the `essbase.cfg` file, see "Essbase.CFG Configuration Settings" in the *Oracle Essbase Technical Reference*.

NOTE: For a detailed explanation of intelligent calculation, see "Intelligent Calculation" in Lesson 15, "Developing and Testing Complex Calculation Scripts."

Setting the Default Database Calculation

- Default calculation string
- Optional default calculation script



ORACLE

Setting the Default Database Calculation

The default database calculation is a simple command (CALC ALL) that executes your outline calculations. You can modify the default calculation or assign a prebuilt calculation script to be the default database calculation.

TIP: During the database development process, you may need to test outline calculations before you have developed calculation scripts. You can use the default database calculation for your testing purposes; however, you should disable intelligent calculation by modifying the default calculation script.

To set the default database calculation:

1. In the navigation frame, select the database node.
2. From the Actions menu, select **Set**, and then select **Default calculation for "*DBName*"**.
3. Perform one of the following actions:
   - Select **Use calculation script** to make an existing calculation script the default script.
   - In the script text box, enter modifications to the default script.
4. Click **OK**.

Summary

In this lesson, you should have learned to:

- Describe data descriptor dimensions
- Design time, scenario, and accounts dimensions
- Explain outline calculations
- Test outline calculations

LESSON 4

Optimizing Data Descriptor Dimensions

Objectives

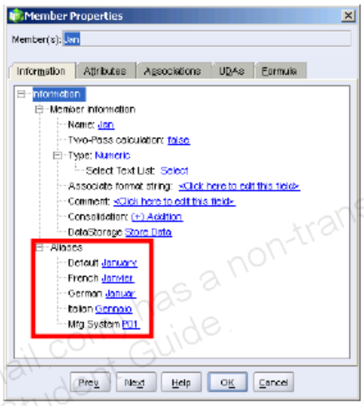
At the end of this lesson, you should be able to:

- Create member aliases
- Describe the available dimension types
- Create period-to-date totals
- Discuss Dynamic Calc members
- Enhance accounts dimensions
- Optimize data storage

Creating Member Aliases

Aliases are:
- Used as reporting names
- Used for name mapping
- Updated in Member Properties
- Stored in alias tables

Alias tables are:
- Created in Outline Properties
- Limited to 32 per database
- Updated separately
- Displayed one at a time



ORACLE

Creating Member Aliases

An alias is an alternative name for a member or shared member. For the example on the slide, the member Jan is identified by the short name Jan and the more descriptive default alias January. Aliases are commonly used for the following reasons:

- To improve the readability of an outline or a report by providing descriptive names
- To accommodate multilingual databases by representing members in other languages
- To facilitate name mapping during data load when data source names do not match Essbase member names (Data can be loaded to member names or aliases.)

Aliases are stored in alias tables. The use of tables enables you to set more than one alias for a member. For the example on the slide, the member Jan has descriptive aliases in English, French, German, and Italian, and also has an alias of P01 to map data coming from a manufacturing system to the Jan member. You can manually update all aliases of a member in the Member Properties dialog box.

- **Alias tables**—Aliases are stored in one or more tables as part of a database outline. An alias table maps a specific, named set of alias names to member names. When you create a database outline, Essbase creates an empty alias table named *Default*. You can create up to 31 additional alias tables for each outline, for a total of 32.

> **NOTE:** Prior to release 11.1.2, alias tables were limited to 10 for each database.

- **Alias table names**—When you view the outline or retrieve data, you use the alias table name to indicate the set of aliases that you want to see. Similarly, when you use rules files to update aliases, you use the alias table name to indicate the set of aliases that you want to update.

> **NOTE:** You can view or update only one alias table at a time.

To create alias tables:

1. In Outline Editor, select the **Properties** tab.

   The outline properties are displayed.

2. Right-click the "Alias tables" node, and select **Create alias table**.

   The Create Alias Table dialog box is displayed.

3. Enter the name of your alias table, and click **OK**.

   The new alias table name is displayed.

Dimension Types

- None (default)
- Accounts
- Time
- Country
- Currency
- Attribute

Time and accounts dimension types enabled

Dimension Types

Essbase provides dimension types to enhance your outline. When you tag a dimension as a specific type, the dimension 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 the dimension. Essbase calculates the two primary dimension types, time and accounts, before other dimensions in the database. By default, all dimensions are tagged as None. The following are the available dimension types:

- Accounts
- Time
- Country
- Currency
- Attribute

The example on the slide shows an outline with both the TIme and Accounts dimension types enabled; the dimension type is displayed next to the dimension name in the outline.

NOTE: Country and currency dimension types are used only in currency conversion applications. For a comprehensive discussion of currency conversion, see “Designing and Building Currency Conversion Applications” in the *Oracle Essbase Database Administrator’s Guide*.

Creating Period-to-Date Totals

- Calculated members
  – Shared member hierarchy
  – Member formula
- Dynamic Time Series

Bigcorp
Current Year Actuals

	January	February	March	Apr	YTD
Net Sales	2,240,192	2,288,914	1,968,497	2,158,263	8,655,866
Cost of Sales	1,823,374	1,864,654	1,562,206	1,751,802	7,002,036
Gross Margin	416,818	424,260	406,291	406,461	1,653,830

ORACLE

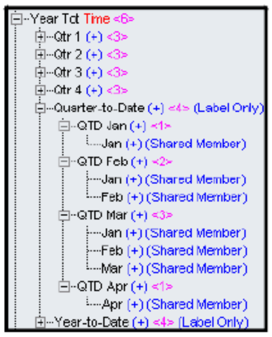
Creating Period-to-Date Totals

Period-to-date reporting requirements are a staple of many databases. There are two methods to accomplish period-to-date analysis in block storage databases: creating calculated members—either using hierarchy structures or member formulas—or enabling Dynamic Time Series members.

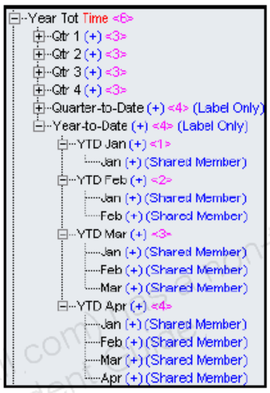
The example on the slide shows a period-to-date report, with data for January through April summed in the YTD column.

Creating Period-to-Date Calculated Members

QTD hierarchy



YTD hierarchy



ORACLE

Creating Period-to-Date Calculated Members

With the calculated-member method, you create a hierarchy in the time dimension for each level of period-to-date analysis that you want to calculate. In each hierarchy, you create calculated members for each distinct calculation range.

For example, in a monthly time dimension, assume that you require both quarter-to-date and year-to-date analysis. In this case, the period-to-date analysis requires two additional hierarchies in the time dimension, with a QTD or YTD total for each month of the year.

The following table describes the hierarchies shown in the example on the slide:

Period-to-date member	Shared member children
QTD Jan	Jan
QTD Feb	Jan, Feb
QTD Mar	Jan, Feb, Mar

Period-to-date member	Shared member children
QTD Apr	Apr
YTD Jan	Jan
YTD Feb	Jan, Feb
YTD Mar	Jan, Feb, Mar
YTD Apr	Jan, Feb, Mar, Apr

Period-to-date hierarchies provide the following benefits:

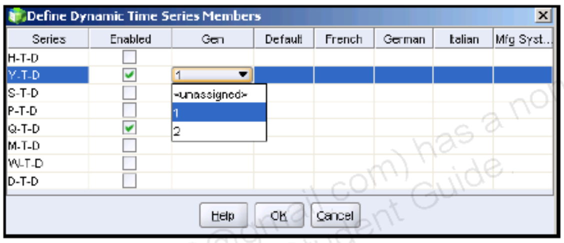
- Physical hierarchy members are easy for users to find.
- Data can be loaded to stored period-to-date members and then allocated to related months for top-down planning.
- Stored calculations give you more control over calculation order.

When using this model, keep in mind the following considerations:

- Additional hierarchies require additional outline setup and maintenance.
- A greater number of stored members requires increased storage space on disk, possibly affecting performance.
- A greater number of stored calculated members increases batch calculation times.

Implementing Dynamic Time Series

- Automatically calculates the sum of a range of level 0 time periods
- Starts with a preset relative generation and ends with th e selected member



ORACLE

Implementing Dynamic Time Series

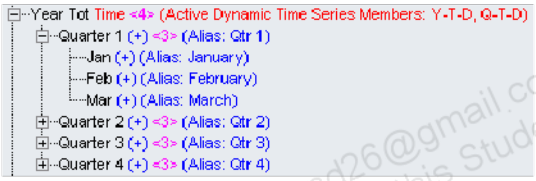
The second method, enabling built-in Dynamic Time Series members, requires that you use the Time dimension tag to identify your time dimension.

Consider the following design implications when using the Time dimension tag:

- You can use the Time dimension tag on only one dimension of an outline.
- All members in the dimension tagged as Time inherit the time property.
- You can add time-related members to dimensions that are not tagged as Time (as with the Years dimension in the generic time design).
- You can create an outline that does not have a dimension tagged as Time.

You do not create Dynamic Time Series members as members of a dimension in the database outline. Instead, you enable predefined Dynamic Time Series members and associate them with generation numbers that define the beginning of the calculation range. Then, when users query the database, they specify the last time period of the calculation range. Essbase starts the calculation range with the first level 0 descendant of the preset generation ancestor, and ends with the specified member.

For example, if you want to calculate quarter-to-date values, you enable the QTD member and associate it with the generation of the time dimension that contains quarterly totals. In the Bigcorp Sales database, the generation containing quarters is generation 2. An enabled Dynamic Time Series member is not shown as part of the regular time dimension hierarchy, but its name is indicated next to the dimension name. The following figure shows a standard monthly time dimension called Year Tot. Next to Year Tot is displayed the Time dimension tag and “Active Dynamic Time Series Members: Y-T-D, Q-T-D.”



When you want to see a quarter-to-date total in a report, you specify the label Q-T-D(*EndPeriod*); Essbase automatically calculates the sum of a range of level 0 time periods, starting with the first period in the *EndPeriod* quarter and ending with *EndPeriod*. The following figure shows a report with columns of data for Apr, May, and Q-T-D(May) where the data in Q-T-D(May) is a sum of Apr and May data.

	Apr	May	Q-T-D(May)
Net Sales	18,702.33	18,767.22	37,469.55
Cost Of Sales	12,916.47	12,611.06	25,527.53
Gross Margin	5,785.86	6,156.16	11,942.02

Because Dynamic Time Series members are not stored, they have no impact on database size or calculation times. However, they do have the following implications:

- You cannot load data to Dynamic Time Series members.
- Users cannot drill down on Dynamic Time Series members.
- Some calculations return incorrect results when calculated dynamically in Dynamic Time Series members.

To set up Dynamic Time Series members:

1. In the outline, select the time dimension.
2. From the Edit menu, select **Edit member properties**.

   The Member Properties dialog box is displayed.
3. From the “Dimension type” drop-down list, select **Time**, and then click the **OK** button.

   The Time dimension tag is displayed next to the name of the time dimension.
4. From the Outline menu, select **Dynamic time series**.

   The Define Dynamic Time Series Members dialog box is displayed.
5. For each Dynamic Time Series member that you want to use, complete the following tasks:

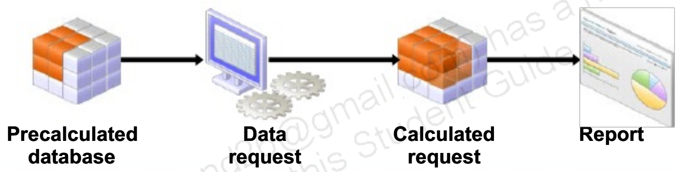
   a. Select **Enabled**.

   b. From the Gen drop-down list, select a generation.

   c. **Optional:** In the columns provided for the alias tables, enter aliases for the Dynamic Time Series member.
6. Click **OK**.

Dynamic Calc Members

- Advantages:
  - Reduce batch database calculation time
  - Reduce disk storage usage
  - Reduce database restructure time
  - Reduce time required to perform a backup
- Considerations:
  - May increase retrieval time
  - May change calculation order



Dynamic Calc Members

Dynamic calculation storage options provide flexibility about how and under what circumstances calculations are performed and data is stored. The immediate benefit of dynamic calculation is reduction of batch calculation time and hard drive storage requirements.

Advantages of Dynamic Calc Members

Dynamic calculation options allow outline members to be calculated when queried by users rather than during the batch calculation process. Calculating some values dynamically achieves the following advantages:

- It reduces the batch calculation time of the database because Essbase has fewer members to calculate.
- It reduces disk usage because Essbase stores fewer calculated data values.

- It reduces database restructure time. Adding or deleting a Dynamic Calc member does not change the database structure, so Essbase does not need to restructure the database for such additions and deletions.
- It reduces the time that is required to back up the database. Because database size is reduced, Essbase requires less time to perform a backup.

Block storage databases provide two dynamic calculation options: Dynamic Calc and Dynamic Calc and Store.

Considerations for Dynamic Calc Members

You can tag any calculated member in your outline as Dynamic Calc. Essbase does not calculate the value for a Dynamic Calc member until a user requests it, and does not store the value for the Dynamic Calc member. Dynamically calculating selected database values can significantly improve the performance of an overall database calculation.

When implementing the Dynamic Calc storage property, keep the following considerations in mind:

- Data values that calculate dynamically can take longer to retrieve. To avoid causing a significant impact on user retrieval times, you should usually limit your use of the Dynamic Calc option to dense dimensions.
- Dynamic calculation order differs from batch calculation order. Thus, dynamic calculation may produce incorrect results. When you change a member from stored to Dynamic Calc, be sure to test calculation results.

Dynamic Calc and Store Option

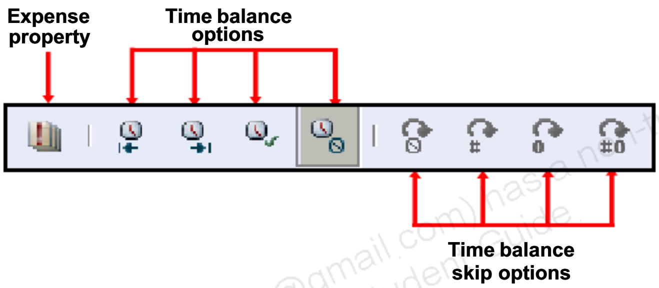
If you tag a member as Dynamic Calc and Store, Essbase defers calculation of the member until the first time that a user queries the member. After calculation, Essbase stores the data for the member. Subsequent retrievals of the data value do not require calculation, unless Essbase detects that the value needs recalculating.

The Dynamic Calc and Store option should be used only in certain circumstances, such as for sparse dimensions with complex calculations. In most cases, consider using Dynamic Calc before Dynamic Calc and Store.

TIP: For more detailed guidelines on deciding which members to calculate dynamically, see the *Oracle Essbase Database Administrator's Guide*.

Enhancing Accounts Dimensions

- Time balance reporting
- Expense reporting



Enhancing Accounts Dimensions

As with the Time dimension tag, the Accounts dimension tag enables additional features:

- **Time balance reporting**—To flag measures that do not aggregate over time (for example, balance sheet accounts)
- **Expense reporting**—To identify favorable and unfavorable variances

You can modify expense and time balance properties with toolbar buttons or in the Member Properties dialog box.

Implementing Time Balance Reporting

Time balance options:

- TB First
- TB Last
- TB Average

Skip options:

- Missing
- Zeroes
- Missing and zeroes

Account	TBTag	Jan	Feb	Mar	Qtr1
Opening Inventory	TB First	**20**	31	41	**20**
Additions		11	10	15	36
EndingInventory	TBLast	31	41	**56**	**56**

ORACLE

Implementing Time Balance Reporting

You use time balance to control the consolidation of balance sheet items across time periods.

Time Balance Options

For accounts, you can set one of the following values for the time balance property:

- **None**—When a member in the accounts dimension represents a value that aggregates over time, the corresponding parent in the time dimension is calculated based on the consolidation operators and formulas of its children. For the example on the slide, the time balance property for the Additions member is set to the default value, None. Thus, Essbase calculates the Qtr 1 member in the time dimension as the sum of its children (Jan, Feb, and Mar).

- **TB First**—When a member in the accounts dimension represents the value at the beginning of a time period, and you want the value carried to the parent in the time dimension, set the time balance property of the account to TB First. For the example on the slide, the Opening Inventory member represents the inventory value at the beginning of the time period. If the time period is Qtr 1, Opening Inventory represents the inventory at the beginning of Jan; that is, Opening Inventory for Qtr 1 and Opening Inventory for Jan are the same. Similarly, Year Tot Opening Inventory equals Qtr 1 Opening Inventory.
- **TB Last**—When a member in the accounts dimension represents the value at the end of a time period, and you want the value carried to the parent in the time dimension, set the time balance property of the member to TB Last. For the example on the slide, Ending Inventory represents the inventory at the end of the time period. If the time period is Qtr 1, Ending Inventory represents the inventory at the end of Mar; that is, Ending Inventory for Qtr 1 and Ending Inventory for Mar are the same. Year Tot Ending Inventory equals Qtr 4 Ending Inventory.
- **TB Average**—When you want parent values in the time dimension to represent the average value of its children for a member in the accounts dimension, set the time balance property of the member to TB Average.

Skip Options

If you set the time balance as TB First, TB Last, or TB Average, you must set the Skip option to tell Essbase what to do when it encounters missing values or zero values. The following table describes how each setting determines what Essbase does when it encounters a missing or zero value.

Setting	Description
None	Zeros and missing values are considered when parent values are calculated. This is the default setting.
Missing	`#MISSING` values are excluded when parent values are calculated.
Zeros	Zero values are excluded when parent values are calculated.
Missing and Zeros	`#MISSING` values and zero values are excluded when parent values are calculated.

Calculating Variances

1. Tag all expense members with the **Expense Reporting** tag.
2. Use @VAR or @VARPER to calculate variances.

	Actual	Budget	Variance	Variance %
Gross Sales	100	120	-20	-16.67
Discounts	100	120	20	16.67

```
Variance = @VAR(Actual, Budget);
```

```
Variance % = @VARPER(Actual, Budget);
```


Calculating Variances

A typical analytic requirement is the ability to perform variance reporting on actual versus budget data. When you budget expenses for a time period, you want the actual expenses to be lower than the budget expenses. When actual expenses are greater than budget expenses, the variance is unfavorable, or negative. Conversely, when you budget nonexpense items, such as revenues, you want actual revenues to be higher than budget revenues. When actual revenues are less than budget revenues, the variance is unfavorable.

For the example on the slide, Gross Sales (a revenue) and Discounts (an expense) have the same data values for the actual and budget scenarios. Usually variances are calculated with a simple difference (Actual - Budget or Budget - Actual). But, in an outline such as Bigcorp Sales, where expenses and revenues have the same sign and offset each other through outline consolidation operators, a simple difference calculation between actual and budget data does not correctly reflect favorable versus unfavorable variances. As you can see in the example, the variance for Gross Sales is negative and the variance for Discounts is positive.

Dealing with this type of variance calculation issue requires two steps:

1. All outline members that represent an expense to the company (or any value that is determined unfavorable when over budget) must be assigned the Expense Reporting tag. For example, Inventory members, Cost of Sales members, and the Discounts member each receive the Expense Reporting tag for variance reporting.
2. Variance calculations must be written with a special function that checks for the presence of an Expense Reporting tag and calculates accordingly. Block storage databases provide two functions for this purpose: @VAR and @VARPER.

For the example on the slide, the formulas for the two scenario variances in the relevant outline are as follows:

```
Variance = @VAR(Actual, Budget);

Variance% = @VARPER(Actual, Budget);
```

The functions trigger Essbase to check for the Expense Reporting tag. If the account has the tag, Essbase transposes the sign of the formula result.

Optimizing Data Storage

Shared data:
- Label-only m embers
- Implied shares

Optimizing Data Storage

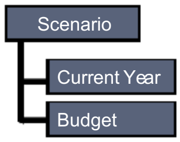
Members added to your outline have a default storage setting of stored. Thus, Essbase reserves storage space on disk for input or calculated data related to the members. However, members to which no data will ever be aggregated or loaded impede the flow of data and restrict visibility for users. A vital design goal is to ensure data visibility at all levels of the hierarchy.

You can provide visibility and maximize the efficiency of your database by using the Label Only storage option. Additionally, in some circumstances Essbase optimizes data storage automatically by creating implied shares between members.

You can modify data storage options in the Member Properties dialog box.

Optimizing with Label-Only Members

- Do not store data
- Group members to ease navigation and reporting
- Point to data of the first stored child



Member	Storage	Qtr1	Qtr2	Qtr3	Qtr4	YearTot
Scenario	LabelOnly	**120**	**125**	**150**	**150**	**545**
CurrentYear	Stored	120	125	150	150	545
Budget	Stored	110	140	140	170	560

ORACLE

Optimizing with Label-Only Members

In some dimensions, you may want to create a parent simply to group members together or to ease navigation and reporting. Label-only members do not store data. However, when viewed in reports or used in calculations, a label-only member points to the data for its first stored child.

For example, consider a Scenario dimension with the members Current Year and Budget as children of Scenario, as shown in the example on the slide.

In this Scenario hierarchy, the member Scenario serves an organizational purpose and does not represent a data set. Data is not aggregated or loaded to the member Scenario, and cells relating to the member will never contain data. Using “label only” as the data storage property for this member reduces the disk space required for the dimension, improves database efficiency, and provides a view of data at every level in the Scenario hierarchy.

For the example on the slide, the member Scenario (label only) displays the data for its first child, Current Year.

When assigning the label-only storage property, keep the following restrictions in mind:

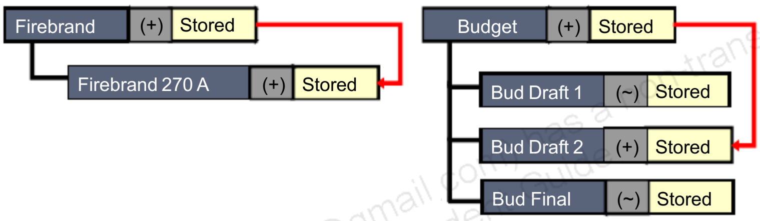
- Level 0 members cannot be label only.
- Label-only members cannot be calculated.

TIP: Writing data to label-only members automatically updates the data in the member's first child instead. For example, in Smart View, submitting data to the member Scenario writes data to the member Current Year.

Automatic Optimization: Implied Shares

An implied share is an automatic internal optimization created by Essbase under the following circumstances:

- A parent has only one child.
- A parent has only one child that consolidates to the parent.



Automatic Optimization: Implied Shares

The shared member and label-only properties define a shared data relationship explicitly, but some members are shared in certain circumstances even if you do not explicitly set them as shared. The relationship, called an *implied share*, is an internal optimization created by Essbase under the following circumstances:

- **A parent has only one child.** In this situation, it is implied that the parent and the child should contain the same data. Essbase ignores the consolidation property on the child and stores the data only once, creating an implied share relationship from the parent to the child. For the example on the slide, the parent Firebrand has only one child, Firebrand 270 A, so Firebrand shares the value of Firebrand 270 A.

- **A parent has only one child that consolidates to the parent.** If the parent has three children, but two of them are marked to be ignored during consolidation (~), the parent and the one child that consolidates contain the same data. Essbase ignores the consolidation property on the child and stores the data only once, creating an implied shared relationship from the parent to the child. For the example on the slide, Budget has only one child, Bud Draft 2, that consolidates to it. The other children are marked to be ignored during consolidation (~), so Budget implicitly shares the value of Bud Draft 2 even though it is not the first child of Budget.

Reducing Report Maintenance

Take advantage of implied shares to facilitate versioning and reduce report maintenance: In the example, Budget shares Bud Draft 2 data. When the budget is final, the administrator updates the consolidation operators, changing Bud Draft 2 to ignore during consolidation (~) and Bud Final to addition (+). Now Budget and any report that uses the member name Budget reflects Bud Final data, instead of Bud Draft 2 data.

