IBM® DB2® OLAP Server™ 8.1

Integration Server Data Preparation Guide

SC18-7006-00

Printed in the U.S.A.

Contents

Preface

Purpose

This guide outlines steps for preparing your relational data for a smooth transition to a Hyperion® Essbase® database. In addition, this guide provides an introduction to OLAP concepts and familiarizes you with Essbase Integration Services components. Finally, it presents an overview of the information that you need for creating and working with OLAP models and metaoutlines.

Audience

This guide is for Essbase Integration Services database administrators who are responsible for installing, implementing, and deploying the system.

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

- Knowledge of where the data for your business resides; for example, in a relational database
- Knowledge of how to create and maintain Open Database Connectivity (ODBC) data sources
- Knowledge of the data requirements for your business, so that you can apply the Essbase Integration Services product family to your specific application
- A fundamental understanding of Microsoft Windows and basic Microsoft Windows terminology, such as dialog box, list box, and button

  See Microsoft Windows documentation for more information on these terms.
- Experience with the setup and operation of Hyperion Essbase OLAP Server
- A basic understanding of multidimensional concepts

Document Structure

This document contains the following information:

- Chapter 1, “About OLAP and Essbase Integration Services,” introduces basic OLAP and multidimensional concepts and describes how to use the Essbase Integration Services Console to create OLAP models and metaoutlines.
- Chapter 2, “Preparing Relational Data Sources,” describes how to prepare relational data for transfer to an Essbase database.
- Index, lists Essbase Integration Services terms and their page numbers. In the PDF version of this guide, select an index entry to view the relevant page.

Sample Applications

The examples used in this book are based on a derivative of the two sample databases provided with Essbase Integration Services. The databases are called TBC (external data source) and TBC_MD (OLAP Metadata Catalog). The TBC_MD OLAP Metadata Catalog contains a sample OLAP model and a sample metaoutline. The OLAP model is called TBC Model, and the metaoutline is called TBC Metaoutline.

Note: The capitalization of column and table names in the sample applications depends on the relational database management system (RDBMS) that you use.

The system administrator who installs Essbase Integration Services is responsible for making the sample databases, the OLAP model, and the metaoutline available to end users. It is the Essbase Integration Services system administrator’s responsibility to respond if any of the following problems occur when users launch Essbase Integration Services Console:

- Users cannot find the sample databases, the OLAP model, or the metaoutline.
- Users do not have adequate access to the sample databases, the OLAP model, or the metaoutline.
- Users do not see any data in the sample databases.

Where to Find Hyperion Documentation

Essbase Integration Services documentation is accessible from the following locations:

- You can find the documentation in softcopy format in the DB2 OLAP Server 8.1 folder.
- Online help is accessible from the Essbase Integration Services Console. Start the product and click the Help button or menu command.
- The DB2 OLAP Server Web site is located at: `www.ibm.com/software/db2/db2olap/library.html`

Online Help

➤ To access online help, click Help in any dialog box or select Help > Help Topics.

➤ To print an online help topic, display the topic and select File > Print or click the Print button.

Conventions

The following table shows the conventions used in this document:

Table i: Conventions Used in This Document

Item	Meaning
➤	Arrows indicate the beginning of a procedure consisting of sequential steps.
Brackets []	In examples, brackets indicate that the enclosed elements are optional.
Bold	Bold in procedural steps highlights major interface elements.
CAPITAL LETTERS	Capital letters denote commands and various IDs. (Example: LOADALL command)

Table i: Conventions Used in This Document (Continued)

Item	Meaning
Ctrl + 0	Keystroke combinations shown with the plus symbol (+) indicate that you should press the first key and hold it while you press the next key. Do not type the + symbol.
`Example text`	Courier font indicates that the material shown is a code or syntax example.
`Courier italics`	Courier italic text indicates a variable field in command syntax. Substitute a value in place of the variable shown in Courier italics.
`ARBORPATH` *`ISHOME`*	When you see the environment variables *`ARBORPATH`* or *`ISHOME`* in Courier italics, substitute the value of ARBORPATH or ISHOME from your site.
n, *x*	Italic *n* stands for a variable number; italic *x* can stand for a variable number or an alphabet. These variables are sometimes found in formulas.
Italics	Italics in a product-related term in the body of a book indicates that the term is also included in the glossary of the book.
Ellipses (...)	Ellipsis points indicate that text has been omitted from an example.
Mouse orientation	This document provides examples and procedures using a right-handed mouse. If you use a left-handed mouse, adjust the procedures accordingly.
Menu options	Options in menus are shown in the following format: *Menu name* > *Menu command* > *Extended menu command* For example: File > Desktop > Accounts

Note: The term right-click, used throughout this guide, means to click the secondary mouse button to open a pop-up menu.