If you do not want a member to be shared implicitly, tag the parent as Never Share. When a parent is tagged Never Share, Essbase duplicates the child data in the parent member.

Summary

In this lesson, you should have learned to:

- Create member aliases
- Describe the available dimension types
- Create period-to-date totals
- Discuss Dynamic Calc members
- Enhance accounts dimensions
- Optimize data storage

LESSON 5

Developing Dimension Designs

Objectives

At the end of this lesson, you should be able to:

- Describe business view dimensions
- Identify attributes in database design
- Combine business views
- Design dimensions with label outlines

Business View Dimensions Overview

Business view dimensions:
- Define a specific cut of the data
- Are developed and maintained through automatic processes
- Incorporate alternate roll ups using shared members
- Are not calculation-intensive
- Are sparsely populated with data

Business View Dimensions Overview

Although most Essbase databases have dimensions in common, the power of Essbase comes from the custom-built dimensions that are specific to the analytic needs of a particular business. These business view dimensions differ from implementation to implementation. They provide users with a specific cut of the data, facilitating a multidimensional richness of analysis that extends beyond the information incorporated in the data descriptor dimensions.

From a design perspective, business view dimensions share the following characteristics:

- **They define a specific cut of data.** Rather than defining the essentials of the data, business view dimensions provide information about a specific aspect of your business. A human resources database might include a dimension with details on employees and office locations, whereas a sales database might include a customer dimension to track sales by customer. Other common business views are legal entities, geographical locations, and products. The choice of which business view dimensions to incorporate in your design and how to structure the dimensions is driven by your company's industry and business practices.
- **They are usually developed and maintained through automatic processes.** Business view dimensions often have hundreds or even thousands of members and many levels in their hierarchies. Therefore, it is common to create and maintain them by loading member names and properties with a dimension build rule, rather than by entering member names and properties directly into the outline.
- **They incorporate alternate hierarchies by using shared members.** Different functional groups in the company usually require different levels of detail for the same information. Alternate hierarchies, created using shared members, enable all users to see the information that they need for analysis.
- **They are not calculation-intensive.** Calculations in business view dimensions are usually simple aggregated rollups of branches in the dimension hierarchy. There are very few (or no) complex models, member formulas, or variances.
- **They are usually sparsely populated with data.** Of all dimensions in a block storage database, business view dimensions are the least likely to have one or more data points (one or more member intersections) occupied. As a result, business view dimensions are usually configured as sparse dimensions.

Attributes in Database Design

- User-defined attributes
  - Defined in member properties
  - Used for any dimension
- Attribute dimensions
  - Defined in a separate dimension
  - Assigned in member properties
  - Associated only with sparse dimensions

Attributes in Database Design

When you classify information into dimensions, you may find that some metadata can be considered a property, or *attribute*, of other metadata. Essbase provides two types of attributes that you can use in your database design: UDAs and attribute dimensions.

UDAs

UDAs are flexible, multipurpose tags that you can assign to any member in the database, regardless of dimension type or hierarchy level. You can create UDAs in the Member Properties dialog box or load them to members using rules files.

Consider a product dimension where each product comes in a number of different colors. The following table lists products and their assigned UDAs:

Product	ColorUDAs
ProductA	Black, Green
ProductB	Black, Red, Green
ProductC	Black, Orange, Red, Green

Notice that each product can be assigned any number of color UDAs.

Attribute Dimensions

Attribute members are characteristics of another dimension that are defined in a stand-alone attribute dimension. When an attribute dimension is associated with a standard dimension, the standard dimension is the *base dimension* for that attribute dimension. (A *standard dimension* is any dimension that is not an attribute dimension.) Attribute members are assigned to base dimension members in the Member Properties dialog or using a rules file.

For example, to track product colors, you can create an attribute dimension called Color with the members Black, Green, Red, and Orange. Product can be the base dimension for the Color attribute dimension.

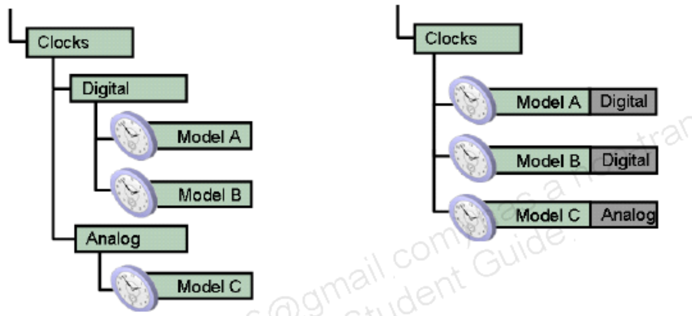
Attribute dimensions provide powerful calculation and analytic advantages that UDAs do not, but also have the following restrictions:

- In block storage databases, attribute dimensions can be associated only with sparse standard dimensions.
- Attribute dimensions can be associated only with one base dimension.
- Attribute members have a one-to-one relationship with members of the base dimension. If attribute members are used instead of UDAs in the preceding example, only one color attribute can be assigned to each product. An attribute solution for this example requires unique product names for each product-color combination.
- In block storage databases, extensive use of attribute dimensions can negatively affect performance.

NOTE: For a comprehensive discussion about attribute dimensions and their performance implications, see Lesson 17, "Creating Attribute Dimensions."

Combining Business Views

- Consider combining several business views in one hierarchy
- Consider using a ttributes



Combined business views **Standard hierarchy with attributes**

Combining Business Views

You can represent each of your business views as a standard dimension in the database. However, while designing the outline, be aware that the number of standard dimensions in a block storage database affects database performance. The more standard dimensions you create, the larger your database and the greater the calculation time.

TIP: Block storage databases should have fewer than ten dimensions for best performance.

Combining Business Views: Guidelines

You can fine-tune your database design and more effectively leverage the multidimensional technology by adhering to the following guidelines:

- Combine dimensions to avoid interdimensional irrelevance. Irrelevance occurs when many members of a dimension are irrelevant across other dimensions. 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.
- Consider making dimensions that describe or classify other dimensions attribute dimensions. Attribute dimensions are not stored, and therefore do not affect database size or batch calculation performance.

Redesigning Dimensions: Example

Consider an example where you need to track data about clocks. You have a list of clock models that represent the individual products, but you would also like information about sales of digital clocks versus analog clocks. Each clock model is either digital or analog, and cannot be both.

One way to accomplish your analytic goals is to create two standard dimensions, a dimension called Products with your clock models as children (Model A, Model B, and Model C) and a dimension called Type with the children Digital and Analog. However, you also want to limit the number of standard dimensions in your database. The following methods are only two of many ways to redesign the dimensions.

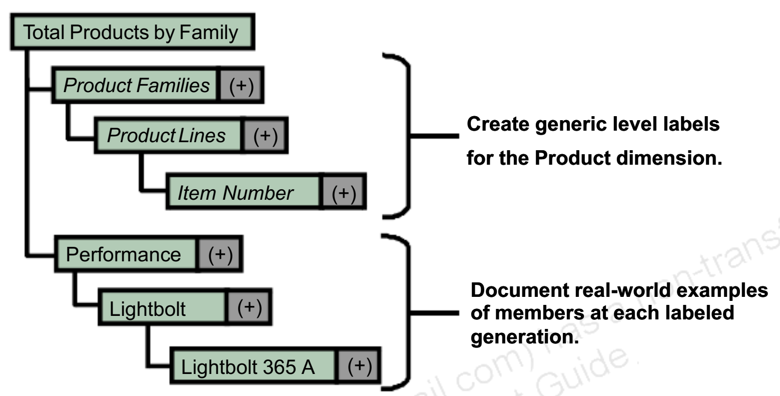
Redesigning by Combining Dimensions

You combine dimensions by creating additional hierarchy levels in a single dimension: Clock models, instead of being in a separate flat dimension, can consolidate to clock types, as shown in the first example on the slide. This variation reduces database size by eliminating interdimensional irrelevance and also provides a logical hierarchy for users.

Redesigning with Attribute Dimensions

Because the clock type can be considered a way to classify the clock models, you can convert the Type dimensions into an attribute of the Products dimension, as shown in the second example on the slide. Thus, each clock model can be assigned a type attribute, and extra consolidation levels are not required.

Developing Label Outlines



Developing Label Outlines

Business view dimensions can have thousands of members and many potential levels in their hierarchies. Planning in advance what levels are required is a necessary part of the implementation process. One planning method is to develop a documented plan, called a *label outline*, for planning purposes before starting to load members into the outline. Label outlines help you define the type of data represented at each level in the dimension and help you ensure symmetry within hierarchy branches.

The following steps summarize the process for creating label outlines:

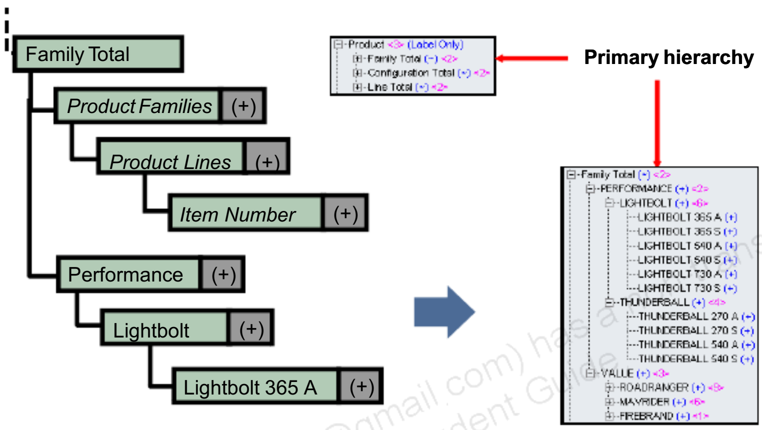
1. Open an empty outline on a client machine.
2. Create an outline member name for the dimension that you are analyzing.
3. Starting from the top of the dimension, add a generic name for each level (for the example on the slide, Product Families is the generic name for the consolidation level under Total Products by Family).

4. Starting from the top at a generation parallel to the label hierarchy, add a real-world business example of the named level (for the example on the slide, Performance is the real-world example of a product family).
5. Continue adding layers of labels with parallel examples until you reach the bottom of the hierarchy.
6. Use a finalized version of the label outline as a specification sheet or design document.

Constructing a label outline during a design meeting provides a useful planning tool and also identifies questions, issues, and potential areas of misunderstanding. This planning method provides the following advantages:

- **It provides a single visual planning model.** Having an abbreviated picture of a dimension hierarchy in the outline editor provides a concrete example that participants can examine and discuss.
- **It exposes nomenclature differences.** Working through the structure top-to-bottom identifies nomenclature differences between functional groups. For example, Finance calls a particular classification level *Product Families*, whereas Manufacturing calls the same classification level *Product Groups*. The label outline helps the functional groups decide which nomenclature to use in the database outline and ensures that all participants understand the classification of data in Essbase.
- **It discloses conflicting hierarchy perceptions.** Working through the structure identifies differences among functional groups in regard to hierarchy needs. For example, a five-level product hierarchy meets the Finance department's needs because product costs are recorded at the lowest level (item number), whereas a shallower, less complex three-level hierarchy meets the Sales department's need for forecasting.
- **It highlights sizing, performance, and partitioning considerations.** In addition to enabling the estimation of how many members are likely to be required at each level, working through the structure stimulates discussion of database size and performance. Size and performance issues are important to identify early in the design process, as they may suggest that a single database should be partitioned into multiple databases.

Designing Primary Hierarchies



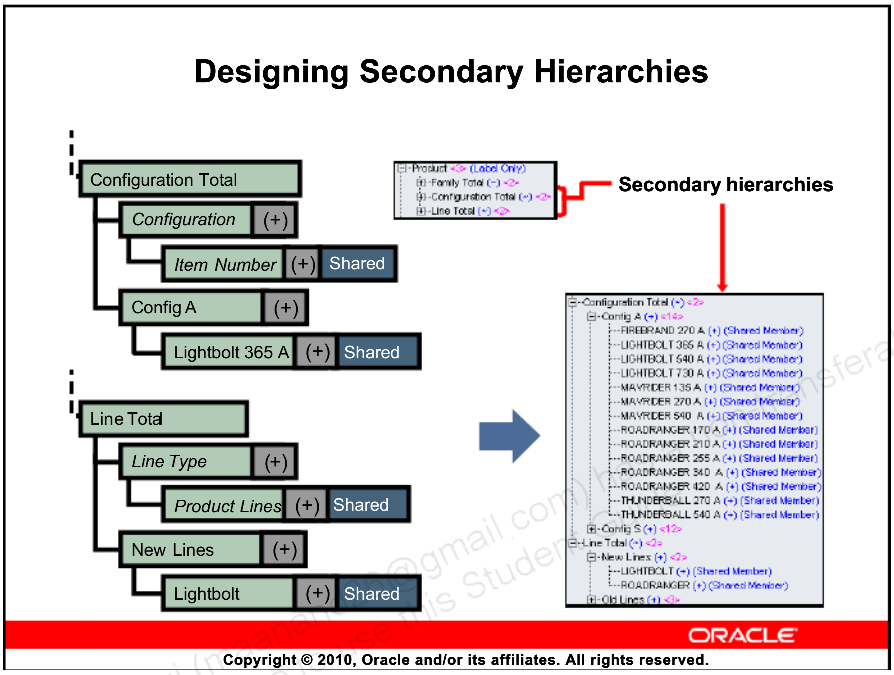
ORACLE

Designing Primary Hierarchies

The primary hierarchy of a dimension is typically the hierarchy with the most levels of detail. Ideally, this hierarchy includes all hierarchy elements that you want to share with alternate hierarchies.

The top member of the primary hierarchy is a generation 2 member, and should represent a total of all elements in the dimension, because this value is typically shared with the dimension member. For the sample Product dimension on the slide, Product is a label-only member, which means that it shares data with Family Total. This arrangement ensures that, when you retrieve the product dimension at the top level, you see a total for all products.

In the label outline example on the slide, the primary hierarchy includes the hierarchy total at generation 2, product families at generation 3, product lines at generation 4, and item numbers at generation 5.



Designing Secondary Hierarchies

You can use the same label outline procedure for analyzing alternate rollups and for identifying levels at which member sharing occurs.

For the first label outline example on the slide, the Configuration Total hierarchy includes the hierarchy total at generation 2, product configuration types at generation 3, and item numbers at generation 4. The item numbers are shared members that index level 0 members in the primary hierarchy.

For the second label outline example on the slide, the Line Total hierarchy includes the hierarchy total at generation 2, line types at generation 3, and product lines at generation 4. The product lines are shared members that index level 1 members in the primary hierarchy.

The top member of a secondary hierarchy may or may not represent a total of all dimension elements and should always be marked ignore during consolidation (~). This way, shared members are not double-counted during the consolidation process.

Summary

In this lesson, you should have learned to:

- Describe business view dimensions
- Identify attributes in database design
- Combine business views
- Design dimensions with label outlines

LESSON 6

Creating Basic Dimension Build Rules Files

Objectives

At the end of this lesson, you should be able to:

- Describe rules files
- Prep Data Prep Editor
- Create dimensions in rules files
- Select dimension build methods
- Define field properties
- Validate dimension build rules files
- Complete dimension build rules files
- Configure dimension maintenance settings

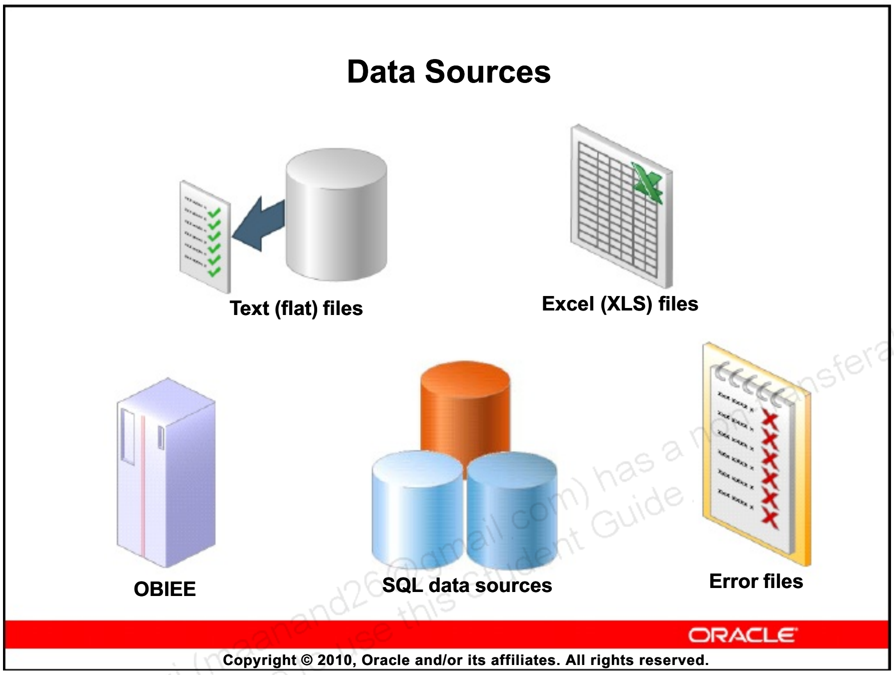
Rules Files Overview

- Load data or build dimensions
- Leverage external data sources
- Created with Data Prep Editor

Rules Files Overview

Rules are stored sets of operations that Essbase performs on data values or on dimensions and members when it processes a data source. A rules file can contain rules for loading data, rules for updating dimensions and members, or both. Although you can load data to databases without using rules files, you must use rules files when you want to update dimensions and members from information in a data source.

You create rules files in Administration Services Console, using Data Prep Editor.



Data Sources

Data sources contain the information that you load to the Essbase database or Essbase outline. Data sources for dimension building can contain member names, member aliases, formulas and consolidation properties, generation and level names, currency names, data storage properties, attributes, and UDAs; data sources for loading data typically contain only member names and data values.

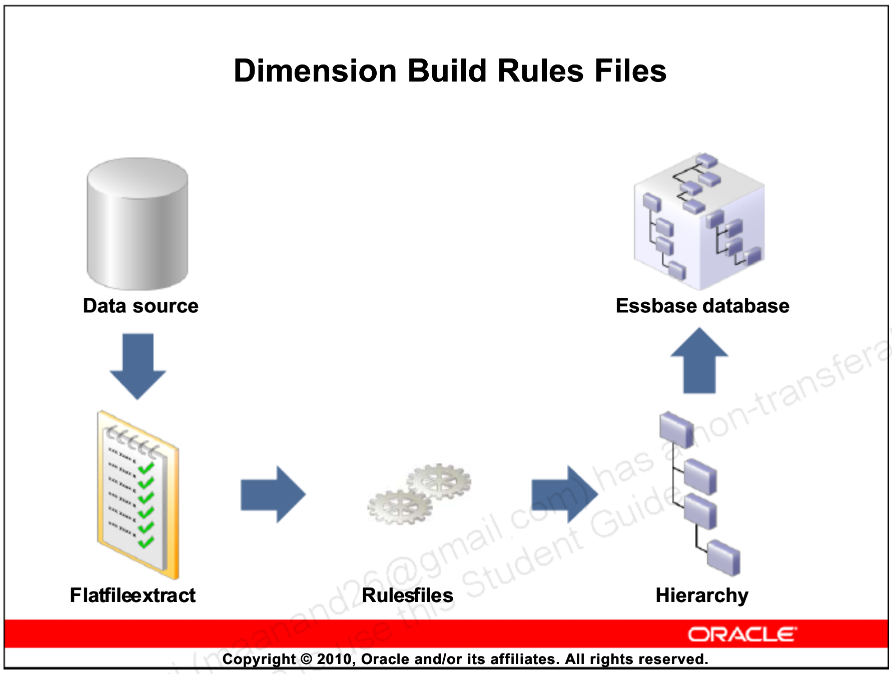
Key Terminology

- **Field**—an individual value in the data source
- **Delimiter**—a character or space indicating the start and end of fields
- **Record**—a structured row of related fields, separated by delimiters

A data source consists of records, which themselves are composed of fields and delimiters. Essbase reads data sources by starting with the first record at the top and proceeding from left to right.

Valid Data Sources

- Text files (flat files) from text backups or external sources
- SQL data sources
- Microsoft Excel files
- Error files generated from a previous dimension build
- Oracle BI Enterprise Edition (OBIEE)



Dimension Build Rules Files

When you create dimensions with thousands of members or with many alternate hierarchies, you want to automate the process, not manually add the members.

You use dimension build rules files to map dimensions and members from a data source to an Essbase outline. During dimension building, the rules file tells Essbase which build method to use, specifies hierarchy designations for members, and tells Essbase how to transform members before loading them.

TIP: It is best to create a separate rules file for each dimension.

The example on the slide shows the process for creating outline hierarchies from external data sources:

1. The data source is prepared, either by creating a SQL query in a rules file, formatting an Excel file, or generating a flat file.
2. You apply a rules file to the data source by executing a dimension build.
3. When you execute the dimension build, Essbase reads the members in the data source, changes them based on the rules in the rules file, and determines the hierarchy structure for the members.
4. Essbase loads the hierarchy into the outline without changing the srcinal data source.

NOTE: You can use a single rules file for any data source that requires the same set of rules.

Hierarchy Management
Use rules files to address the following issues of hierarchy management:

- Loading complex hierarchies
- Loading thousands of member at once
- Sorting, adding, and deleting members
- Automating hierarchy maintenance with batch processes

Troubleshooting
You can also use rules files to troubleshoot the following issues with your data:

- Changing the order of data fields
- Duplicating, parsing, and concatenating fields to construct hierarchies
- Adding prefixes or suffixes to names to provide clarity and to comply with unique member name requirements
- Creating aliases

NOTE: You cannot build dimensions from external data sources without rules files.

Creating Dimension Build Rules Files

General Setup	1. Open a sample data source. 2. Set data source properties. 3. Hide raw data (optional). 4. Set the view to dimension build fields. 5. Associate the rule with a database outline. 6. If necessary, format the file.
Actions	7. If necessary, create a dimension. 8. Select the dimension build method. 9. Define field properties.
Completion	10. Validate the rule. 11. Save the rule. 12. Update the outline.

ORACLE

Creating Dimension Build Rules Files

Before beginning the procedure outlined on the slide and in the following table, you must open Data Prep Editor. You can use the following table for a quick reference when creating dimension build rules files. The table lists each step in the procedure with its corresponding menu and menu command:

Procedure Step	Menu	Command
1. Open a sample data source.	File	**For flat files, Excel files, or error files:** Open data file **For SQL data sources:** Open SQL
2. Set data source properties.	Options	Data source properties
3. Hide raw data (optional).	View	Raw data (toggle)
4. Set the view to dimension build fields.	View	Dimension build fields (toggle)

Procedure Step	Menu	Command
5. Associate the rule with a database outline.	Options	Associate outline
6. If necessary, format the file.	Field	Various
7. If necessary, create a dimension.	Options	Dimension build settings (Dimension Definition tab)
8. Select the dimension build method.	Options	Dimension build settings (Dimension Build Settings tab)
9. Define field properties.	Field	Properties (Dimension Build Properties tab)
10. Validate the rule.	Options	Validate
11. Save the rule.	File	Save
12. Update the outline.	Outline (in Outline Editor)	Update Outline

TIP: Do not use the Open toolbar button to open a data source in your rules file; for Data Prep Editor to read a data source, you must select “Open data file” from the File menu.

Prepping Data Prep Editor (Steps 1–6)

Steps 1 through 6 ensure that you do the f ollowing:

- Provide yourself with samples of data on which to base your rules.
- Define correct file delimiters.
- Identify and ignore header records.
- Set the view in Data Prep Editor to the correct mode fo r dimension building.
- Create an association betwee n the rules file and the database outline that you want to update.
- Format data, if necessary.

Prepping Data Prep Editor (Steps 1–6)

The general setup steps in the procedure are to prepare Data Prep Editor for building the rules file. You do not have to execute these steps in any particular order, but if you follow a standard setup routine, you are less likely to forget an important step. Therefore, it is recommended that you complete the steps in the same order each time you create a rules file.

The general setup steps ensure that you do the following:

- Provide yourself with samples of data on which to base your rules
- Define correct file delimiters
- Identify and ignore header records
- Set the view in Data Prep Editor to the correct mode for dimension building
- Create an association between the rules file and the database outline that you want to update

Step 1: Opening a Sample Data Source

After opening Data Prep Editor, you can open data sources such as text files, spreadsheet files, and SQL data sources. The data source provides a frame of reference for building the rules file; it is displayed in Data Prep Editor so that you can see what needs to be changed and how to create your mappings. The sample data source that you use to build the rule should have the same format as the actual data source that you use to update your members.

To open text or spreadsheet files:

1. From the File menu, select **Open data file**.

   The Open dialog box is displayed.
2. Browse to the file location, and then select the file.
3. Click **OK**.

   Essbase reads the data records into Data Prep Editor.

To open SQL data sources:

1. From the File menu, select **Open SQL**.

   The Select Database dialog box is displayed.
2. Select the Essbase server, application, and database for which you are creating the rule.
3. Click **OK**.

   The Open SQL Data Sources dialog box is displayed.
4. From the “SQL data sources” list, select the desired data source.
5. In the Connect text boxes, enter your connection information.
6. In the SQL Statement text boxes, enter your SQL query.
7. Click **OK/Retrieve**.

   Essbase reads the queried data records into Data Prep Editor.

NOTE: You must configure data source names and ODBC connections before you open SQL data sources. For instructions, see the *SQL Interface Guide*.

Step 2: Setting Data Source Properties

This step comprises two substeps: selecting a file delimiter for the data source and identifying and ignoring header records in the data source.

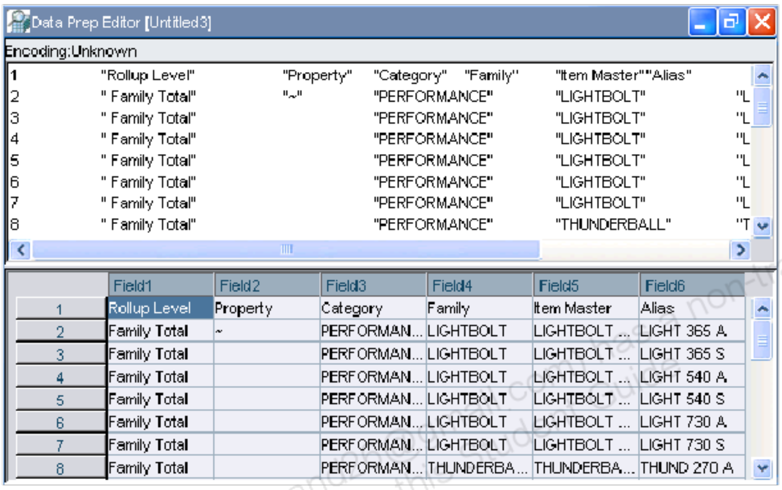
- **File delimiter**—The character used to separate fields in the data source. By default, a rules file expects fields to be separated by tabs. You can set the file delimiter to be a comma, tab, space, fixed-width column, or custom value. You do not need to set file delimiters for SQL data sources.
- **Header records**—Records, usually located at the top of the data source, that contain information describing the data source contents. You must instruct the rule to skip these records.

To set data source properties:

1. From the Options menu, select **Data source properties**

   The Data Source Properties dialog box is displayed.
2. On the Delimiter tab, select the appropriate file delimiter.
3. Select the **Header** tab.
4. In the “Number of lines to skip” text box, enter the appropriate number.
5. Click **OK**.

Step 3: Hiding Raw Data (Optional)
By default, the Data Prep Editor window is organized into two frames. As shown in the following example, the top frame contains your source (raw) data, and the bottom frame shows how the rules file presents the data to Essbase.



The two frames enable you to compare the rules file results to the srcinal data source. However, because many steps in the dimension build rules file procedure involve opening dialog boxes on top of the rules file, hiding the source data can give you a better view of the rule information as you complete the procedure.

You can toggle the raw data frame on and off from the View menu.

Step 4: Setting the View to Dimension Build Fields
When developing rules files, you work in one of the following view modes:

- **Dimension build fields view**—Use this view when you create rules to automate loading or maintenance of members in the outline.
- **Data load fields view**—Use this view when you create rules to load data (for example, units and dollars) to members in a database.

Although you may not see a difference in the rule when you change the view mode, it has a pronounced effect on later steps in the procedure, namely assigning field properties and validating your rule.The default view in Data Prep Editor is “Data load fields.”

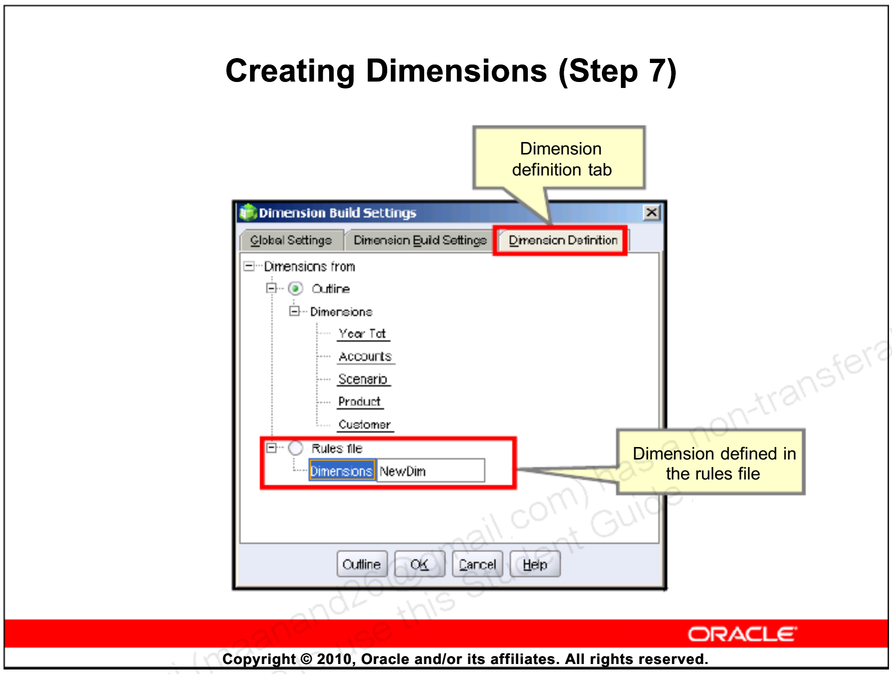
Step 5: Associating the Rule with a Database Outline
You must associate your rules file with the database outline for which you are creating the rule. Similar to step 4, this step provides no visual cues after completion, but if you do not associate your rule with an outline, you cannot successfully construct or validate the rules file for errors.

Because the outline association is not saved as part of the rules file, you can use a rules file created for one database against other databases. For example, if two databases include the same Product dimension, you can create one rules file and load the same source data to both databases. However, you must reassociate the rules file with a

database outline every time you open the saved rule file.
You associate an outline with your rules file from the Options menu.

Step 6: Formatting the File (If Necessary)
If your data source columns are not in the correct order for building the dimension or if you must format the columns in other ways, you can use the following Field menu commands to format data source columns:

- Move
- Split
- Join
- Create using join
- Create using text



Creating Dimensions (Step 7)

Essbase gives you the option to define a new dimension (its name and properties) in a rules file instead of adding the dimension manually to the outline. You may want to use this option under the following circumstances:

- During the design phase of your implementation, to build rules files for dimensions that are not yet added to the outline
- After you deploy the database and as you maintain dimensions, to delete and rebuild a dimension on a regular basis

To create dimensions in rules files:

1. From the Options menu, select **Dimension build settings**

   The Dimension Build Settings dialog box is displayed.
2. Select the **Dimension Definition** tab.

3. Select **Rules File**.
4. In the Dimension text box, enter the dimension name, and then press **Enter**.

You can also use the Dimension Definition tab to set properties for new or existing dimensions.

To set dimension properties in rules files:

1. From the Options menu, select **Dimension build settings**.

   The Dimension Build Settings dialog box is displayed.
2. Select the **Dimension Definition** tab.
3. Right-click a dimension, and then select **Edit properties**.

   The Dimension Properties dialog box is displayed.
4. Select the appropriate settings, and then click **OK**.

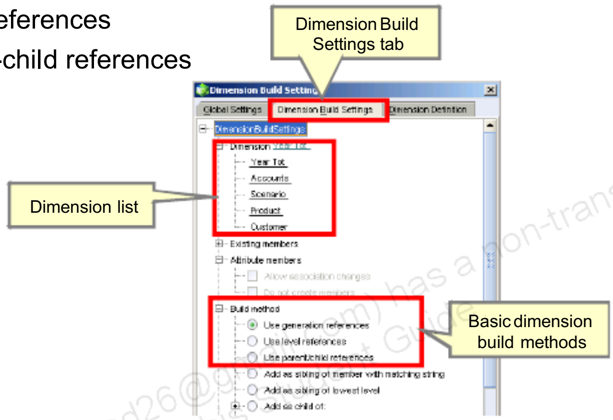
Some dimension properties are valid only in specific database types. The Dimension Properties dialog box does not differentiate between database types when you construct rules files. You can use the following table as a guide to the properties that are valid for each database type.

Property Category	Properties Valid in Block Storage	Properties Valid in Aggregate Storage
Dimension types	• Existing definition or none • None • Accounts • Time • Country • Currency name	• Existing definition or none • None • Accounts • Time
Member names unique in dimension (Duplicate member names must be enabled in the outline)	• Existing definition or unique • Member names unique in dimension • Member names duplicated in dimension	• Existing definition or unique • Member names unique in dimension • Member names duplicated in dimension

Property Category	Properties Valid in Block Storage	Properties Valid in Aggregate Storage
DataStorage	• Existingdefinitionorstore • Store data • Never share • Label only • Dynamic calc and store • Dynamic calc	• Existing definition or store • Store data • Label only
Configuration	• Existingdefinitionorsparse • Dense • Sparse	not applicable
Hierarchytype	notapplicable	• Existingdefinitionorstored • Multiple hierarchy enabled • Stored • Dynamic
Dimensionsolveorder	notapplicable	• Existingvalueor0 • New value
Solveorder	notapplicable	• Existingvalueor0 • New value
AggLevelUsage	not applicable	• Default • Consider all levels • Do not aggregate • Consider bottom level only • Consider top level only • Never aggregate to intermediate levels

Selecting the Dimension Build Method (Step 8)

- Generation references
- Level references
- Parent-child references



ORACLE

Selecting the Dimension Build Method (Step 8)

Whether you are creating a dimension or adding members to a dimension, you must tell Essbase what build method to use. You must specify a build method for each dimension that you create or modify in a rules file.

The build method that you select determines the algorithm that Essbase uses to add, change, or remove dimensions, members, and aliases in the outline. The kind of build method that you select depends on the layout of the records in the data source.

The three primary build methods for building and maintaining standard dimensions are generation references, level references, and parent-child references. The following table provides guidelines to help you select the appropriate build method for the data source by listing the build method, type of data in each record, and examples of records for the build method:

BuildMethod	TypeofDatainEachRecord	Examples
Generation references	Top-down data: Each record specifies the parent name, the child name, the name of the child of that child, and so on.	Year, Quarter, Month
Level references	Bottom-up data: Each record specifies the name of the member, the name of its parent, the name of the parent of its parent, and so on.	Month, Quarter, Year
Parent-child references	Parent followed by its child: Each record specifies the name of the parent and the name of the child, in that order, although other information (such as member properties) can also be specified.	Year, Quarter Quarter, Month

To select dimension build method:

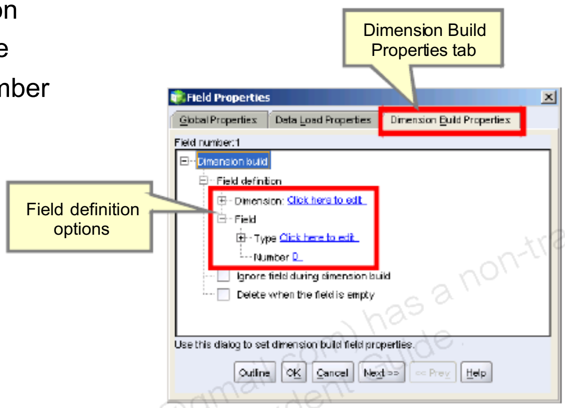
1. From the Options menu, select **Dimension build settings**.

   The Dimension Build Settings dialog box is displayed.

2. Select the **Dimension Build Settings** tab.
3. Under Dimension, double-click the dimension for which you are selecting a build method.
4. Under "Build method," select the appropriate build method.

Defining Field Properties (Step 9)

- Dimension
- Field type
- Field number



Defining Field Properties (Step 9)

In a dimension build, each field in the data source is part of a column that describes a member in the outline. Fields can contain information about member names, member properties, or attribute associations. For Essbase to process this information, you must specify the following information when defining fields:

- **Dimension**—The dimension to which members of the current field belong
- **Field type**—The type of information in the field, such as a generation or alias (The field type you select depends on the data source and the build method that you selected in the preceding step.)
- **Field number**—The generation or level number of the members in the current field

To define dimension build field properties:

1. From the Field menu, select **Properties**.

   The Field Properties dialog box is displayed

2. Select the **Dimension Build Properties** tab.
3. Under "Field definition," complete the following tasks:
   - For Dimension, double-click or enter the name of the dimension that you are building.
   - For Field Type, double-click or enter the appropriate type for the selected field.
   - For Field Number, enter the appropriate number for the selected field.
4. Click **Next** to continue to the next field, or click **OK** to finish.

Field Types

The following table gives definitions for the most common field types and the build methods for which they are valid:

Field Type	Definition	Valid Build Methods
Generation	The name of a member in the specified generation	Generation references
Level	The name of a member in the specified level	Level references
Parent	The name of a parent	Parent-child references
Child	The name of a child	Parent-child references
Alias	A member alias	Any
Property	A member property (For a complete list of valid member property codes, see the *Oracle Essbase Database Administrator's Guide*.)	Any
Formula	A member formula	Any
UDA	A member UDA (user-defined attribute)	Any

Assigning Field Numbers: Generation Build

The numbers that you assign to fields depend on the build method and field type that you select. Use the following rules as a guide for assigning field numbers when using the generation build method:

- Generation 1 (the dimension name) is not valid.

- If generation numbers do not start at 2, the members of the first specified generation must currently exist in the outline.
- Generation numbers must form a contiguous range. For example, if generation 3 and generation 5 exist, you must also define generation 4.
- Group GEN fields sequentially within a dimension; for example:

```
GEN2,PRODUCTGEN3,PRODUCT GEN4,PRODUCT
```

Assigning Field Numbers: Level Build

Use the following rules as a guide for assigning field numbers when using the level build method:

- Each record must contain a level 0 member.
- Level numbers must form a contiguous range. For example, if Level 0 and Level 2 exist, you must also define Level 1.
- Group LEV fields sequentially within a dimension; for example:

```
LEV0,PRODUCTLEV1,PRODUCT LEV2,PRODUCT
```

Assigning Field Numbers: Parent-Child Build

If a field type is parent or child, enter 0 (zero) in the Number text box.

NOTE: For all build methods, place alias, property, formula, and UDA fields after the member field with which they are associated, and specify the same number as the associated member; for example, an alias field for generation 3 products is `ALIAS3, PRODUCT`.

Validating Dimension Build Rules Files (Step 10)

Common validation errors:

"There is an unknown member or no member in the field name."

"The field type for this field is not valid for the build method associated with this field's dimension."

"The LEVEL number for this field is not contiguous (starting with 0) with other fields' LEVEL values."

Validating Dimension Build Rules Files (Step 10)

You must validate your rules file to ensure that the members and dimensions in the rules file map to the associated outline. While a correct validation does not ensure that the data source loads properly, validation errors typically result in an incorrect build.

The following are common validation error messages and their possible resolutions:

There is an unknown member or no member in the field name.

- The view mode is not currently set to dimension build fields, and Essbase is validating data load field rules. Change the view mode to dimension build fields.
- Field properties are not defined for the specified fields. Define the field properties, or set the field to be ignored during dimension builds.

The field type for this field is not valid for the build method associated with this field's dimension.

Your field type does not correspond to your chosen build method. Change the build method to reflect field designations, or change field designations to match the build method.

The LEVEL number for this field is not contiguous (starting with 0) with other fields' LEVEL values.

A higher level (or generation) number was assigned without a supporting lower level.

Change the numeric designations to be contiguous.

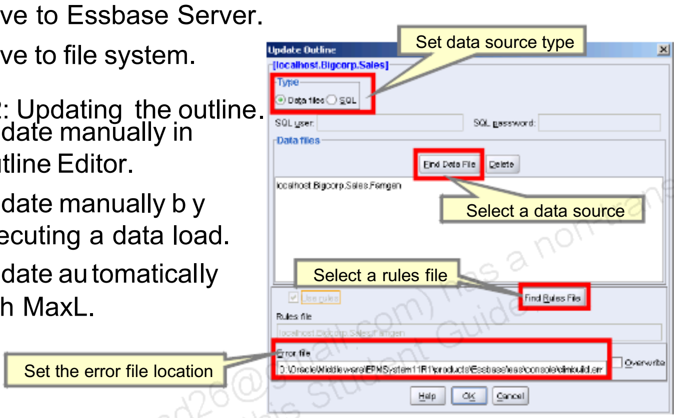
Completing Dimension Build Rules Files (Steps 11 and 12)

Step 11: Saving the rules file.

- Save to Essbase Server.
- Save to file system.

Step 12: Updating the outline.

- Update manually in Outline Editor.
- Update manually by executing a data load.
- Update automatically with MaxL.



Completing Dimension Build Rules Files (Steps 11 and 12)

The dimension build rules file procedure ends with saving the rule, either to an Essbase server or to a file system location, and finally updating the outline.

Step 11: Saving Rules Files

You can reuse a single rules file with multiple data sources and Essbase databases. By selecting the appropriate tab in the Save dialog box, you can choose to save your rule to a location on an Essbase server (in a database or application directory) or to a location in your file system. Essbase saves rules files with a RUL extension.

Step 12: Updating the Outline

After your dimension build rules file is complete, you can use it to load dimensions and members into your outline. You can use any of the following methods to update the outline:

- Update manually in Outline Editor

- Update manually by executing a data load
- Update automatically with a MaxL script

Of the three choices, only the first gives you the option of discarding the changes to the outline after Essbase executes the dimension build. After executing a dimension build in Outline Editor, you can close the outline without saving the changes made by the load. However, when you execute a dimension build through the data load process or a MaxL script, Essbase performs an automatic save and restructure of the database. During the design phase of an implementation, the Outline Editor method is commonly used; after database deployment, it is more common to use an automatic update.

To update the outline in Outline Editor:

1. From the Outline menu, select **Update Outline**.
2. Select a data source type.
3. Select source data files.

NOTE: For SQL data sources, you do not select data files. Instead, you enter your SQL user name and password in text boxes.

4. Select a rules file.
5. Specify an error file location.

Configuring Dimension Maintenance Settings

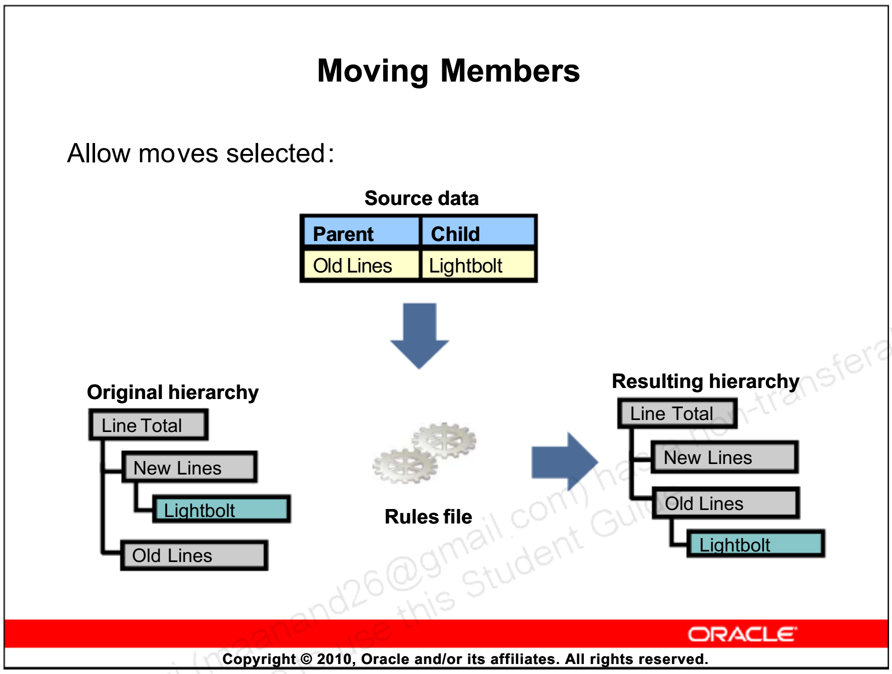
Considerations for updating dimensions:

- Moving members
- Modifying member properties
- Sorting members
- Selecting member update mode

Configuring Dimension Maintenance Settings

When updating, rather than creating, dimensions, consider the following when building rules files.

- Whether to allow moves of members
- Whether to allow modification of member properties
- Whether to sort members
- Whether to use the data source to modify or replace the dimension



Moving Members

By default, Essbase does not move a member in the hierarchy, even when the outline position of the member differs from the position of the member in the data source.

The following table, which applies only to dimensions with unique member names, shows the default behavior of each dimension build method when the parent of a member in a data source record differs from the parent of the member in the outline.

Build Method	Dimension Build Result
Generation references	Essbase skips the record and writes it to the error file. There is no change to the member
Level references	Essbase skips the record and writes it to the error file. There is no change to the member.
Parent-child references	Under the new parent, Essbase creates a shared member that references the srcinal member.

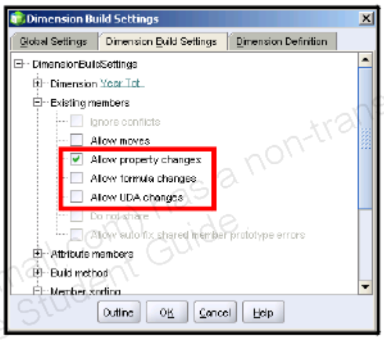
For dimensions in which duplicate names are allowed, when data source and outline positions differ, all three build methods create a duplicate member name under the new parent.

In some cases, you may want to move members in the hierarchy. For the example on the slide, the Sales group wants to realign the Lightbolt product line with the Old Lines total. To realign, you create a rules file with Allow Moves specified in the Dimension build settings. With Allow Moves selected, Essbase moves Lightbolt from New Lines to Old Lines, instead of returning an error or creating a shared member.

Modifying Member Properties

Essbase requires explicit commands to:

- Update member properties
- Update member formulas
- Replace member UDAs



Modifying Member Properties

As you maintain your outline, you may need to change the properties, formula, or UDA assignments of a member. The Dimension Build Settings dialog box provides three check boxes for updating member information:

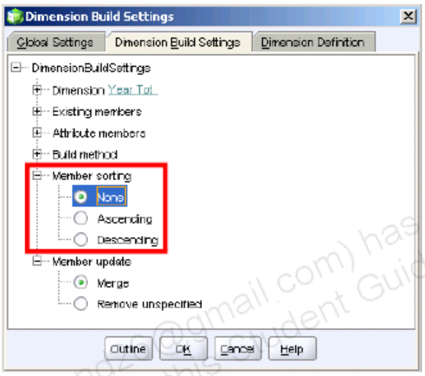
- **Allow property changes**—This option enables you to update consolidation operators, data storage properties, account properties, and any other element of member information that you can assign with a property field. By default, if Essbase encounters a property field for an existing member in a dimension build rule, it returns the error "Property changes for member *MemberName* not allowed."
- **Allow formula changes**—This option enables you to update member formulas from information in a formula field. By default, if Essbase encounters a formula field for an existing member in your dimension build rule, it returns the error "Formula changes for *MemberName* not allowed."