Chapter

1

About OLAP and Essbase Integration Services

Essbase Integration Services provides a suite of graphical tools to create OLAP models and metaoutlines and populate Essbase databases. You use tables in a relational data source to define a logical model that represents the data in an online analytical processing (OLAP) context. You then use the OLAP model to create a metaoutline that serves as a template for an Essbase database outline.

This chapter explains basic OLAP concepts, provides an overview of Essbase Integration Services, describes OLAP models and metaoutlines, and defines the workflow for creating OLAP models and metaoutlines.

This chapter contains the following topics:

About OLAP

OLAP is a decision-support computing environment for business managers who need to analyze consolidated enterprise data in real time. OLAP enables users to answer complex "what if" questions and to create sales and marketing scenarios to test budgeting, sales promotion, and sales planning strategies. Essbase OLAP Server supports a multidimensional, multi-user database that enterprise users can access using standard retrieval tools, such as spreadsheets.

Essbase OLAP Server supports multiple views of data sets for users who need to analyze relationships between data categories. For example, a decision-support user might ask the following questions:

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

You can use Essbase Integration Services to create an Essbase database that enables users to answer these types of questions quickly and easily.

You can use Essbase Integration Services Console to create a logical data model that represents the *relational databases* that contain the data that you want to analyze. Thus, you take the first step in advancing from relational databases to an Essbase database.

About Multidimensional Databases

A *multidimensional database* (MDDB) stores consolidated data at the intersections of its members and dimensions. For example, if a company sells a total of `20` units of all products in the East region in the first quarter, Essbase stores `20` at the intersection of Product, East, Quarter1, and Unit Sales.

A *dimension* is a data category that represents a core component of a business plan, and it often relates to a business function. Product, Region, and Year are typical dimensions. In most databases, dimensions are static, rarely changing over the life of the application.

A *member* is an individual component of a dimension. For example, Product A, Product B, and Product C might be members of the Product dimension. Each member has a unique name. A dimension can contain a large number of members. In some dimensions, members change frequently over the life of the application.

Simultaneously, members can be parents of some members and children of other members. The Essbase outline indents members below one another to indicate a consolidation relationship. For example, sales totals for the Products dimension might be totalled by product description, broken down by product code, and further broken down by product ID.

User Interactions with Data

Essbase OLAP Server consolidates and calculates data to provide different views of the data. Using a multidimensional database, here are some of the tasks that users can perform:

Consolidate (aggregate or roll up) data

The *consolidation* process computes the data relationships for all parent and child combinations within a dimension. For example, the consolidation for the Year dimension is as follows:

```
Year = Quarter1 + Quarter2 + Quarter3 + Quarter4
```

A consolidation is typically additive, but it can be any type of calculation.

Create sophisticated “what if” scenarios

Assume that you set a sales goal of ten percent growth for the upcoming year on all product lines. With Essbase, you can compare sales forecasts estimated by planners with actual sales data (retrieved from the *online transaction processing* [OLTP] database) to see how close you are to achieving your sales goals. If actual sales run lower than projected sales for a given period, salespeople can access the forecast data, input new product sales scenarios (for example, What if I sell 2000 widgets to our biggest corporate customer?), update the forecast data, and provide revised figures to headquarters.

Input strategic planning assumptions

Assume that your company is planning for 50 percent growth over the next three years. You have a fairly good idea of how many new products you need, but how do you know how many new engineers, salespeople, and support personnel you can afford to add while still optimizing profits and gross margin?

With Essbase, you can input projected sales and expenses for each product and calculate downward to determine the cost of goods sold. Thus, you obtain a realistic picture of the bottom line. If the picture does not look practical, you can create different scenarios with different mixes of products, people, and expenses until you produce the profit picture that you require.

Conduct spreadsheet operations

To *drill down* or *drill up* on data retrieves progressively more detailed or progressively more generalized data relative to a selected dimension. Drilling down on a database dimension provides you with greater detail on the dimension. Drilling up provides you with a higher level (more summarized) view of the dimension. For example, you can drill down on the Year dimension to view the values for each quarter. Then you can drill down on Quarter1 to view the values for each month of Quarter1. You can view Quarter1 sales for the Chicago office and then drill up to view sales totals for the entire East region.

Pivoting data alters the data perspective. When Essbase retrieves a dimension, it displays a configuration of rows and columns. A user can pivot (rearrange) the data to obtain a different viewpoint.