- **Allow UDA changes**—This option enables you to replace a member's existing UDAs with the ones listed in the data source. By default, if Essbase encounters a UDA field for an existing member in your dimension build rule, it creates a UDA and leaves the srcinal UDA intact. For example, the customer IBM has a UDA of Class A. To change the UDA for IBM to Class L, you must select Allow UDA change; otherwise, IBM will be assigned both Class A and Class L UDAs after dimension build.

NOTE: Before selecting "Allow UDA changes," you must select "Allow property changes."

Sorting Members

- Sort in ascending or descending order
- Sort entire dimensions



ORACLE

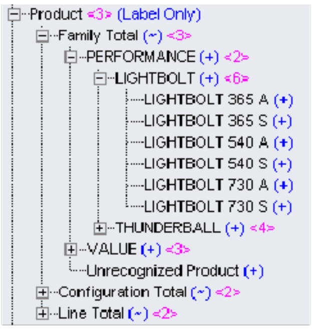
Sorting Members

When you add members to a dimension, you may want to enforce a sequential order. By default, Essbase adds a member as the last child of its parent.

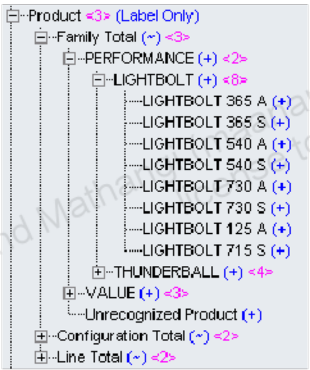
You can have Essbase arrange members within a dimension in alphabetical order (A to Z) or reverse alphabetical order (Z to A).

TIP: Because sorting is applied to the entire dimension, you should use it only on single-hierarchy dimensions with simple aggregation consolidations. Sorting dimensions with multiple hierarchies or complex consolidation paths can have unpredictable results.

In the following example, Lightbolt products are listed in numeric order, from Lightbolt 365 A through Lightbolt 730 S:



You run a dimension build that includes the new products Lightbolt 125 A and Lightbolt 715 S. With no sort options selected, Essbase adds the two new products at the end of the list of products, after Lightbolt 730 S, as shown in the following figure:



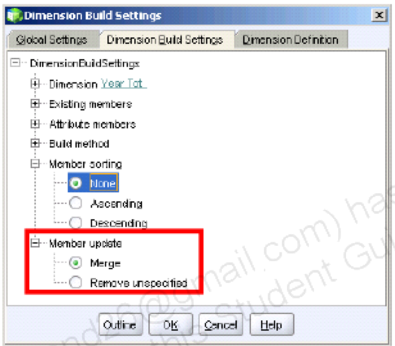
You can correct the sequence automatically during the dimension build by applying a member sorting option in the Dimension Build Settings dialog box.

When sorting is enabled, keep in mind that Essbase applies sorting to every set of siblings in the dimension that you are building, not just to the members being loaded. If you applied sorting during the preceding example, Essbase would place the Lightbolt products in the correct sequence. However, Essbase would also move the Configuration Total hierarchy before Family Total, thus interfering with the shared member references and changing how users view data.

Updating Members

Two load modes:

- Merge
- Remove unspecified



Updating Members

Essbase provides two member update modes for dimension building:

- Merge—Add members to a dimension
- Remove unspecified—Replace an entire dimension

Although most dimension maintenance builds simply add members, in some cases, you may want to replace a dimension in its entirety. In these situations, you can select the “Remove unspecified” update option in the Dimension Build Settings dialog box. When you do, Essbase deletes all members of the dimension and rebuilds the dimension from the information in the data source.

During this process, Essbase also deletes all data associated with members that are deleted and not subsequently rebuilt. Members that are rebuilt retain their data.

Therefore, use this option with caution.

Summary

In this lesson, you should have learned to:

- Describe rules files
- Prep Data Prep Editor
- Create dimensions in rules files
- Select dimension build methods
- Define field properties
- Validate dimension build rules files
- Complete dimension build rules files
- Configure dimension maintenance settings

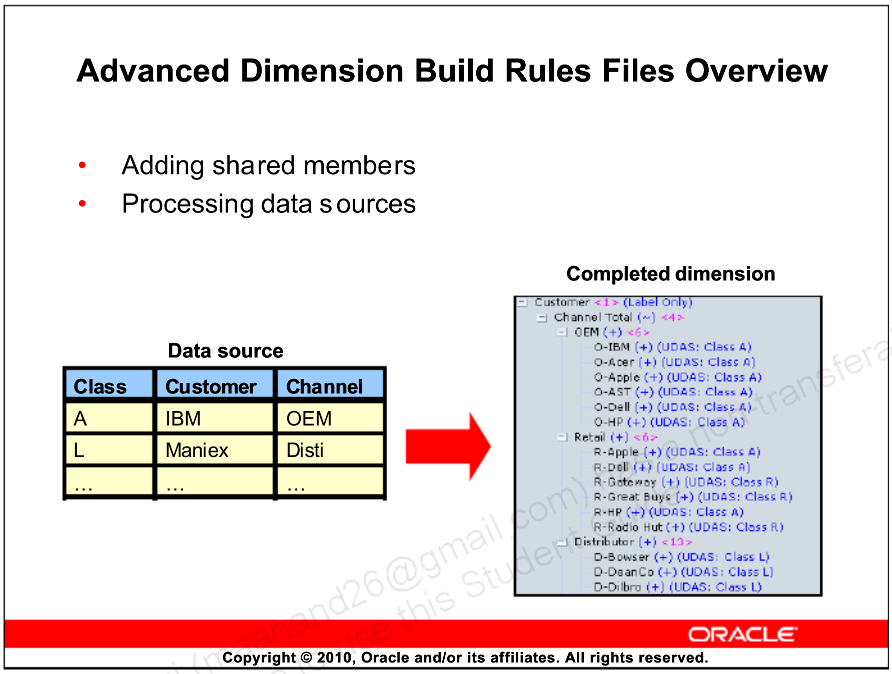
LESSON 7

Creating Advanced Dimension Build Rules Files

Objectives

At the end of this lesson, you should be able to:

- Describe advanced dimension build rules files
- Create shared members
- Arrange, alter, and ignore fields
- Create used-defined attributes
- Create attribute dimensions



Advanced Dimension Build Rules Files Overview

Although the 12-step procedure for creating dimension build rules provides a solid starting point for creating rules files, you may encounter dimension build tasks or data source issues that require special consideration:

- Adding shared members to your outline
- Processing data sources that are out of order or include member names that do not match what you want in your outline

The example on the slide shows a data source listing customer classes (a single-letter code), customer names, and channels. You can create a rules file to convert the information in the sample data source to the completed customer dimension shown on the slide. The dimension displays channels as children of the Channel Total member, and customers with channel-specific prefixes (for example, R- as the retail channel prefix) as children of channels. The customer classes are given a prefix of “Class” and assigned to customers as UDAs.

Creating Shared Members

- Parent-child references
- Other methods
  - Sharing members at the same generation
  - Sharing members at different generations
  - Sharing non-level 0 mem bers

Shared members

Creating Shared Members

The data associated with a shared member comes from a stored member with the same name as the shared member. The shared member creates a pointer to data contained in the stored member; thus, the data is shared between the members and is stored only once.

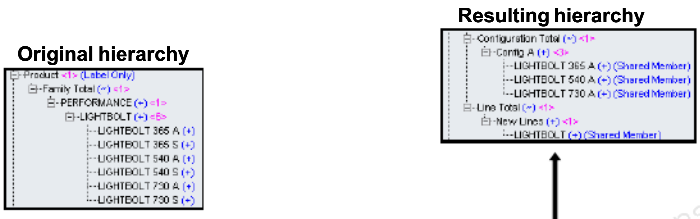
Adding shared members in Outline Editor is a simple process; You assign the Shared data storage property to the member. But the process of using a rules file to build shared members differs from the process of using a rules file to build stored members; you must choose the build method and format the data source carefully.

Parent-child references automatically create shared members at any level of your hierarchy. If you cannot use parent-child references, then your ultimate hierarchy design substantially drives the methods you can use to create shared members.

Before determining a build method to use, you must determine which sharing scenario best describes your hierarchy:

- Sharing members at the same generation
- Sharing members at different generations
- Sharing non-level 0 members

Parent-Child Build Method



PARENT0,PRODUCT	CHILD0,PRODUCT
Configuration Total	ConfigA
ConfigA	Lightbolt365A
LineTotal	NewLines
NewLines	Lightbolt
…	…

Parent-Child Build Method

Parent-child references provide the most flexible method for loading and maintaining shared members. By default, a parent-child build automatically adds shared members, without the field designations that are required in level and generation builds. The principal requirements are that a matching stored member exists in the outline and that the matching member has a parent other than the parent of the shared member that Essbase is currently loading.

Creating and maintaining shared members with parent-child references solves a variety of sharing issues that are difficult to manage with other methods. For example:

- Creates shared members automatically without special setup requirements
- Enables sharing at any level or generation (For the example on the slide, the Configuration Total shared members and the Line Total shared members are added in one build, despite being shared at different levels or generations.)
- Enables building complete secondary rollup hierarchies in one build (With other build methods, it is more complicated to add complete secondary rollups.)

NOTE: Selecting Do Not Share in the Dimension Build Settings dialog box disables all sharing, for the selected dimension, during a parent-child build.

For the example on the slide, the srcinal product hierarchy is shown with the product family Lightbolt and its children (such as Lightbolt 365 A) consolidating to the member Performance. which is a child of the member Family Total. The example also shows a parent-child data source for building an alternate shared-member hierarchy, The following table lists the parent and child information in the sample data source:

Parent	Child
Configuration Total	Config A
Config A	Lightbolt 365 A
Line Total	New Lines
New Lines	Lightbolt

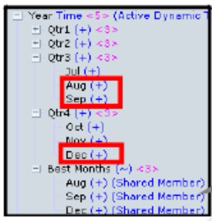
The resulting hierarchy shown on the slide is created by running a parent-child dimension build based on this sample data source—Essbase creates two new hierarchies in the product dimension by adding members in the following order:

1. Configuration Total is added as a child of the dimension Product (as a sibling to Family Total).
2. Config A is added as a child of Configuration Total.
3. Lightbolt 365 A is added as a shared member under Config A.
4. Line Total is added as a child of the dimension Product (as a sibling to Configuration Total).
5. New Lines is added as a child of Line Total.
6. Lightbolt is added as a shared member under Line Total.

TIP: Members listed in the Parent field before being listed in the Child field are automatically added as children of the dimension member (for example, Configuration Total and Line Total).

Other Methods for Creating Shared Members

- Sharing members at the same generation
- Sharing members at different generations
- Sharing non-level 0 members



Level 0 members shared at the same generation

ORACLE

Other Methods for Creating Shared Members

If you cannot use parent-child references, the rules for adding shared members with rules files are more complex and depend on your completed hierarchy design. For example, generation references can be used only to add shared members when the shared members are at the same generation in the hierarchy and the parents of the shared members belong to the same hierarchy branch.

TIP: For detailed descriptions and examples of alternative methods for building shared members, see “Building Shared Members by Using a Rules File” in the *Oracle Essbase Database Administrator’s Guide*.

Different rules file methods exist for different types of shared members. The following table lists types of shared members and valid build methods for each type:

Shared Member Type	Valid Build Methods
Same-generation shared members, where shared member parents are in the same hierarchy branch as srcinal parents	• Generation build (DUPGEN) • Level build • Parent-child build
Different-generation shared members, or same-generation shared members where shared member parents are in different hierarchy branches from srcinal parents	• Level build • Parent-child build
Non-level 0 shared members	• Level build (DUPLEV) • Parent-child build

The slide shows an example of shared members that can be added using generation references. The Year dimension contains quarters at generation 2 and months at generation 3. An additional generation 2 member called Best Months consolidates data from August, September, and December, all listed as shared members under Best Months.

You can build the hierarchy shown on the slide by creating a data source that includes the shared member and both the srcinal parent and the shared member parent. In the following table, Field 1 contains the srcinal parent, Field 2 contains the shared member parent, and Field 3 contains the member to be shared between the two parents:

Field 1	Field 2	Field 3
Qtr3	Best Months	Aug
Qtr3	Best Months	Sep
Qtr4	Best Months	Dec

In your rules file, you define the fields as follows:

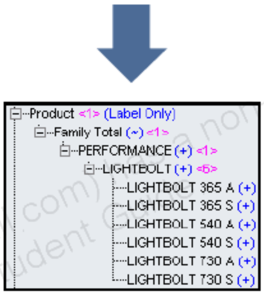
- Field 1: GEN2,Year
- Field 2: DUPGEN2,Year
- Field 3: GEN3,Year

Manipulating Fields

Arranging fields:
- Move
- Split
- Join
- Create using join
- Create using text

Altering fields:
- Replace text
- Add prefixes and suffixes
- Ignore fields

Group	ModelD	Line	Type
Performance	365-15-150	Lightbolt	ATA
Performance	540-01-100	Lightbolt	SCSI
...	...	...	...



ORACLE

Manipulating Fields

Data sources may contain information that is irrelevant or unusable in its srcinal format. One of the challenges that you encounter as a database developer is how to process an unformatted data source so that it accomplishes the following objectives:

- Returns all information needed to build your outline hierarchy
- Formats names as you want them formatted
- Conforms to Essbase rules for unique names (not required in dimensions that enable duplicate names)

Rules files give you a two-step alternative for addressing these issues, by preprocessing source data before loading it to your database:

1. Arrange fields by using Field menu functions.
2. Alter fields by using Field Properties dialog box options.

The example on the slide shows a data source where fields must be rearranged and altered to produce the Product dimension in Bigcorp Sales.

Arranging Fields

Data source

Group	ModelD	Line	Type
Performance	365-15-150	Lightbolt	ATA
Performance	540-01-100	Lightbolt	SCSI
…	……		…

Final arrangement after rules file processing

FamilyTotal	Group	Line	LineModType	ellID
FamilyTotal	Performance	Lightbolt	Lightbolt 365 ATA	-15-150
FamilyTotal	Performance	Lightbolt	Lightbolt 540 SCSI	-01-100
……		…	…	…

ORACLE

Arranging Fields

Your data source fields may not be in the order required for dimension building, or you may need to duplicate, parse, or concatenate fields in your data source to produce the member names that you want in your outline. Rather than changing the data source, you can build field-arrangement actions into your rules files.

You can use the Field menu functions in the following table to arrange fields:

Function	Description
Move	Changes the order of fields (For dimension building, fields must be arranged in a specific order.)
Split	Parses fields at a given number of characters
Join	Joins two or more fields (often used to create unique member names)

Function	Description
Create Using Join	Joins two or more columns, creating a new field and leaving existing fields intact (can also be used to duplicate a single field)
Create Using Text	Creates fields based on text that you enter (The same text is displayed for all records in the data source. Often used to create spacing characters for joined fields.)

Essbase records your field arrangement actions in sequence in the rules file and follows the recorded sequence of actions when you execute the rule.

Arranging Fields Example: Overview

The example on the slide shows a data source for building the Family Total rollup in the Bigcorp Sales Product dimension. The following describes the four fields in the data source:

- **Group**—This field maps directly to the product family names (generation 3) in the Family Total rollup.
- **Model ID**—This field includes three hyphenated numeric codes. The first represents the model number, the second the manufacturing location, and the third the size of the hard drive.
- **Line**—This field maps directly to the product line names (generation 4) in the Family Total rollup.
- **Type**—This field notes the type of hard drive. Bigcorp hard drives are built in both ATA and SCSI interface configurations.

Arranging Fields Example: Analysis

You want to use generation references to convert the data source into a format that builds the Family Total hierarchy. However, the data source has a number of problems that require attention, as follows:

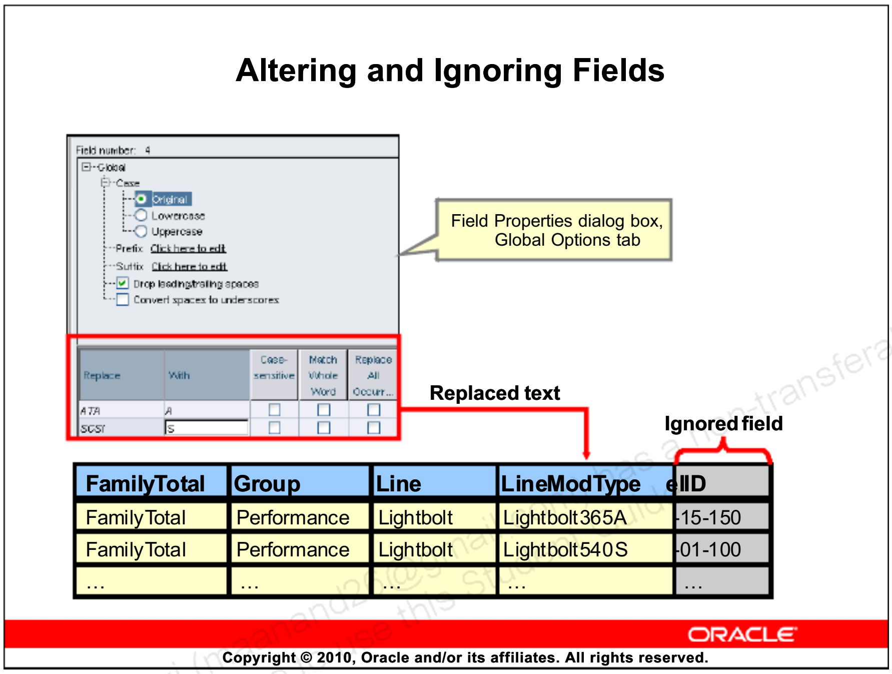
- The total for this hierarchy (Family Total) is not listed in the data source.
- The model ID code is not user-friendly. Additionally, your Essbase database design specifies that you need only the model number, not the manufacturing location or drive size.
- Model numbers are not unique from product line to product line, thus, you must prefix the model number with the product line name to create unique names.
- Product types must be incorporated into the product name instead of listed as a separate field.

- Information from the Model ID, Line, and Type fields must be combined to create unique product names. For example, a product in the Lightbolt product line with the model ID of 365-15-150 and a type of ATA should be represented in Essbase as Lightbolt 365 A.
- The fields are not in the correct order for a generation build.

Arranging Fields Example: Process
In the first part of the process, you arrange fields by completing the following sequence of actions:

1. Add a field for Family Total, using the “Create using text” function.
2. Move the Line field, using the Move function.
3. Isolate the model number (first three numbers) in the Model ID field, using the Split function.
4. Create a duplicate of the Line field, using the “Create using join” function.
5. Add spaces between the line, model number, and type fields, using the “Create using text” function.
6. Concatenate the line, model number, type, and space fields to create the complete product name (for example, Lightbolt 365 ATA), using the Join function.

NOTE: You can undo the last field operation that you performed by using the Undo command from the Edit menu. You can also undo field operations if you performed intervening actions. Undoing field operations is sequential; you must undo field operations from the last operation to the first operation. Essbase provides a sequential list of field operations on the Field Edits tab in the Data Source Properties dialog box, and you can delete the last action listed in the sequence.



Altering and Ignoring Fields

In addition to arranging fields, you may need to include rules to alter the text from the data source, either by replacing text, adding prefixes or suffixes, or ignoring superfluous fields.

Unlike field arrangement options, field alteration settings are defined in the Field Properties dialog box and are field-specific. The following table lists field alteration settings, provides the location of the settings in the Field Properties dialog box, and gives a brief description of each setting:

Field Alteration Setting	Field Properties Tab	Description
Case	Global Properties	Alter text case. Options include Original, Lowercase, and Uppercase.
Prefix	Global Properties	Prefix field values with a text string. Often used to create unique names.

Field Alteration Setting	Field Properties Tab	Description
Suffix	Global Properties	Suffix field values with a text string. Often used to create unique names.
Replace	Global Properties	Replace text in fields with a specified text string. You can also replace text with no string. Includes the following options: • **Case-sensitive:** Select to construct a condition based on a case-sensitive string. • **Match Whole Word:** Select to replace the text string only when it occurs as an entire word. If the text string is embedded in another word, it is not replaced. For example, you want to change the field value FIREBRAND 540 A to FIREBRAND 540 ATA, so you create a rule to replace A with ATA. Because the word FIREBRAND contains an A, if you do not select Match Whole Word (and assuming that Replace All Occurrences is not selected), the resulting string is FIREBRATAND 540 A. • **Replace All Occurrences:** Select to replace multiple instances of a text string in a single field value. By default, Essbase replaces only the first occurrence. For example, if you replace all occurrences of A with ATA (and assuming that Match Whole Word is not selected), FIREBRAND 540 A becomes FIREBRATAND 540 ATA.
Ignore during dimension build	Dimension Build Properties	Instruct Essbase to skip the field when processing the data source.

Altering and Ignoring Fields Example: Analysis

Continuing the Bigcorp example on arranging fields, after you arrange your data source fields, you need to alter some fields to map values to your outline, as follows:

- The remainder of the Model ID field (containing the manufacturing location and hard drive size) is unnecessary.
- The types (ATA and SCSI) must be abbreviated in the product name.

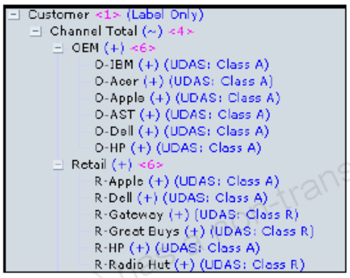
Altering and Ignoring Fields Example: Process

To finish the process of manipulating fields, you must complete the following tasks:

- In Field 4 (Line Mod Type), create rules to replace ATA and SCSI with A and S, respectively.
- Instruct Essbase to ignore Field 5 (el ID) during dimension builds.

Creating User-Defined Attributes

Uses for UD As:
- Calculation scripts
- Data loading
- Reports
- Security filters
- Partition definitions



Creating User-Defined Attributes

You can create UDAs for members. For the example on the slide, each customer in the Bigcorp Sales outline has a UDA that describes customer class. Class A customers are major accounts, Class R customers are retail accounts that are not major accounts, and Class L customers are distributor accounts.

Uses for UDAs

You can use UDAs in the following places:

- **Calculation scripts**—After defining a UDA, you can query a member for its UDA in a calculation script. For example, you create a budget that is 115% of actual data for Class A customers, 110% of actual data for Class R customers, and 105% of actual data for Class L customers.
- **Data loading**—Using a UDA, you can change the sign of a data value as it is loaded into the database.

- **Reports**—In a report, you can create a list of members with similar UDAs.
- **Security filters**—Using member UDAs, you can define a user's access to data and metadata.
- **Partition definitions**—Using member UDAs, you can define partition areas.

Rules for Creating UDAs

Follow these rules when creating UDAs:

- You can define multiple UDAs per member.
- You can set the same UDA for multiple members.
- A UDA name can duplicate a member, alias, level, or generation name.
- You cannot create a UDA on shared members or members of attribute dimensions.
- A UDA applies only to the specified member. Descendants and ancestors of the member do not inherit the UDA.
- You can define UDAs manually in the Member Properties dialog box.

You can create UDAs with rules files by using the UDA field type in the Dimension Build Properties dialog box.

TIP: By default, Essbase does not update UDAs when you run a dimension build. If you want to add a UDA to a member, select "Allow property changes" in the Dimension Build Settings dialog box. If you want to replace the UDAs of a member, select "Allow property changes" and Update UDAs in the Dimension Build Settings dialog box.

Creating Attribute Dimensions with Rules Files

1. Add the attribute dimension in association with its base dimension.
2. Build the attribute dimension hierarchy.
3. Associate attributes with members of the base dimension.

ORACLE

Creating Attribute Dimensions with Rules Files

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 in rules files involves the following sequential operations:

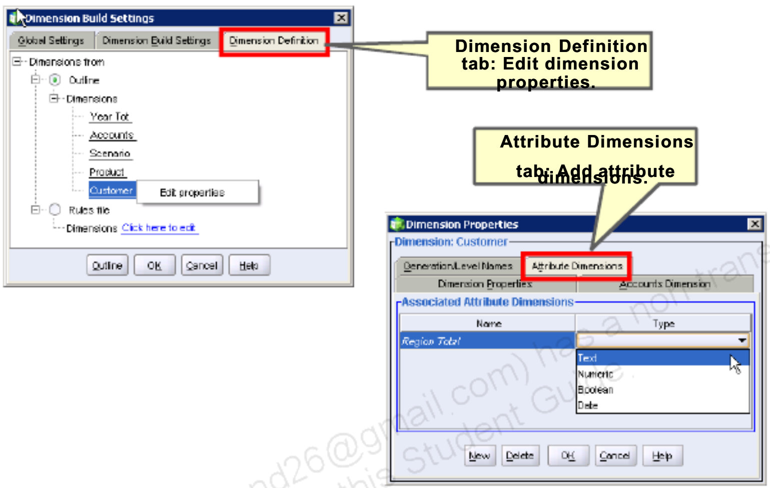
7. If the attribute dimension does not exist, you must create it and associate it with its base dimension.
8. You must build the attribute dimension hierarchy.
9. You must associate members of the attribute dimension with members of the base dimension.

NOTE: Rules files cannot be used to create or modify varying attribute associations. However, you can use Essbase Studio to create or modify many varying attribute associations at a time. See Lesson 18, "Analyzing Varying Attributes."

You can use either of the following approaches to perform these operations:

- Build the attribute dimension and perform the associations in one rules file. If the base dimension exists, you can build an attribute dimension and associate its attributes with the members of the base dimension in one step. You need to define only the attribute associations in the rules file.
- Build the attribute dimension with one rules file, and then perform the associations in another rules file. If the base dimension exists, you can build an attribute dimension and associate its attributes with the members of the base dimension in separate steps. Build the attribute dimension as you build a standard dimension, and then associate attribute members with members of the base dimension. You must use this approach when you build multilevel attribute dimensions.

Adding Attribute Dimensions with Rules Files



Adding Attribute Dimensions with Rules Files

When adding an attribute dimension, use the procedure for creating dimension build rules files described in"Creating Dimension Build Rules Files"in Lesson 6, "Creating Basic Dimension Build Rules Files."During the procedure, replace the procedure for adding a standard dimension in step 7 with the following procedure:

To add attribute dimensions in rules files:

1. In the Dimension Build Settings dialog box, select the **Dimension Definition** tab.
2. Right-click the standard dimension to associate with the attribute dimension, and select **Edit properties**.

   The Dimension Properties dialog box is displayed.
3. Select the **Attribute Dimensions** tab.
4. In the Name text box, enter your attribute dimension name.
5. In the Type drop-down list, select an attribute dimension type.
6. Click **OK**.

Assigning Attributes in Rules Files

Field properties for attribute members:

- Dimension: base dimen sion
- Field type: attribute dimension
- Field number: level or generation number of the base dimension members with which you are associating attributes

LEVEL0, Customer	Region Total0, Customer
O-IBM	East
O-Acer	South
O-Apple	West
O-AST	MidWest
…	…

Field definition for attribute members

ORACLE

Assigning Attributes in Rules Files

Whether you build the attribute dimension and associate the attribute members with the members of the base dimension in one step or in separate steps, you use the same method to define the field properties in your rules file.

Every record of the data source must include at least two columns, one for the member of the base dimension and one for the attribute value of the base dimension member. You must position the field for the member of the base dimension before the felds for the members of the attribute dimension. You can create attribute associations by using any of the three main dimension build methods.

If attribute dimensions exist in your outline, or if you defined attribute dimensions in the rules file, the list of field types includes the names of the attribute dimensions. To associate attributes with base dimension members, define the field properties for the attribute dimension members as follows:

- **Dimension:** base dimension
- **Field type:** attribute dimension
- **Number:** level or generation number of the base dimension members with which you are associating attributes

Prohibiting the Addition of Attribute Members

By default, Essbase adds members to the attribute dimension as it associates attributes with base dimension members. If you are associating members from a multilevel attribute dimension, or if your attribute dimension is already built, you must not allow Essbase to add members to the attribute dimension.

To prevent the addition of attribute members:

1. Open the **Dimension Build Settings** dialog box.
2. In the dimension list, double-click the base dimension.
3. in the Attribute Members area, select **Do not create members**
4. Click **OK**.

Updating Attribute Associations

You can update attribute associations, but you must first allow association changes.

To allow attribute association changes:

1. Open the **Dimension Build Settings** dialog box.
2. In the dimension list, double-click the base dimension.
3. In the Attribute Members area, select **Allow association changes**
4. Click **OK**.

Summary

In this lesson, you should have learned to:

- Describe advanced dimension build rules files
- Create shared members
- Arrange, alter, and ignore fields
- Create used-defined attributes
- Create attribute dimensions

LESSON 8

Loading Data

Objectives

At the end of this lesson, you should be able to:

- Describe loading data in Essbase
- Prep Data Prep Editor
- Define field properties
- Reference missing dimensions
- Set data load options
- Load data to Essbase
- Select and reject data records
- Capture new members during data load

Data Load Overview

Load data:
- Without rules
- With rules

Data Load Overview

When your outline is complete, you can start to load data (that is, numbers) into your database for analysis. A data source for loading data must contain not only the values that you want to load but also identification of the location to which Essbase should load the values: a data address that includes one member from every standard dimension in your outline.

Although you can load some data sources without rules files, most data sources typically do require rules to inform Essbase where data should be stored. Essbase supports the following types of data sources for loading data:

- Text files (flat files) from text backups or external sources
- SQL data sources
- Microsoft Excel files
- Error files generated from a previous dimension build

- Oracle BI Enterprise Edition (OBIEE)
- Essbase export files (Export files can be loaded without a rules file.)
- Spreadsheet audit log files
- User input submitted from Smart View, Spreadsheet Add-in, or Planning forms

Free-Form Data Sources

Simple free-form data source:

```
"O-IBM" "Lightbolt 365 A" "Current Year" Jan   Units 105
"O-IBM" "Lightbolt 365 S" "Current Year" Jan   Units 85
"O-IBM" "Thunderball 540 A" "Current Year" Jan  Units 57
```

Formatted free-form data source:

```
Units "O-IBM"
                                     Jan  Feb  Mar

 "Current Year" "Lightbolt 365 A"    105  101   98
                "Lightbolt 365 S"     85   72   82
                "Thunderball 540 A"   57   53   16
```


Free-Form Data Sources

If a data source contains all information required to load its data values into the database and is formatted properly, you can load the data source without using a rules file. This kind of load is called a *free-form data load*.

To load a data value successfully, Essbase must encounter one member from each dimension before encountering the data value. If Essbase encounters a data value before encountering a member of each dimension, it stops loading the data source, resulting in a partial load of the data values.

To map perfectly to a database, a data source must contain only the following:

- One or more valid members from each dimension. Member names must be enclosed in quotation marks if they contain spaces, numeric characters (0-9), dashes, plus signs, or ampersands (&).
- One or more valid data values.
- Spaces as delimiters.

System and User-Generated Free-Form Data Sources

The most common types of free-form data sources in a production environment are system- and user-generated:

- **Essbase data export**—A file containing data exported from Essbase does not require a rules file to import.
- **Spreadsheet-based data entry**—Data submitted by users from Smart View or Spreadsheet Add-in does not require a rules file.

Simple Free-Form Data Sources

The simplest way to format records in free-form data sources is for each record to include a member from each dimension and a data field, as in the first example on the slide. Given a database with five dimensions—Scenario, Time, Customer, Product, and Accounts— the first record in the example includes one member from each dimension (for example, O-IBM, Lightbolt 365 A, Current Year, Jan, and Units). The last field in the record is the data that relates to the unique combination of members.

Formatted Free-Form Data Sources

Essbase also supports more complex free-form data source formatting, as illustrated by the second example on the slide.

1. The Accounts and Customer dimensions are represented by members Units and O-IBM, respectively, in the first line of the file. Essbase reads these members as file headers, and applies these headers to every data point in the file.
2. Essbase reads the column headers Jan, Feb, and Mar (Time dimension) and applies these headers to the data in their respective columns.
3. Each data record contains references to the remaining two dimensions, Scenario and Product, in addition to three data points relating to the three columns of time periods.

For example, the first data record contains the members Current Year and Lightbolt 365 A and the data points 105, 101, and 98. Essbase determines the complete address for the value 105 as a combination of the file header, column header, and row headers (Units, O-IBM, Jan, Current Year, Lightbolt 365 A).

NOTE: For a comprehensive discussion about loading free-form data sources, see "Data Sources That Do Not Need a Rules File" in the *Oracle Essbase Database Administrator's Guide*.

Data Sources That Require Rules Files

Use rules files when you have any of these requirements:

- Load data from a SQL data source
- Build dimensions while loading data
- Change the data
- Change member names

Data Sources That Require Rules Files

If your data source does not meet free-form loading requirements, you need a rules file to load it. Rules files enable you to compensate for data sources that do not map perfectly to your database.

Use rules files when you have any of the following requirements:

- **Load data from a SQL data source.** Using the SQL interface to access external relational data sources requires the use of rules files to store your SQL queries and identify your data source.
- **Build dimensions while loading data.** Building dimensions from external sources requires the use of rules files.
- **Change the data.** You can use rules files to scale data, change the sign on data, or ignore unnecessary fields.
- **Change member names.** You can use rules files to add prefixes or suffixes to member names, reference missing dimensions, and manipulate fields.

Data Load Errors

During free-form data loads, when Essbase encounters an incorrect member field, the data load is interrupted and Essbase does not write to an error file. However, during data loads using rules files, when Essbase encounters an incorrect member field, an error message is logged in an exception file (`dataload.err`) along with the record that caused the error. Essbase then continues processing all the records in the data source after the incorrect record.

Creating Data Load Rules Files

General Setup	1. Open a sample data source. 2. Set data source properties. 3. Hide raw data (optional). 4. Set the view to data load fields. 5. Associate the rule with a database outline. 6. If necessary, format the file.
Actions	7. Define field properties. 8. If necessary, define a data load header for missing dimensions. 9. Select data load values options.
Completion	10. Validate the rules file. 11. Save the rules file. 12. Load data.

ORACLE

Creating Data Load Rules Files

Before beginning the procedure outlined on the slide and in the following table, you must open Data Prep Editor. You can use the following table for a quick reference when creating data load rules files. The table lists each step in the procedure with its corresponding menu and menu command:

Procedure Step	Menu	Command
1. Open a sample data source.	File	**For flat files, Excel files, or error files:** Open data file **For SQL data sources:** Open SQL
2. Set data source properties.	Options	Datasourceproperties
3. Hiderawdata(optional).	View	Rawdata(toggle)
4. Set the view to data load fields.	View	Dataloadfields(toggle)

Procedure Step	Menu	Command
5. Associate the rule with a database outline.	Options	Associate outline
6. If necessary, format the file.	Field	Various
7. Define field properties.	Field	Properties (Data Load Properties tab)
8. If necessary, define a data load header for missing dimensions.	Options	Data load settings (Header Definition tab)
9. Select data load values options.	Options	Data load settings (Data Load Values tab)
10. Validate the rules file.	Options	Validate
11. Save the rules file.	File	Save
12. Load data.	Actions (Database must be selected in the navigation panel.)	Load data for *"DBName"*

TIP: Do not use the Open toolbar button to open a data source in your rules file; for Data Prep Editor to read a data source, you must select "Open data file" from the File menu.

Prepping Data Prep Editor (Steps 1–6)

Steps 1 through 6 ensure that you do the following:

- Provide yourself with samples of dat a on which to base your rules.
- Define correct file de limiters.
- Identify and ignore header records.
- Set the view in Data Prep Editor to th e correct mode for loading data.
- Create an association between the rules file and the database outline that you want to update.
- Format data, if ne cessary.

Prepping Data Prep Editor (Steps 1–6)

As in the procedure for creating dimension build rules files, the general setup steps in the procedure for creating data load rules files prepare Data Prep Editor for building the rules file. You do not have to execute these steps in any particular order, but if you follow a standard setup routine, you are less likely to forget an important step. Therefore, it is recommended that you complete the steps in the same order each time you create a rules file. For a detailed description of the general setup steps, see "Prepping Data Prep Editor (Steps 1–6)" in Lesson 6, "Creating Basic Dimension Build Rules Files."

Step 1: Opening a Sample Data Source

After opening Data Prep Editor, you can open data sources such as text files, spreadsheet files, and SQL data sources. The data source provides a frame of reference for building the rules file; it is displayed in Data Prep Editor so that you can see what needs to be changed and how to create your mappings. The sample data source that you use to build the rule should have the same format as the actual data source that you use to update your members.

Step 2: Setting Data Source Properties
This step comprises two substeps: selecting a file delimiter for the data source and identifying and ignoring header records in the data source.

Step 3: Hiding Raw Data (Optional)
By default, the Data Prep Editor window is organized into two frames that enable you to compare the rules file results to the srcinal data source. However, because many steps in the rules file procedure involve opening dialog boxes on top of the rules file, hiding the source data can give you a better view of the rule information as you complete the procedure.

You can toggle the raw data view on and off from the View menu.

Step 4: Setting the View to Data Load Fields
When developing a rules file, you work in one of the following view modes: dimension build fields or data load fields.

Although you may not see a difference in the rule when you change the view mode, changing view mode has a pronounced effect on later steps in the procedure, namely assigning field properties and validating your rule. The default view in Data Prep Editor is "Data load fields."

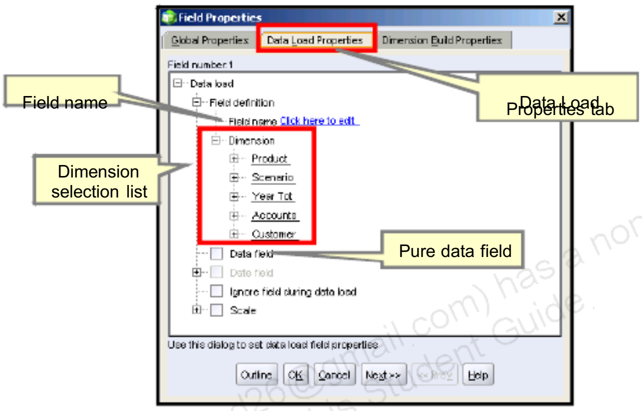
Step 5: Associating the Rule with a Database Outline
You must associate your rules file with the database outline for which you are creating the rule. Similar to step 4, this step provides no visual cues after completion, but, if you do not associate your rule with an outline, you cannot successfully construct or validate the rules file for errors.

You can associate an outline with your rules file from the Options menu.

Step 6: Formatting the File (If Necessary)
If your data source columns are not in the correct order for building the dimension or if you must format the columns in other ways, you can use Field menu commands to format the data source columns.

Defining Field Properties (Step 7)

For each field, define a field name.



Defining Field Properties (Step 7)

Data load rules files, like rules files for dimension building, rely on the organization of fields in the data source. To enable Essbase to process your data source, you must define each field in the rules file by selecting the field name.

Data sources can contain member name fields, member data fields, and pure data fields. Consider a data source consisting of scenario names in the first field, customer names in the second field, product names in the third field, account names in the fourth field, and data values in fields 5 through 7 (representing January through March).

The data source contains both member name and member data fields, as shown in the following figure:

	Field1	Field2	Field3	Field4	Field5	Field6	Field7
1	Scenario	Customer	Product	Accounts	Jan	Feb	Mar
2	Prior Year	O-IBM	LIGHTBOLT 365 A	Units	137	207	111
3	Prior Year	O-IBM	LIGHTBOLT 365 A	List Price	174	174	174
4	Prior Year	O-IBM	LIGHTBOLT 365 A	Discount %	0.035	0.052	0.035
5	Prior Year	O-IBM	LIGHTBOLT 365 A	Labor/Unit	26.1	26.1	26.1
6	Prior Year	O-IBM	LIGHTBOLT 365 A	Matl/Unit	62.64	62.64	62.64
7	Prior Year	O-IBM	LIGHTBOLT 365 A	Overhead R...	1.044	1.044	1.044
8	Prior Year	O-IBM	LIGHTBOLT 540 S	Units	114	115	181
9	Prior Year	O-IBM	LIGHTBOLT 540 S	List Price	217.5	217.5	217.5
10	Prior Year	O-IBM	LIGHTBOLT 540 S	Discount %	0.035	0.052	0.052
11	Prior Year	O-IBM	LIGHTBOLT 540 S	Labor/Unit	34.8	34.8	34.8

- **Member name fields**—These fields contain member names. In the Data Load Properties dialog box, you must identify member name fields with their related dimension name. In the preceding example, the first four fields are member name fields and must be identified by the relevant dimension names.
- **Member data fields**—These fields contain numeric data values that apply to a specific member, usually identified in the header row. In the Data Load Properties dialog box, you must identify each member data field with a single member from a dimension. Essbase determines the member intersection for a data value in a member data field as a combination of the member names in the record to the left of the data value and the member name in the data value's field definition. In the preceding example, fields 5 through 7 are member data fields and are identified by members of the Year Tot dimension (Jan, Feb, and Mar).

- **Pure data fields**—These fields also contain numeric data values, but the member intersection for the data value is completely defined by the member name fields to the left of the data value. Consider a data source consisting of scenario names in the first field, customer names in the second field, product names in the third field, time periods in the fourth field, account names in the fifth field, and data points in the sixth field. The data source contains member name fields and a pure data field (Field 6), as shown in the following figure:

	Field1	Field2	Field3	Field4	Field5	Field6
1	Scenario	Customers	Products	Time Periods	Accounts	Data
2	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Units	150.1
3	Budget	O-IBM	LIGHTBOLT 365 A	Jan	List Price	200
4	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Discount %	0.07
5	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Labor/Unit	25
6	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Matl/Unit	52.5
7	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Overhead R...	1.2
8	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Material Vari...	1,140.85
9	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Labor Varian...	1,953.51
10	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Overhead V...	1,570.62
11	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Obsolete Ch...	275.39
12	Budget	O-IBM	LIGHTBOLT 365 A	Jan	Gross Margi...	0.25
13	Budget	O-IBM	LIGHTBOLT 365 A	Feb	Units	226.1
14	Budget	O-IBM	LIGHTBOLT 365 A	Feb	List Price	200
15	Budget	O-IBM	LIGHTBOLT 365 A	Feb	Discount %	0.07

TIP: A data source can contain member data fields or pure data fields, but not both.

To define field properties for unique member name or data fields:

1. Open the Field Properties dialog box.
2. Select the **Data Load Properties** tab.
3. Depending on the field type, do one of the following:
   - For a member name field, in the Field Name text box, enter the related dimension name.
   - For a member data field, in the Field Name text box, enter the related member name.

TIP: Double-clicking a member in the dimension selection list adds the member to the Field Name text box.

   - For a pure data field, select **Data field**.

Defining Data Load Headers (Step 8)

If necessary, define a da ta load header for missing dimensions.

	Field1	Field2	Field3	Field4	Field5
1	Customers	Products	Time Periods	Accounts	Data
2	O-IBM	LIGHTBOLT 365 A	Jan	Units	150.1
3	O-IBM	LIGHTBOLT 365 A	Jan	List Price	200
4	O-IBM	LIGHTBOLT 365 A	Jan	Discount %	0.07
5	O-IBM	LIGHTBOLT 365 A	Jan	Labor/Unit	25
6	O-IBM	LIGHTBOLT 365 A	Jan	Matl/Unit	52.5
7	O-IBM	LIGHTBOLT 365 A	Jan	Overhead Rate	1.2
8	O-IBM	LIGHTBOLT 365 A	Jan	Material Variances	1,140.85
9	O-IBM	LIGHTBOLT 365 A	Jan	Labor Variances	1,953.51

Missing Scenario dimension

ORACLE

Defining Data Load Headers (Step 8)

To load data, Essbase requires a reference to every dimension in your database. Data sources may not contain all dimensions necessary to load data values. If dimension information is missing from your data source, you can define hidden data load headers in the rules file to supply the missing information.

Essbase determines the intersection for the data value as a combination of the member name fields in the record to the left of the member data field, the member names in the field definition, and the member names in the data load header. For the example on the slide, the first data value corresponds to the intersection of O-IBM, Lightbolt 365 A, List Price, Jan, and Prior Year. Because Prior Year is not defined in the data source, it must be defined in the data load header.

To define data load headers:

1. From the Options menu, select **Data load settings**.

   The Data Load Settings dialog box is displayed.

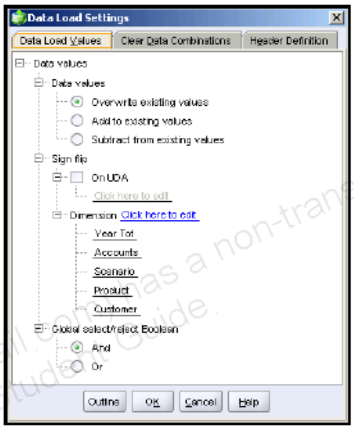
2. Select the **Header Definition** tab.
3. For header name, enter the member name for the data load header.

TIP: You can enter names from multiple missing dimensions. Separate multiple names with commas.

Setting Data Load Values Options (Step 9)

Data load values options:

- Overwrite, add to, or subtract from existing data values
- Change the sign of data upon loading



ORACLE

Setting Data Load Values Options (Step 9)

Data load rules files provide more ways to process data than free-form loads provide. You can choose how you want a data load to interact with values stored in your database and set conditions to change the sign of data as it is loaded.

Interacting with Existing Data

By default, Essbase overwrites existing values in a database with new values from the data source. Certain types of data sources may require different processing. For example:

- Data sources that contain records that duplicate the intersections of other records.

  For example, two salespeople record sales transactions of Lightbolt 365 A hard drives to IBM in the month of January. The data source contains each transaction in a separate record. However, in the Bigcorp Sales database, data is not tracked by salesperson. When the salesperson identifier is removed from the records, they become references to the same intersection in Essbase, and the final record overwrites the preceding ones unless they are aggregated during the data load.

- Data sources that contain a piece of an iterative, cumulative data load.

  In the following example, the data source contains multiple daily records for one customer, product, scenario, and account intersection. Because the Date field must map to a month, rather than a day, in the Bigcorp Sales database, you must aggregate all common records to provide a monthly total for each intersection.

Scenario	Item Number	CustID	Date	Units
Current Year	LIGHTBOLT 365 A	IBM	01/01/06	158
Current Year	LIGHTBOLT 365 A	IBM	01/02/06	238
Current Year	LIGHTBOLT 365 A	IBM	01/03/06	128
Current Year	LIGHTBOLT 365 A	IBM	01/04/06	128
Current Year	LIGHTBOLT 365 A	IBM	01/05/06	131
Current Year	LIGHTBOLT 365 A	IBM	01/06/06	132
Current Year	LIGHTBOLT 365 A	IBM	01/07/06	208
Current Year	LIGHTBOLT 365 A	IBM	01/08/06	208
Current Year	LIGHTBOLT 365 A	IBM	01/09/06	228
Current Year	LIGHTBOLT 365 A	IBM	01/10/06	218
Current Year	LIGHTBOLT 365 A	IBM	01/11/06	128
Current Year	LIGHTBOLT 365 A	IBM	01/12/06	128
...	...	...	...	...

Rules files provide the option to overwrite, add to existing values, or subtract from existing values on load. You change the load values options on the Data Load Values tab of the Data Load Settings dialog box.

NOTE: Using the add or subtract options makes it more difficult to recover from a data load interruption, although Essbase lists the number of the last row committed to the database in the application event log.

Changing the Sign of Data upon Loading

If, during the design phase of your implementation, you chose not to keep credit and debit account balances (such as negative balances for revenues or assets), you may need to reverse the sign on the data that you load into your database. Sign reversal is a two-step process:

1. Assign a UDA to all members (usually accounts) whose data you want to reverse on load.
2. When creating a rule for loading data, open the Data Load Settings dialog box on the Data Load Values tab and select the "Sign flip" option. You must also specify the UDA name and the dimension in which the UDA is assigned.

Validating Data Load Rules Files (Step 10)

Common validation error:

"There is an unknown member or no member in the field name."

ORACLE

Validating Data Load Rules Files (Step 10)

You must validate your rules file to ensure that the members and dimensions in the rules file map to the associated outline. While a correct validation does not ensure that the data source loads properly, invalid rules files typically result in an incorrect load.

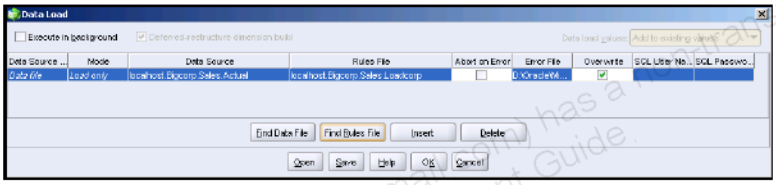
The most common data load rules file validation error is "There is an unknown member or no member in the field name." This error indicates that no header is defined for the field or that the field needs to be ignored.

NOTE: During rules file validation, Essbase does not check whether all dimensions are represented in the data source or the load headers. Therefore, when you validate your rule, do your own dimension check.

Completing Data Load Rules Files (Steps 11 and 12)

Step 11: Save the rules file:
- Save to Analytic Server
- Save to file system

Step 12: Load data:
- Manually in Administration Console
- Automatically with MaxL



ORACLE

Completing Data Load Rules Files (Steps 11 and 12)

The data load rules file creation procedure ends with saving the rules file, either to an Essbase server or to a file system location, and finally loading data by using one of two methods.

Step 11: Saving Rules Files

You can reuse a single rules file with multiple data sources and Essbase databases. By selecting the appropriate tab in the Save dialog box, you can choose to save your rule to a location on an Essbase server (in a database or application directory) or to a location in your file system. Essbase saves rules files with a RUL extension.

Step 12: Loading Data

After your data load rules file is complete, you can use it to load data into your database.

You can use either of the following methods to load data with rules files:

- Load manually by executing a data load in Administration Services Console

- Load automatically with a MaxL script

The first method provides an option to save the list of data sources and associated rules files as an XML file, so you can reuse the list in the future. If you save the XML file to a shared network location, other users can open the list that you saved.

To load data in Administration Services Console:

1. In Administration Services Console, select the database node.
2. From the Actions menu, select **Load data to "*DBName*"**.

   The Data Load dialog box is displayed.
3. From the Data Source drop-down list, select a data source type.
4. From the Mode drop-down list, select one of the following load modes:
   - **Load only** processes only data load rules from the rules file.
   - **Build only** processes only dimension build rules from the rules file.
   - **Both** processes both dimension build and data load rules from the rules file, in that order.
5. Click **Find Data File** to select a data source.
6. Click **Find Rules File** to select a rules file.
7. **Optional:** If you want Essbase to stop processing the load if it encounters errors, select **Abort on Error**.
8. Specify an error file location.
9. **Optional:** If you want Essbase to overwrite previous errors with new errors, select **Overwrite**.
10. If you selected a SQL data source type, enter the SQL user name and password.
11. Click **OK**.

Selecting and Rejecting Records

Load a subset of data from a data source.

Scenario	CustID	ItemNumber	Period	Units
CurrentYear	Apple-OEM	Lightbolt365A	Jan	158
CurrentYear	Apple-OEM	Lightbolt540S	Jan	238
CurrentYear	Apple-OEM	Thunderball540S	Jan	128
PriorYear	Apple-OEM	Lightbolt365A	Jan	128
PriorYear	Apple-OEM	Lightbolt540S	Jan	131
PriorYear	Apple-OEM	Thunderball540S	Jan	132
PriorYear	Apple-OEM	Roadranger170A	Jan	208
Budget	Apple-OEM	Lightbolt365A	Jan	208
Budget	Apple-OEM	Lightbolt540S	Jan	228
Budget	Apple-OEM	Thunderball540S	Jan	218
Budget	Apple-OEM	Roadranger170A	Jan	128
…	…	…	…	…

Selecting and Rejecting Records

By default, Essbase accepts all records in a data source. However, you can perform operations at a record level to select or reject records by using specific string or numeric criteria.

Selecting Records

You can specify which records Essbase loads into the database by setting selection criteria. Selection criteria are string and number conditions that must be met by one or more fields within a record before Essbase loads the record. If fields in the record do not meet the selection criteria, Essbase does not load the record. For the example on the slide, to load only Current Year data, create a selection criterion to load only records in which the first field is Current Year.

Rejecting Records
You can specify which records Essbase ignores by setting rejection criteria. Rejection criteria are string and number conditions that, when met by one or more fields within a record, cause Essbase to reject the record. If fields in the record meet the rejection criteria, Essbase does not load the record. For the example on the slide, to reject Budget data and load only Current Year and Prior Year, create a rejection criterion to reject records in which the first field is Budget.

Combining Multiple Select and Reject Criteria
When you define select and reject criteria on multiple fields, you can specify in the Data Load Values dialog box how Essbase combines the rules across fields; that is, whether the criteria are connected logically with AND or with OR. If you select AND as your Boolean operator, the fields must match all criteria. If you select OR as your Boolean operator, the fields must match only one of the criteria. AND is the default global Boolean setting. For the example on the slide, to reject only Budget data for the product Roadranger 170 A, select AND as your Boolean operator, and create the following rejection criteria:

- Field 1: Reject Budget.
- Field 3: Reject Roadranger 170 A.

NOTE: If both selection and rejection criteria apply to a record (that is, you try to select and reject the same record), Essbase rejects the record.

Capturing New Members

Scenario	Customer	Product	Account	Jan
CurrentYear	JC'sHardware	Lightbolt365A	Units	200
CurrentYear	Apple-Retail	Lightbolt365A	Units	105
…	…	…	…	…

- Without dimension build ru les:

```
\\ Member JC's Hardware Not Found In Database
Current Year  JC's Hardware   Lightbolt 365 A Units    200
```

- With dimension build rules:

ORACLE

Capturing New Members

After you deploy your database, you can expect dimensions to change. When you add products, customers, or accounts to your transaction systems, you may also need to add them to your Essbase outline. If the dimension build and data load processes are out of sync, you see errors during your data loads as Essbase tries to load values to members that do not exist.

For managing the flow of data, you have many options, including the following:

- Allow data loads to generate error records, which identify new members. Add the identified members manually or with a dimension build, and then reload the records in the error file. This option is best for slow-changing dimensions and for environments in which corporate processes or ownership issues do not allow other options.
- Incorporate dimension build rules and data load rules into one rules file. This option is good for frequently updated dimensions.

Allowing Data Loads to Generate Error Records

The slide shows a data source containing a new customer called JC's Hardware. If no dimension build information exists in the rules file when this data source is loaded, Essbase generates an error file containing all the records for JC's Hardware, as shown in the second example on the slide. You can institute the following process for handling new members in data files:

1. Data is loaded, and records for new members are written to error files.
2. Error files are examined manually to determine which members must be added to the database.
3. Members are added manually or using rules files.
4. Error files are loaded to the database using the same rules file as the srcinal data load to ensure data integrity.

Combining Dimension Builds and Data Loads

If a data source contains new members andtheir ancestors, you can use one of the three main build methods (generation, level, or parent-child).

However, if a data source contains new members and does not specify the ancestors of the new members, Essbase must decide where in the outline to add the members. The following table lists three special build methods in rules files, and provides a description of each method:

Method	Description
Add as sibling of member with matching string	Compares the new member to existing members in the outline by using string matching; for example, a new member called Lightbolt 810 S is matched to other Lightbolt products.
Add as sibling of lowest level	Assigns the new member to the lowest level in a hierarchy; particularly useful if you are loading to a flat dimension list.
Add as child of (selected member)	Assigns the new member as a child of a member in the outline; for example, capturing new customers as children of a member called "Unrecognized Customer."

When you combine any of the three special dimension build methods with data load rules in a single rules file, some extra considerations apply, as follows:

- If your data source contains a mix of new members and existing members, you do not want the build to apply to existing members. In Dimension Build Settings, select "Ignore conflicts" to avoid error messages for existing members.

- The methods have different requirements for dimension build properties; on the Dimension Build Properties tab of the Field Properties dialog box, select only a dimension. Do not select a field type or a field number.
- You define dimension build properties only for fields with new member names. For all other fields, select “Ignore during dimension build.”
- When you execute your data load, you must select Both for the load mode. This selection instructs Essbase to process the dimension build rules and add the new members before processing the data load rules in the rules file.

For the third example on the slide, a rules file has been created that includes dimension build rules to add new customers as children of “Unrecognized Customer.” The example represents the following process for handling new members in data files:

1. Data is loaded using a rules file with an automatic dimension build rules in addition to data load rules.
2. New members are added to the outline, and the outline is saved.
3. Data records for all members are loaded, and no error records are generated.
4. The outline must be examined regularly to ensure data integrity. In the example, JC’s Hardware must be moved under the Distributor parent for correct consolidation subtotals.

Summary

In this lesson, you should have learned to:

- Describe loading data in Essbase
- Prep Data Prep Editor
- Define field properties
- Reference missing dimensions
- Set data load options
- Load data to Essbase
- Select and reject data records
- Capture new members during data load

LESSON 9

Getting Started with Smart View

Objectives

At the end of this lesson, you should be able to:

- Describe the Smart View architecture and user interface
- Configure data sources
- Create ad hoc reports
- Set the point of view
- Associate data sources with worksheets
- Create free-form grids

Navigating Smart View

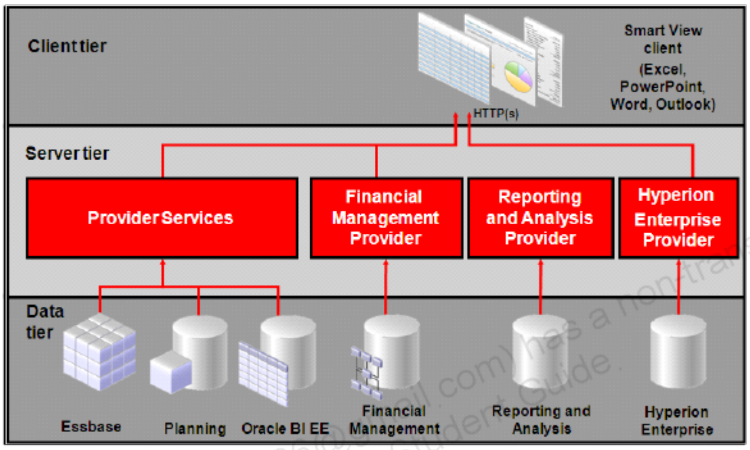
Navigating Smart View

Smart View provides a common Microsoft Office interface (Excel, Word, PowerPoint, Outlook) for Essbase, Financial Management, Planning, the Reporting and Analysis repository (Financial Reporting, Interactive Reporting, SQR Production Reporting, and Web Analysis), Hyperion Enterprise, and Oracle BI EE. The single interface enables you to leverage multiple products simultaneously. Smart View enables you to perform the following tasks:

- Use Excel to query data and create reports
- Import existing Reporting and Analysis content (reports and graphs) into Excel, Word, and PowerPoint
- Expose functions for Financial Management and Essbase content in Word and PowerPoint
- Access Financial Management and Planning forms and task lists

The lessons in this course concertrate on using Smart View to access Essbase data. For information about other data sources, see the *Oracle Smart View for Office User's Guide*.

Smart View Architecture



Smart View Architecture

Smart View incorporates powerful architectural features to handle a wide range of analytic applications across large, multiple-user environments.

The **client tier** includes a client component for Microsoft Office programs (Excel, PowerPoint, Word, and Outlook) that you must install on your personal computer before you can use Smart View to access data sources.

NOTE: Outlook supports functions only available in Financial Management and Planning. Essbase functions are not supported in Outlook.

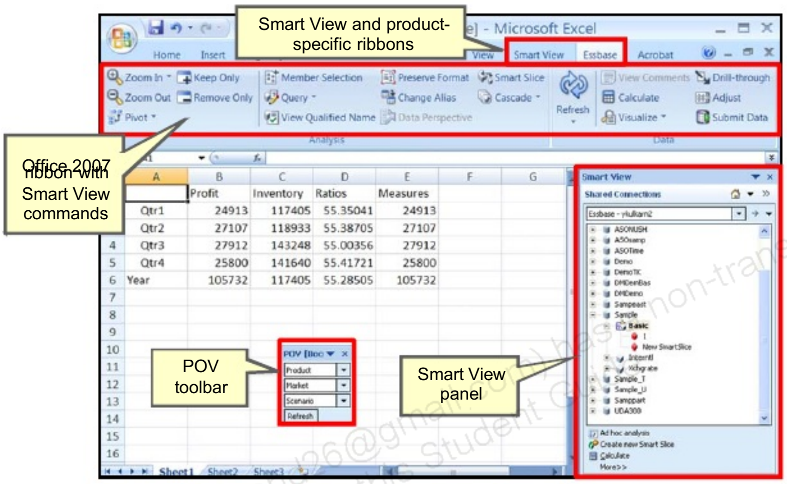
The **server tier** (**middle tier**) retrieves requested information and manages security, communication, and integration. It contains the following providers to access data from various data sources.

- Provider Services (a common provider to access data from Essbase, Planning, and Oracle BI EE)
- Financial Management Provider
- Reporting and Analysis Provider
- Hyperion Enterprise Provider

The **data tier** includes various data sources (Essbase, Financial Management, Planning, Oracle BI EE, Hyperion Enterprise, and Reporting and Analysis) from which Smart View can retrieve information.

NOTE: The Financial Management, Reporting and Analysis, and Hyperion Enterprise providers exist on the application tier and are integrated into the Financial Management, Reporting and Analysis, and Workspace installations. Provider Services exists on the Web tier and requires a separate installation from Essbase.

Smart View Excel User Interface



Smart View Excel User Interface

After installation, Smart View is automatically enabled for Excel, Word, PowerPoint, and Outlook (Outlook is used only for Planning and Financial Management). The Microsoft Office 2007 interface for Smart View includes the Smart View and provider-specific ribbons (for example, Essbase), the context-specific ribbon, the Smart View panel, and the point of view (POV) toolbar.

Smart View Ribbon

The Smart View ribbon enables you to set Smart View options and perform commands that are common for all data source providers.

Essbase Ribbon

The Essbase ribbon contains commands that enable you to view, navigate, and analyze Essbase data.

POV Toolbar

Dimensions that are not displayed in columns or rows of a data grid are displayed on the POV toolbar, which identifies a slice of the database for a grid. For default ad hoc grids, all but two database dimensions are displayed on the POV toolbar at the dimension level.

Smart View Panel

You use the Smart View panel to connect to your Smart View data sources and manage connections.

Enabling and Disabling Smart View

If you do not use Smart View with a Microsoft Office product, you can disable the functionality and enable it later.

To disable Smart View:

1. Launch the application.
2. On the Smart View ribbon, click **Help**, and then **About**.
3. Clear the **Enable Add-in** check box.
4. Click **OK**.

   The following message is displayed: "This change will take effect the next time you start an Office application."
5. Click **OK**.

   When you start the application, the Smart View tab is displayed, but it contains only the About option.

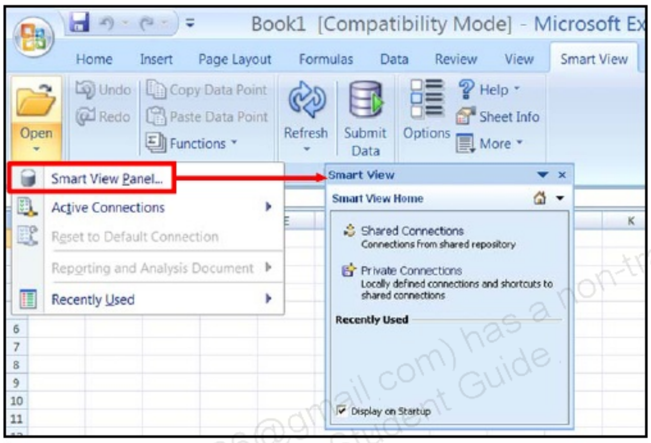
To enable Smart View:

1. Launch the application.
2. On the Smart View ribbon, click **Help**, and then **About**.
3. Select the **Enable Add-in** check box.
4. Click **OK**.

   The following message is displayed: "This change will take effect the next time you start an Office application."
5. Click **OK**.

   When you start the application, the Smart View ribbon contains all available menu commands.

Connecting to Data Sources



ORACLE

Connecting to Data Sources

You connect to data sources through shared connections or private connections on the Smart View panel. You can also select from a list of recently used connections on the Smart View panel home page to connect to a data source.

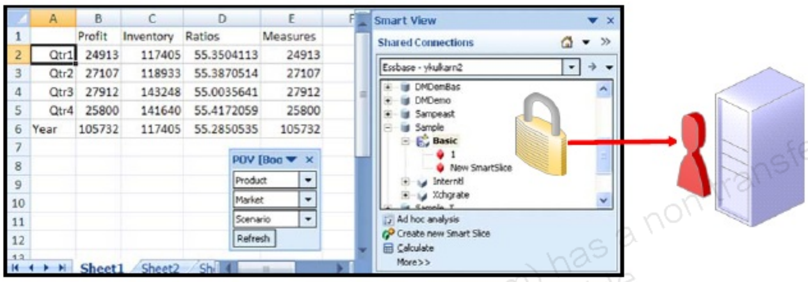
The Smart View panel also enables you to manage connections, create and manage smart slices and queries, open data forms, or initiate ad hoc analysis.

To open the Smart View panel:

1. On the Smart View ribbon, click **Open**.
2. In the drop-down list, select **Smart View Panel**.

 The Smart View panel is displayed.

Working with Shared Connections



ORACLE

Working with Shared Connections

If a data source provider is registered with Shared Services, you can access it through shared connections, which are:

- Stored in a central location
- Available to multiple users
- Created and maintained by administrators

Before you can work with shared connections, a Smart View administrator must specify the shared connections URL and add at least one server that hosts data sources to Smart View.

To specify the Shared Connections URL:

1. On the Smart View ribbon, click **Options**, and then **Advanced**.
2. In the Shared Connections URL section, enter the following URL:

```
http://servername:19000/workspace/SmartViewProviders
```

NOTE: URLs for all Smart View connections are case-sensitive.

3. Click **OK**.

To add servers:

1. On the Smart View panel home page, click **Shared Connections**.

   The Connect to Data Source dialog box is displayed.
2. Enter a user name and a password.
3. Click **Connect**.

   The Shared Connections view of the Smart View panel is displayed.
4. Click the drop-down arrow on the right, and select **Add new server**.

   The Check User Authorization dialog box is displayed.
5. Enter the following information:
   - Product—data source type (Essbase)
   - Product Server Name—name of the computer on which the data source server is located
   - User Name—user name to connect to the Essbase server
   - Password—password to connect to the Essbase server
6. Click **OK**.

   The server is available for selection from the drop-down list of servers. Users can access the server through Shared Connections, and can connect to all data sources on the server to which they have access.

To remove servers:

1. In Shared Connections, click the drop-down button on the right.
2. From the drop-down list, select **Remove server**.

   A dialog box asks you to confirm the deletion.
3. Click **Yes**.

   The Check User Authorization dialog box is displayed.
4. Enter the Essbase server user name and password, and click **OK**.

   The Essbase server is removed from Smart View.

NOTE: Only administrators can add or remove servers.

To connect to data sources:

1. In Shared Connections, select a server from the drop-down list.

   The Connect to Data Source dialog box is displayed.
2. Enter a user name and a password.
3. Click Connect.

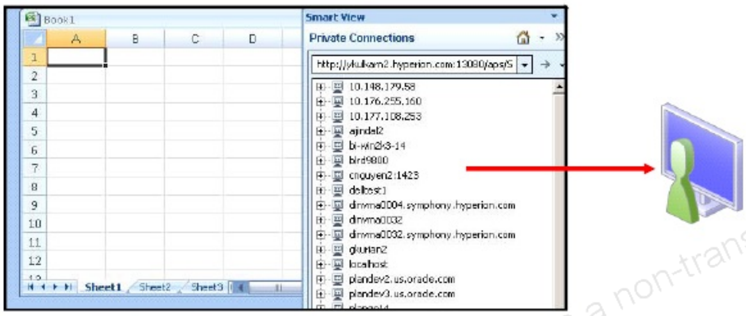
   An expandable tree list of available data sources is displayed.
4. Expand an application, and right-click a database.
5. Click **Connect**.

   A list of context-sensitive action items for the database is displayed in the Smart View panel.

NOTE: If you want to create an ad hoc report, you can connect to a data source by right-clicking it and selecting "Ad hoc analysis."

You can also set the default data source that will be automatically used when you retrieve data into an Excel worksheet.

Working with Private Connections



Working with Private Connections

You can create a private connection by adding a connection from Shared Connections to Private Connections, or by entering a direct URL for a provider.

The following table summarizes direct URLs for providers:

Provider	Default URL
Essbase	`http://servername:13080/aps/SmartView`
Planning	`http://servername:8300/HyperionPlanning/SmartView`
Financial Management	`http(s)://servername/hfmofficeprovider/hfmofficeprovider.aspx`
Reporting and Analysis	`http(s)://servername:19000/workspace/browse/listXML`
Hyperion Enterprise	`http://servername/heofficeprovider/heofficeprovider.aspx`

To open the list of private connections:

Perform one of the following actions:

- On the Smart View panel home page, click **Private Connections**
- In Shared Connections, click the drop-down arrow next to the home icon and select **Private Connections**

To create direct connections to data sources:

1. In Private Connections, click the drop-down arrow on the right.
2. Select **Create New connection**.

   The Add Connection - Provider Type/URL dialog box is displayed.
3. In the Provider drop-down list, select a provider type:
   - **Hyperion Provider**—for Financial Management, Hyperion Enterprise, Essbase, and Planning providers
   - **Hyperion Reporting and Analysis - System 9 Provider**—for Reporting and Analysis providers
4. In the Location drop-down list, select or enter the URL for the provider.

NOTE: The URL for Smart View Provider is case-sensitive.

---

5. Click **Next**.

   The Select Add Connection - Application/Cube dialog box is displayed.
6. Expand the **Servers/Clusters** node, and double-click the server name (for example, localhost).

   The list of available applications and databases is displayed.
7. Select a database, and click **Next**.

   The Add Connection Name/Description dialog box is displayed.
8. Enter a name and an optional description for the connection, and click **Finish**.

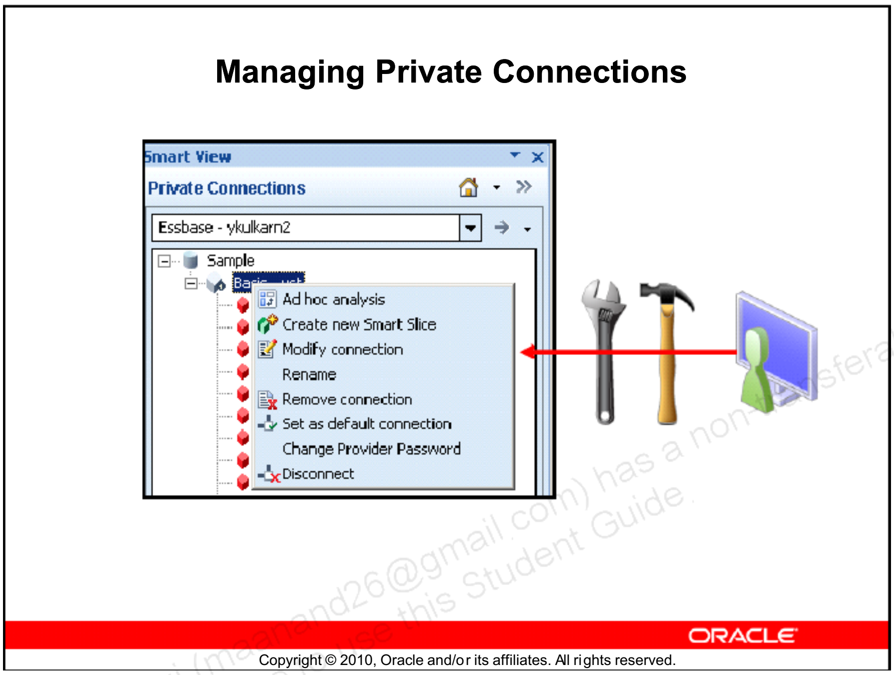
   The connection is displayed in Private Connections.

NOTE: For quick connectivity, you can also enter the provider URL in the server name field below Private Connections, and click the arrow button.

To add connections from Shared Connections to Private Connections:

1. In Shared Connections, right-click a database (for example, Bigcorp Sales).
2. Select **Add to Private connections**

   The database is available from Private Connections.



Managing Private Connections

You can edit a connection to change the information stored in it, or you can delete a connection from the list when you no longer need to connect to a particular data source.

You can also set the default data source that will be automatically used when you retrieve data into an Excel worksheet.

To delete or edit connections:

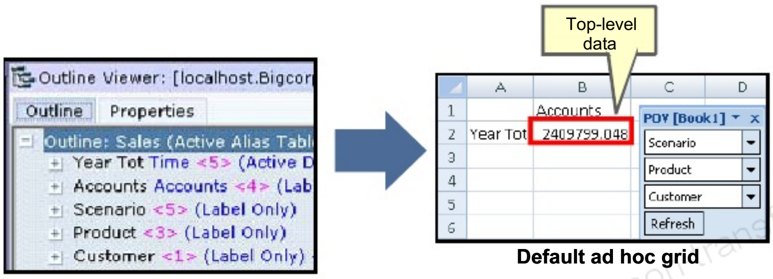
- Right-click the data source, and select **Modify connection** or **Remove connection**

To set the default data source:

1. Select a data source connection.
2. Right-click and select**Set As Default**.

   The data source name is displayed in bold text.

Creating Ad Hoc Grids



Database outline

Default ad hoc grid

Creating Ad Hoc Grids

Ad hoc grids are used to analyze data from a data source. An ad hoc report is a query to the database server based on the dimensions and members currently displayed in the worksheet. Ad hoc queries enable you to view data from data sources without creating spreadsheet functions.

After connecting to an Essbase server and opening a worksheet, you can initiate ad hoc reports against databases on the connected server. Ad hoc queries initiated at the database level on an empty worksheet return data from the top levels of each database dimension, positioning the dimensions in the following order:

1. First dimension in the database outline: default row dimension
2. Second dimension in the database outline: default column dimension
3. Remaining dimensions: default POV dimensions

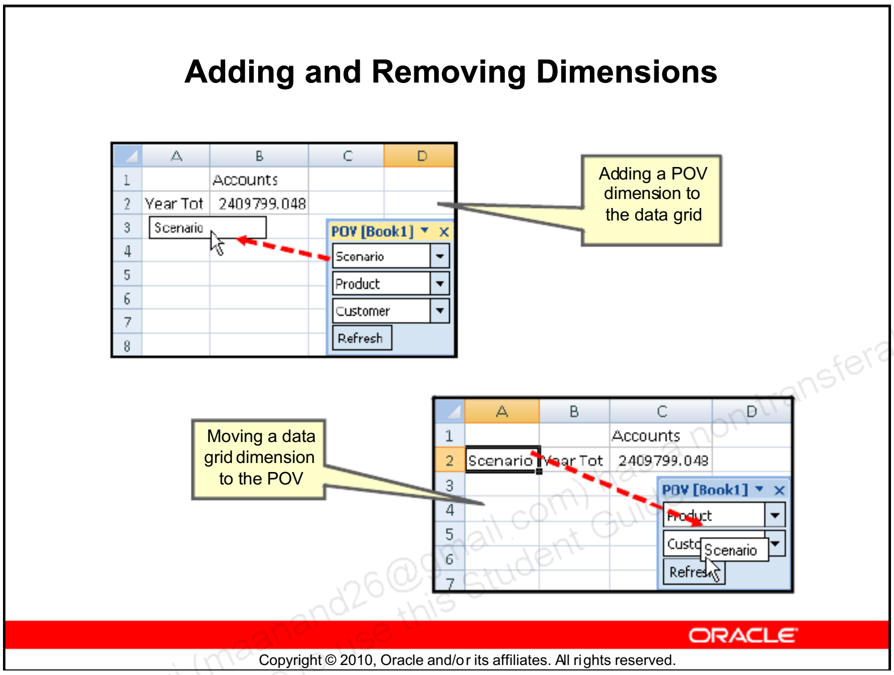
You can use the top-level data as a starting point to navigate, or drill down, into levels of detailed data. The example on the slide shows the Bigcorp Sales database outline, with the dimension order Year Tot, Accounts, Scenario, Product, and Customer. The default ad hoc grid for Bigcorp Sales is created with Year Tot in the rows, Accounts in the columns, and Scenario, Product, and Customer in the POV.

NOTE: You can initiate ad hoc reports against smart slices, which represent subsets of databases. Ad hoc grids based on smart slices are displayed with the layout defined in the smart slices and are not dependent on outline order. For a detailed discussion of smart slices, see "Creating Shared Database Perspectives" on page 10-13

To open default ad hoc grids:

Perform one of the following actions:

- On the Smart View panel, right-click a connection, and select **Ad hoc analysis**.
- Associate a connection with a worksheet, and double-click in an empty cell or click **Refresh** on the Smart View ribbon.



Adding and Removing Dimensions

After retrieving top-level data, you can modify the default grid layout by moving dimensions from the POV to the rows or columns of the grid or by moving dimensions from the rows or columns of the grid to the POV.

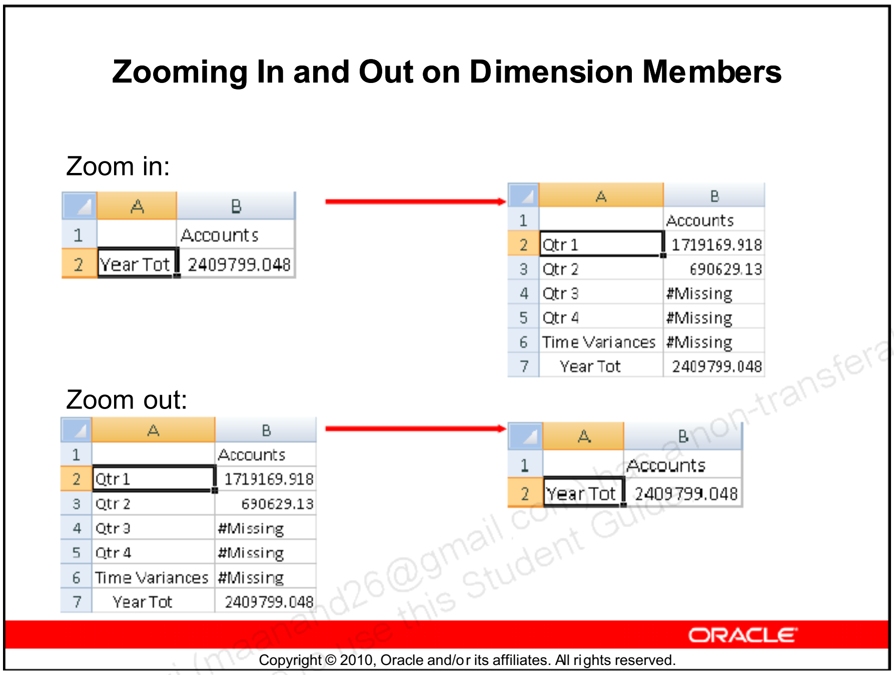
To move dimensions to the g rid:

- Left-click and drag a dimension from the POV to the grid.

To move dimensions t o the POV:

- Right-click and drag a member from the grid to the POV.

NOTE: You must have at least one row and one column dimension at all times.



Zooming In and Out on Dimension Members

Smart View provides two commands that support navigation operations:

- **Zoom In:** Drills down to display details. In the top example on the slide, applying the Zoom In command to the member Year Tot displays the members below Year Tot in the outline hierarchy: Qtr 1, Qtr 2, Qtr 3, Qtr 4, and Time Variances.
- **Zoom Out:** Drills up by collapsing the member tree. In the bottom example on the slide, applying the Zoom Out command to any child of Year Tot collapses the entire time hierarchy back up to the member Year Tot.

To zoom in on a member:

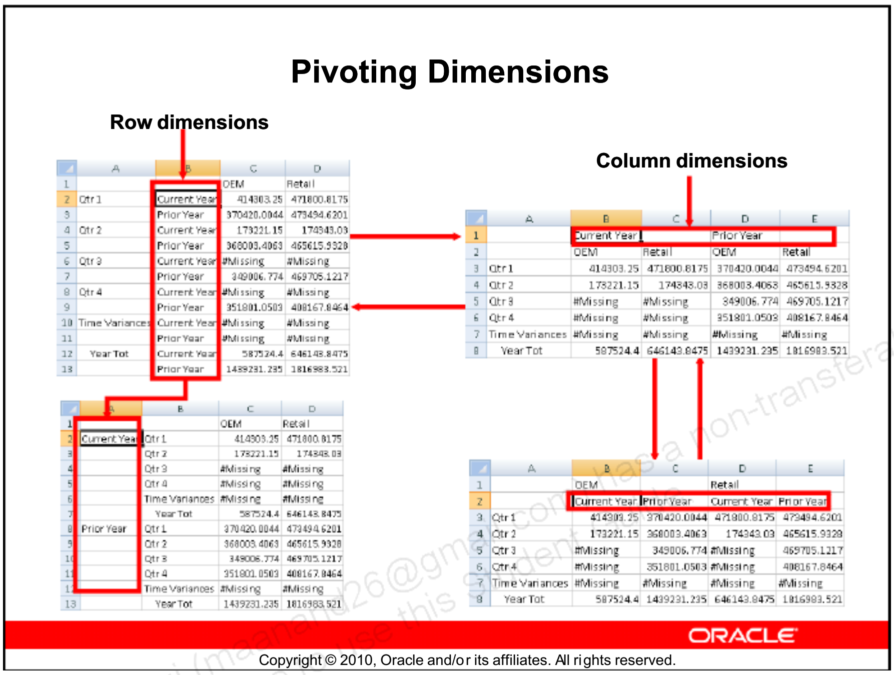
Perform one of the following actions:

- Select the member and then, on the Essbase ribbon, click **Zoom In**.
- Double-click the member.

To zoom out on a member:

Perform one of the following actions:

- Select the member and then, on the Essbase ribbon, click **Zoom Out**.
- Double right-click the member.



Pivoting Dimensions

You can change the orientation ofworksheet data by pivoting dimensions within a report

- Move row dimensions to columns
- Move column dimensions to rows
- Change the order of grouped column or row dimensions

TIP: The grid must always include at least one dimension on rows and one dimension on columns.

The example on the slide shows a three-dimension grid, with Year Tot dimension members as the outer row dimension, Scenario members as the inner row dimension, and Customer members as the column dimension. From this starting position, the Scenario dimension can be pivoted to any of the following positions:

- Outer row dimension
- Upper column dimension
- Lower column dimension

After pivoting the Scenario dimension to any of these positions, the data in the grid remains the same, but is reorganized in different ways depending on the chosen pivot operation.

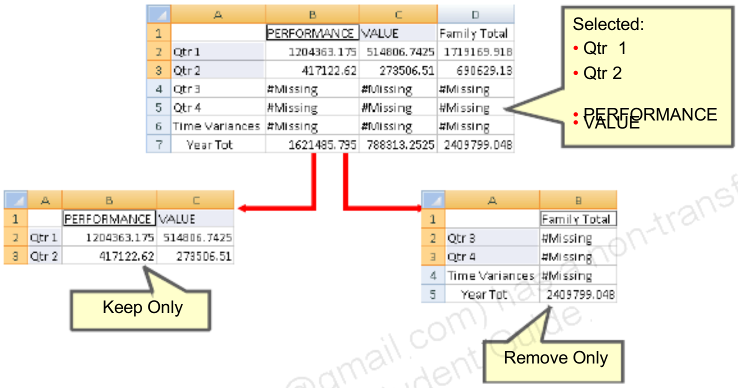
To pivot dimensions :

1. Select a dimension member.
2. On the Essbase ribbon, click **Pivot**.

To change the order of grouped column or row dimensions (special pivot):

- Right-click and drag a dimension member to the new position on the grid.

Keeping and Removing Dimension Members



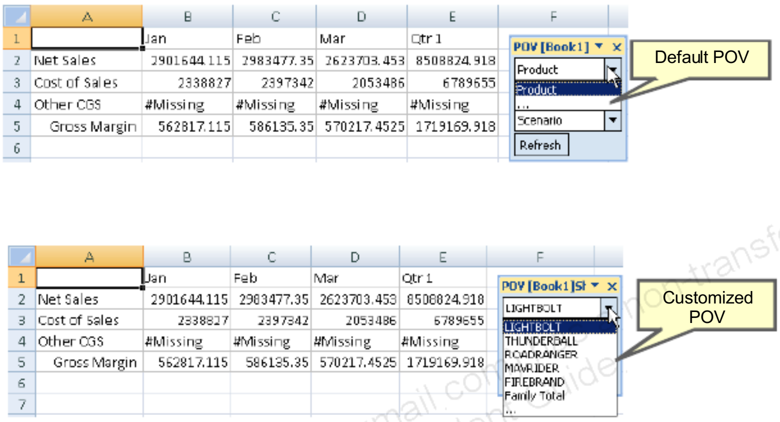
Keeping and Removing Dimension Members

To keep data or remove data from reports, you can use the Keep Only or Remove Only commands on the Essbase ribbon.

- The Keep Only command retains selected members and removes unselected members and their associated rows or columns of data.
- The Remove Only command removes selected members and their associated rows or columns of data and retains unselected members.

The example on the slide shows a report with quarterly time periods and Year Tot in the rows and the product families Performance and Value and Family Total in the columns. The row members Qtr 1 and Qtr 2 and the column members Performance and Value are selected. When you use the Keep Only command, only the selected members are retained in the report. By contrast, when you use the Remove Only command, only the unselected members are retained in the report (that is, Qtr 3, Qtr 4, Family Total, and so on).

Setting the Point of View



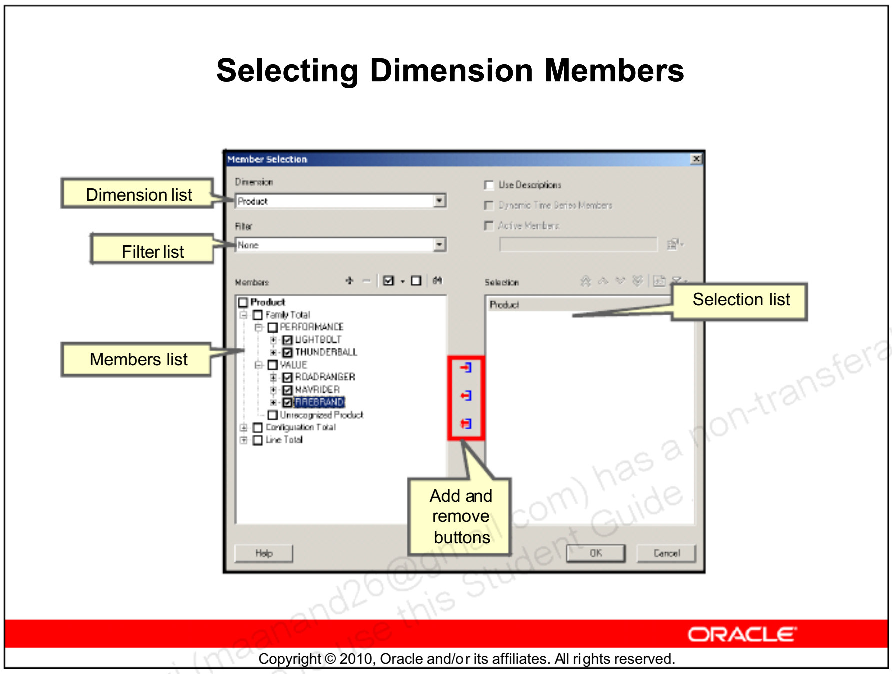
Setting the Point of View

The POV toolbar identifies a specific slice of the database for a grid. Although each dimension is displayed as a drop-down list, by default, only the top dimension member is available for selection. You can customize which members are displayed in the dimension drop-down lists on the POV toolbar.

For the examples on the slide, the top example shows a default POV, with only the member Product displayed in the Product dimension selection list. The bottom example shows a customized POV, with multiple product lines and the Family Total member displayed in the Product dimension selection list.

To change the POV:

1. From each POV dimension drop-down list, select a member.
2. On the POV toolbar, click **Refresh**.



Selecting Dimension Members

You can customize the list of members displayed in the POV drop-down lists, enabling you to focus on the members that you need for analysis.

You select members using the Member Selection dialog box:

- **Dimension list:** a list of database dimensions
- **Filter list:** a list of filters to fine-tune your member list
- **Members list:** a list of dimension members
- **Selection list:** a list of selected members
- **Add and remove buttons:** buttons to move members to and from the Selection list

To select members for POV drop-down lists:

1. From a POV dimension drop-down list, select the ellipsis.

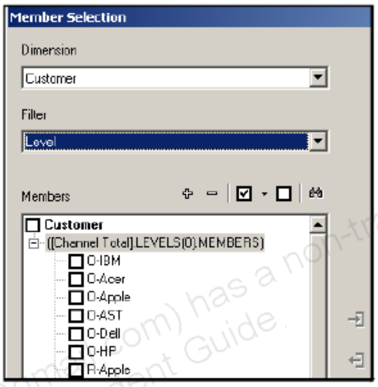
   The Member Selection dialog box is displayed.

2. **Optional:**Click  to remove the default member selection.
3. From the Members list, select members.
4. Click  to add members to the Selection list.
5. Click **OK**.

   The members in the Selection list are displayed in the POV dimension drop-down list.

Filtering Dimension Member Selections

- Children
- Descendants
- Level
- Generation
- UDA
- Attribute
- Varying Attribute
- Subset



Filtering Dimension Member Selections

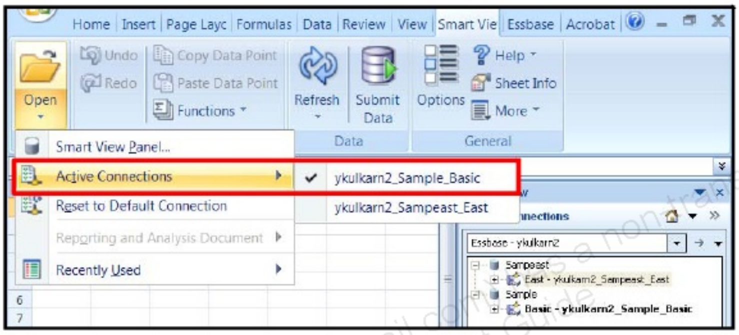
The Member Selection dialog box provides the following set of member lists for filtering the hierarchy view:

- Children
- Descendants
- Level
- Generation
- UDA
- Attribute
- Varying Attribute
- Subset

You can filter the entire dimension, or you can use a specific dimension member as the filter parameter. To apply advanced filtering, highlight a member without selecting its check box, and then select a filter option from the Filter drop-down list. For example, if you highlight OEM and then change the filter list to Level 0, only the level 0 descendants of OEM are displayed.

NOTE: The Attribute, Varying Attribute, and Subset member lists use attributes for filtering. However, the Subset member list enables you to specify and combine multiple attributes by using AND-OR logic. The Attribute and Varying Attribute filters enable you to select only one attribute or varying attribute.

Associating Data Sources with Worksheets



ORACLE

Associating Data Sources with Worksheets

You may need to associate a worksheet with a data source to perform certain actions in Smart View; for example, to create a free-form grid with the Member Selection dialog box, or to create queries with Query Designer.

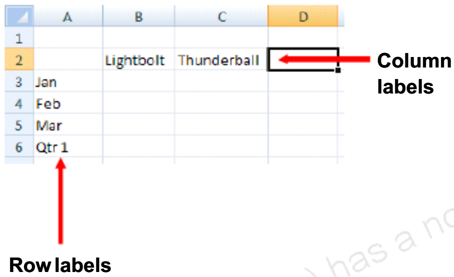
You must save a shared connection as a private connection to associate it with a worksheet.

To associate data sources with worksheets:

1. Right-click a private connection. and select **Connect**.
2. On the Smart View ribbon, click **Open**.
3. Select **Active Connections**, and then **[Connection Name]**.

   A check mark is displayed next to the connection name.

Creating Free-Form Grids



Creating Free-Form Grids

If you are familiar with the member names (labels) of dimensions in the database outline, you can enter them into spreadsheets to create grid layouts.

Free-Form Label Placement Guidelines

When you create a free-form grid, Essbase initiates a label-scanning process to match report labels on the spreadsheet with members in the outline. The POV section of the worksheet is scanned first, followed by column and row sections.

When at least one label is matched for each dimension, Essbase can place data in the spreadsheet, assuming that the labels were created according to the following general, POV, row, and column label rules. Otherwise, Essbase stops the requested retrieve action and displays a message that describes the error condition.

General Rules:

- Worksheet labels must match outline member names or their aliases.
- All standard dimensions must be represented in the POV or row-column sections for Essbase to resolve a data point.
- All member names that consist of numbers must be preceded with a single quotation mark. For example, if the account number for sales is 14000, enter `'14000` in the spreadsheet cell.

POV Rules:

- The selected POV members define all data on the worksheet for their dimensions.
- If a dimension is represented in the POV, members from the dimension cannot be displayed in a row or column.
- In a sheet that contains row and column dimension members, if no member from a dimension is found in the free-form grid during the label-scanning process, Essbase places the dimension name (generation 1) of the missing dimension in the POV

Row and Column Rules:

- A report must include at least one row and one column dimension.
- Each column dimension must be on a separate row that precedes the rows containing row dimensions. All members of a given column dimension must be displayed on the same row.
- Each row dimension must be in a separate column that precedes the columns containing column dimensions. All members of a given row dimension must be displayed in the same column.
- A row or column dimension can be grouped by, or nested in, another row or column dimension. The number of nested dimensions is unlimited, as long as at least one row dimension and one column dimension exist.

Adding Member Names

To create free-form grids, you can type some dimension member names directly into the spreadsheet, and use the Member Selection dialog box to enter other member names.

To paste member selections into empty cells:

1. Associate a connection with a worksheet.
2. Select an empty cell, and click **Member Selection** on the Essbase ribbon.

   The Dimension Name Resolution dialog box is displayed.
3. In the Dimensions drop-down list, select a dimension.
4. **Optional:** Select the **Vertical Orientation** check box to designate a row dimension (as opposed to a column dimension)
5. Click **OK**.

   The Member Selection dialog box is displayed.
6. Create a member selection list, and click **OK** to close the Member Selection dialog box.

   The dimension members are pasted into the spreadsheet.

Summary

In this lesson, you should have learned to:

- Describe the Smart View architecture and user interface
- Connect to data sources
- Create ad hoc grids
- Set the point of view
- Associate data sources with worksheets
- Create free-form grids

LESSON 10

Creating Reports with Smart View

Objectives

At the end of this lesson, you should be able to:

- Update Essbase data
- Integrate Essbase data with Microsoft Office
- Create shared database perspectives
- Create custom reports

Updating Essbase Data

- Adjusting data
- Submitting data
- Calculating data

Updating Essbase Data

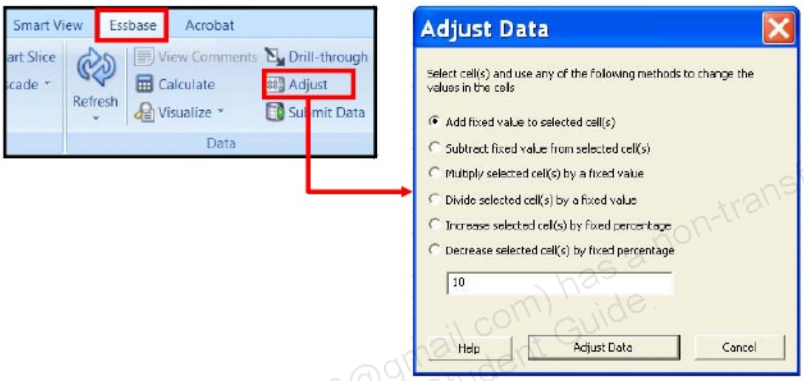
During database development, the spreadsheet environment is a key tool for validating data and calculations. Depending on your database needs, users might use Smart View to adjust and submit data during production cycles, as in an interactive write-back forecasting application.

Here is a typical data entry process:

1. Adjust or enter data.
2. Submit data to Essbase.
3. Recalculate the database using a calculation script.

Smart View provides tools to adjust data, write data back to Essbase databases, and execute calculation scripts.

Adjusting Data Values



Adjusting Data Values

Smart View enables you to quickly adjust data through the following operations:

- Increase or decrease selected cells by a percentage
- Add or subtract a fixed value to selected cells
- Multiply or divide selected cells by a fixed value

You cannot adjust data values that display `#Missing` or `#NoAccess` values. When you change cells in Smart View grids, the changed cells (also called *dirty cells*) are displayed by default with a different background color than the rest of the cells in the grid. When you submit data to Essbase, only dirty cells are submitted.

NOTE: You can customize the formatting for dirty cells in your Smart View options to use different fonts or font styles instead of color cues (for example, strikeout or italic font). If Excel formatting is enabled in Smart View options, dirty cell formatting is not displayed.

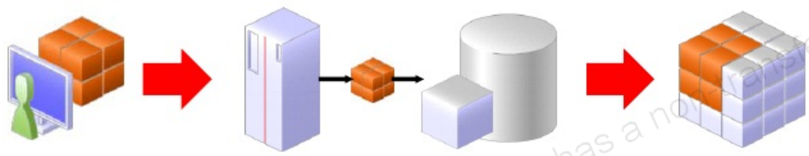
To adjust data va lues:

1. In an ad hoc grid, select the data cells whose values you want to adjust.
2. On the Essbase ribbon, click**Adjust**.

   The Adjust Data dialog box is displayed.
3. Select an adjustment option.
4. In the text box, enter the percentage or number by which you want to adjust the value of the selected cells.
5. Click **Adjust Data**.

   The values in the selected cells are adjusted.

Submitting Data



Submitting Data

You must have write permission for the database cells that you are changing. Otherwise, the following message is displayed: "You do not have sufficient access to perform a lock on this database."

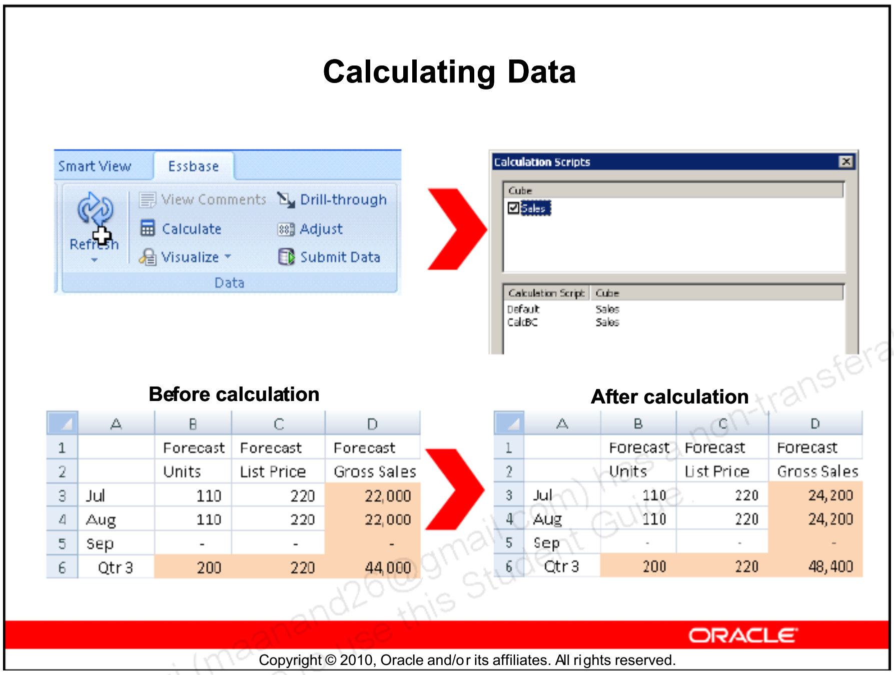
Here is the data submission process:

1. The user enters new data in Smart View and submits the data to Essbase.
2. Provider Services sends the adjusted data cells to the Essbase server.
3. Changed cells are updated in the database.

To submit data:

- On the Smart View or Essbase ribbon, click **Submit Data**.

  The changed cells are updated in the data source.



Calculating Data

After submitting data to the Essbase database, you must recalculate the database to see the results of your changes. You can calculate only the calculation scripts to which the administrator granted you privileges.

On the slide, the left example shows that a user changed the number of units forecast in July and August from 100 to 110. However, calculated values such as Gross Sales (Units * List Price) and Qtr 3 (Jul + Aug + Sep) still display their previously calculated totals. In the example on the right, the data was recalculated and the totals for Gross Sales and Qtr 3 now reflect the new unit values.

To calculate d ata:

1. On the Essbase ribbon, click**Calculate**.

   The Calculation Scripts dialog box is displayed.

2. From the Calculation Script list, select a calculation script.

3. Click **Launch**.

   The following message is displayed: “Calculation script has been processed.” A status message indicates whether the calculation was successful.

4. Click **OK**.

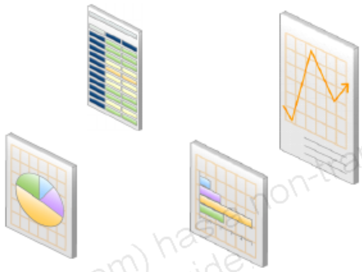
5. Click **Close** to return to your worksheet.

6. On the Essbase or Smart View ribbon, click **Refresh** to retrieve calculated data into the spreadsheet.

NOTE: New calculation scripts to which you have access are automatically displayed in the Calculation Scripts dialog box as they are added to the database.

Integrating Essbase Data with Microsoft Office

- Dynamic data points
- Linked views



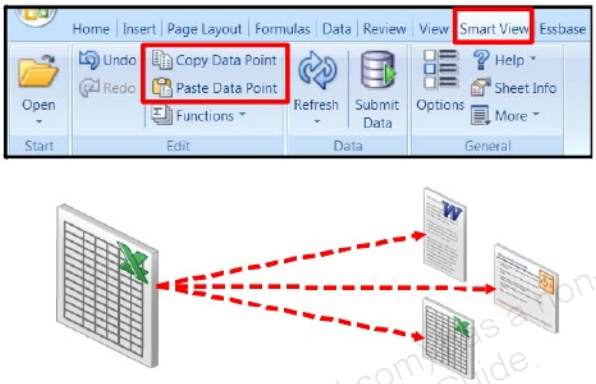
Integrating Essbase Data with Microsoft Office

In addition to providing Excel integration, Smart View enables you to view and refresh Essbase data in PowerPoint, Word, and Outlook. You can copy data points and move them from one Office component to another, creating *dynamic data points*. Users with Smart View access can see the dimensionality and detail of the data points by opening views called *linked views*. Dynamic data points and linked views provide the following benefits:

- Office-based workflow
- One-click visualizations of underlying data
- Multiple data source grids

NOTE: Dynamic data points and linked views for Essbase data are available only in Smart View 9.3 or later.

Copying Dynamic Data Points



Copy data points from Excel

Paste data points into Word, PowerPoint, or another Excel worksheet

Copying Dynamic Data Points

You can copy dynamic data points between Office components. The dynamic data points retain their srcinal point of view and connection information.

TIP: To view and refresh data in Outlook, you must set Word as the text editor.

To copy dynamic d ata points:

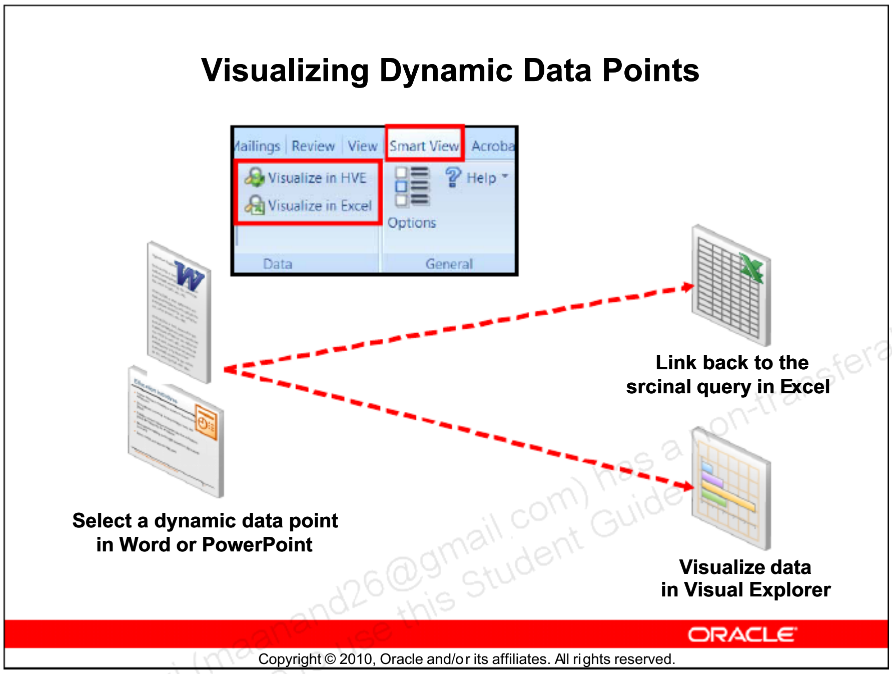
1. In an Excel worksheet, select a range of Essbase data cells (you can also include member names in your selection).
2. On the Smart View ribbon, click **Copy Data Point**
3. Open a Word document, PowerPoint document, Outlook e-mail, or another Excel spreadsheet.

4. On the Smart View ribbon, click **Paste Data Point**

   A grid containing dynamic data points is displayed. The data cells contain the same information—`#NEED_REFRESH`—until you refresh the formulas. Member names are copied, but they are not dynamic. After pasting, you can change member name fields to custom text without affecting the data points.

5. On the Smart View ribbon, click **Refresh**.

   Data cells are refreshed from the database.



Visualizing Dynamic Data Points

From a dynamic data point in Word or PowerPoint, you can either display the srcinal Excel query that created the data point or visualize the same query in Oracle® Essbase Visual Explorer.

Linking Back to the Original Excel Query

Dynamic data points store the entire database query from which they were srcinally copied, which enables you to re-create the query in a new Excel worksheet. After executing the query, you can perform ad hoc analysis on the data in Excel.

Visualizing Data in Visual Explorer

Visual Explorer creates a graphical representation of the data based on the underlying query that produced the dynamic data point, and provides a selection of chart types as your query is launched. The data is rendered in your selected chart format, and you can perform ad hoc visual analysis on the query.

To retrieve an Excel query from which data points were copied:

1. In Word or PowerPoint, select a dynamic data point.
2. On the Smart View ribbon, click **Visualize in Excel**.

To visualize data in Visual Explorer:

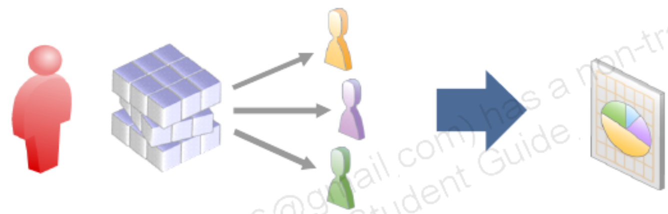
1. In Word or PowerPoint, select a dynamic data point.
2. On the Smart View ribbon, click **Visualize in HVE**.

   The Show Me! dialog box is displayed.
3. Select a data view option, and click **OK**.

Creating Shared Database Perspectives

Leveraged by users:

- As a predefined starting point for ad hoc operations
- As a starting point for customized data filters
- To create report components in Report Designer



Creating Shared Database Perspectives

A smart slice is a reusable perspective of a data source that contains a restricted set of dimension members.

An organization can have as many smart slices as it needs to accommodate the data requirements of its users. For example, you can create smart slices for geographical regions, product lines, time frames, or any combination of such entities.

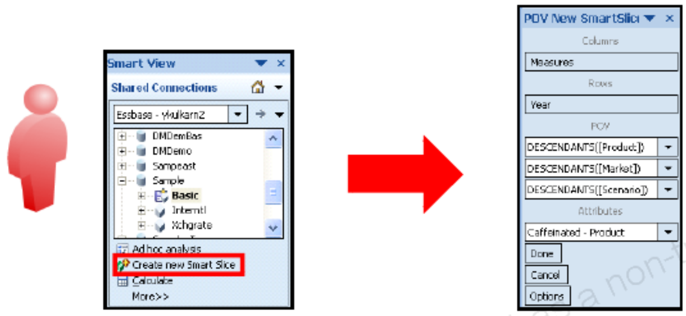
You can view and work with any data within the boundaries of s smart slice. For example, Bigcorp administrators can create a smart slice for srcinal equipment manufacturer (OEM) customer representatives that, by default, shows product sales only for OEM customers.

After smart slices are defined, users can leverage them in the following ways:

- **As a predefined starting point for ad hoc operations:** Users can create ad hoc reports by selecting a smart slice for ad hoc analysis instead of the database node.

- **As a starting point for customized data filters:** Users can modify smart slices by adding them to Report Designer. User-modified smart slices are stored locally with the worksheet, and are not reusable like administrator-created smart slices.
- **To create report components in Report Designer:** Users can add smart slices and modified smart slices to Report Designer in the Smart View panel. They can then create interactive report components based on the smart slices or modified smart slices.

Creating Smart Slices



Smart View panel **Smart Slice designer**

Creating Smart Slices

Smart slices are created by administrators who have Administrator privileges for the relevant data source. Smart slices are available to users in the Smart View panel under the corresponding database.

To create smart slices:

1. Connect to a data source.
2. In the Smart View Panel, select the data source.
3. Click **Create new Smart Slice**.

   The Select Alias Table dialog box is displayed.
4. Select an alias table, and click **OK**.

   The New SmartSlice - Design dialog box is displayed.

5. Arrange the dimensions and select dimension members:

   a. Drag dimensions to arrange them under Columns, Rows, and POV.

TIP: The column and row layout areas must always contain at least one dimension. To move dimensions from columns or rows, first move replacement dimensions to the layout areas.

---

   b. Click column and row dimensions to select dimension members.

   c. From the a POV dimension drop-down list, select **..** (ellipsis) to select dimension members.

6. **Optional:** Click **Options** to preset query options such as suppression and zoom behavior.

7. Click **Done**.

   The POV Member Selection dialog box is displayed.

8. Set the POV using member selection lists, and click **OK**.

   A text box is displayed in the Smart View panel.

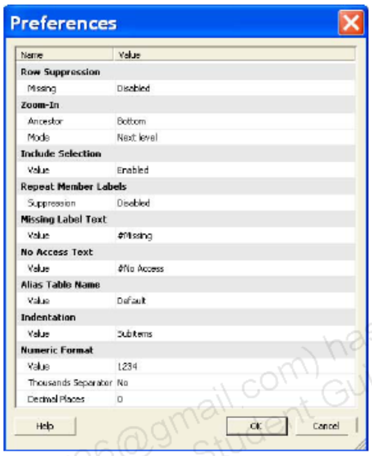
9. Enter a name for the smart slice.

10. Click  .

    The smart slice is displayed in the Smart View panel under the database node.

NOTE: You can also create a smart slice from an existing ad hoc report, by clicking Smart Slice on the Essbase ribbon.

Setting Smart Slice Preferences



Setting Smart Slice Preferences

The preferences that you specify for a smart slice are stored as part of the smart slice definition. Your specified preferences override the global preferences that are set in the Options dialog box.

To set smart slice preferences:

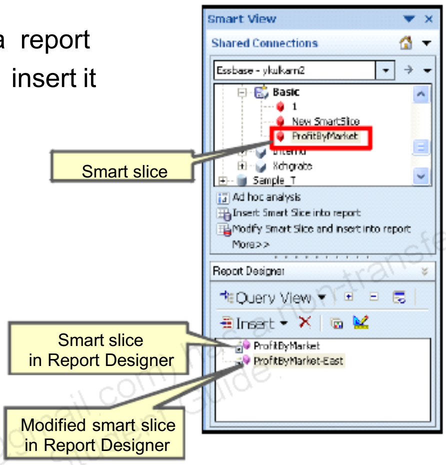
1. Perform one of the following actions to display the Preferences dialog box:
   - From the smart slice designer, select **Options**.
   - Right-click a smart slice, and select **Smart Slice preferences**.

2. Select preferences. The following table describes available settings:

Name	Value
Row Suppression	Disabled: Displays data rows that contain only cells with missing data Enabled: Suppresses data rows that contain only cells with missing data
Zoom-In: Ancestor	Bottom: Retrieves data for the lowest members of a dimension Top: Retrieves data for the highest member of a dimension
Zoom-In: Mode	Children: Retrieves data for the children of the current member Descendants: Retrieves data for the descendants of the current member Base: Retrieves data for level 0 members
Include Selection	Enabled: Retains the selected member and all members that a zoom operation retrieves
Repeat Member Labels	Disabled: Displays repeated member names Enabled: Suppresses repeated member names
Missing Label Text	Displays specified text in cells that contain missing data
No Access Text	Displays specified text in cells that contain data which the user does not have privilege to view
Alias Table Name	Overrides the alias table that is selected for the smart slice
Indentation	None:Doesnotindent Subitems: Indents only descendants Totals: Indents only ancestors
Numeric Format: Value	A sample number that displays your thousands-separator and decimal-places selections
Numeric Format: Thousands Separator	Yes: Uses a thousands separator No: Does not use a thousands separator
Numeric Format: Decimal Places	Specifies the number of decimal places that are displayed for data entries.

Creating Custom Reports

- Insert a smart slice into a report
- Modify a smart slice and insert it into a report



ORACLE

Creating Custom Reports

Custom reports comprise report components that display the results of smart slice queries or user-modified smart slice subqueries.

You can add smart slices and modified smart slices from any data source for which smart slices are supported, enabling you to combine data from multiple sources in one report or dashboard. Report definitions (including modified smart slices) are saved with the worksheet. When you open a saved workbook that contains a custom report, you can access your report definition through the Smart View panel.

To add smart slices to reports:

1. In the Smart View panel, select a smart slice.
2. Select **Insert Smart Slice into report**.

   The smart slice is displayed in the Report Designer frame at the bottom of the Smart View panel.

To insert modified smart slices into reports:

1. In the Smart View panel, select a smart slice.
2. Select **Modify Smart Slice and insert into report**.

   The smart slice designer dialog box is displayed.

3. Modify and name the smart slice. For details, see "Creating Smart Slices" in this lesson.

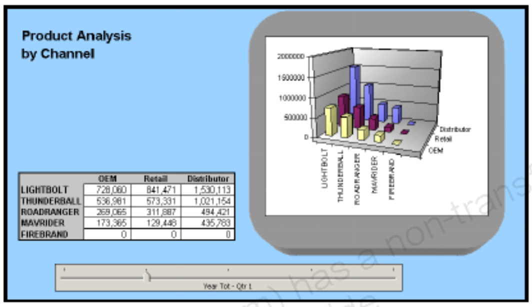
   The modified smart slice is displayed in the Report Designer frame at the bottom of the Smart View panel.

NOTE: Modified smart slices are created locally by users, and thus are not reusable like unmodified smart slices (which are created by administrators and stored centrally).

Interactive Report Components

Reporting objects:
- Function grid
- Table
- Chart

Controls:
- POV
- Slider



ORACLE

Interactive Report Components

Smart View provides the following report objects to create custom dashboards based on smart slices:

- **Function grid:** A static grid with data represented by HsGetValue functions. Function grids are highly formattable, but you cannot drill down or change the grid layout. The characteristics of a cell are displayed when you position your cursor over the cell.
- **Table:** An interactive grid in which you can drill and pivot. Tables float on the document and can be moved and resized. Tables are useful for displaying large queries in a small space; their scroll bars provide quick access to rows and columns while occupying only a set space on the worksheet.
- **Chart:** Microsoft Chart objects. Charts float on the document and can be moved and resized.

Additionally, Smart View provides controls for POV dimensions that you can attach to smart slices in the Report Designer panel. The control affects any report component that is attached to the same smart slice as the control.

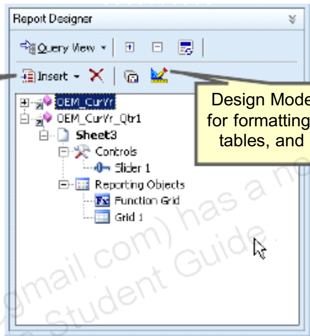
NOTE: In Report Designer, although you can attach multiple report components to one smart slice, you can attach only one type of control to a smart slice.

Smart View provides the following controls:

- **POV:** A report POV that is linked to all function grids, tables, and charts attached to the same smart slice as the POV control. Similar to an ad hoc POV, a report POV contains all possible POV dimensions for its related components and is customizable in the Member Selection dialog box. If you use a POV control for a smart slice, you cannot also use a slider.
- **Slider:** A slider control that is linked to all function grids, tables, and charts attached to the same smart slice as the slider control. Sliders are used to change a single POV dimension of their related report components. You can attach multiple sliders to a smart slice—one for each POV dimension. If you use sliders for a smart slice, you cannot also use a POV control.

Creating Reports with Report Designer

1. Insert smart slices into Report Designer.
2. Insert report components into the report by attaching them to specific smart slices.
3. Format the report.



Report Designer toolbar

Design Mode toggle for formatting sliders, tables, and charts

Creating Reports with Report Designer

After adding all your smart slices to Report Designer, you attach report components to the smart slices in the Report Designer frame. You can attach multiple components to one smart slice. Function grids inhabit cells in the worksheet, and you can format or move them with standard Excel formatting tools. However, tables, charts, and sliders lay on top of the worksheet cells, and you must move and resize them in design mode.

To insert report components into the report:

1. In the Report Designer frame, select a smart slice.
2. Click **Insert**.
3. Select a type of component.

   The component is displayed in the worksheet.

To format tables, charts, and sliders:

1. In the Report Designer frame, click  .
2. In your report, drag the object control handles to resize or move a table, chart, or slider.
3. In the Report Designer frame, click  again to reenter edit mode.

Summary

In this lesson, you should have learned to:

- Update Essbase data
- Integrate Essbase data with Microsoft Office
- Create shared database perspectives
- Create custom reports