For more information about spreadsheet operations, such as drill down and pivot, see the Essbase Spreadsheet Add-in documentation.

Sources of Data

The data in a multidimensional database can originate from a variety of sources, such as OLTP databases, text files, and spreadsheet files. OLTP databases contain a wealth of operational data, such as the following information:

- How many units of Product A are on hand?
- What is the current customer’s name, address, and billing status?
- What is an employee’s salary, job title, and address?

Data in an OLTP database must be up-to-date and easy to change, and must use as little space as possible. The data is stored in rows and columns in relational database tables.

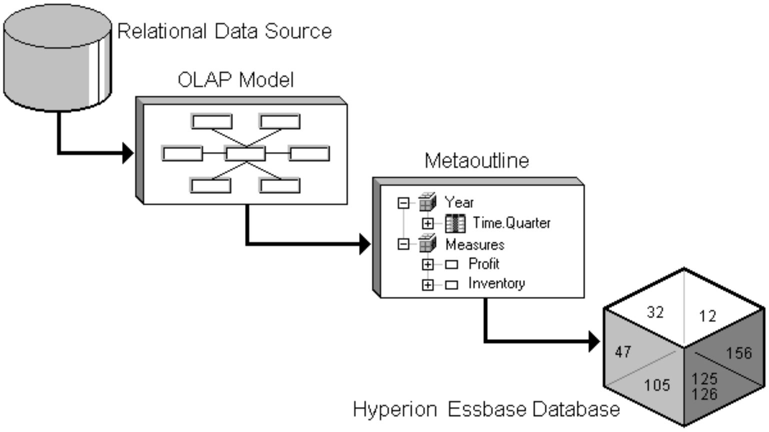
Essbase Integration Services enables you to access data for OLAP models or metaoutlines from one or more data sources that support SQL 89 or later. You can access data in the data sources using Open Database Connectivity (ODBC), the Merant Wire Protocol, or the client software provided by the RDBMS manufacturer.

About Essbase Integration Services

Essbase Integration Services quickly transfers data from relational tables to an Essbase database quickly. To perform the transfer, you must determine which data to transfer and consolidate the selected data into a form that is useful for decision-support users. Then you must identify the tables, rows, or columns that contain the data and determine how they map to the structure of the multidimensional database.

Figure 1illustrates the Essbase Integration Services workflow:

Figure 1: Workflow for Creating an Essbase Database from a Relational Data Source



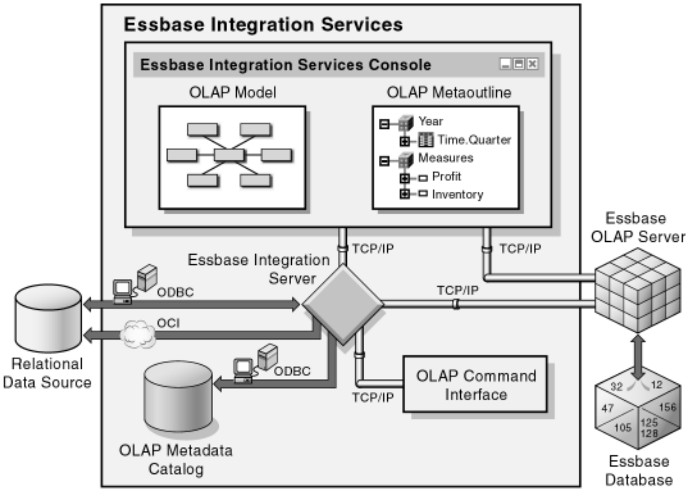
Use Essbase Integration Services Console to perform the following tasks:

- Use the tables, views, and columns in one or more relational databases to create an OLAP model. An *OLAP model* is a logical *star schema* consisting of a fact table that is surrounded by related dimension tables.
- Use the OLAP model to create a *metaoutline*, an outline template that contains the structure and the rules required to generate an Essbase outline.
- Use the metaoutline to create and populate an Essbase database.

Components of Essbase Integration Services

Essbase Integration Services, pictured in Figure 2, consists of two major components: Essbase Integration Services Console and Essbase Integration Server.

Figure 2: Essbase Integration Services



Essbase Integration Services Console

You use Essbase Integration Services Console, a graphical user interface, to create OLAP models and metaoutlines and to populate Essbase databases.

To create an OLAP model or metaoutline, you must connect to an OLAP Metadata Catalog and the appropriate relational data sources. For more information, refer to the online help.

Essbase Integration Server

Essbase Integration Server is the primary component of Essbase Integration Services. Essbase Integration Server is software that uses the information stored in the OLAP Metadata Catalog to extract from the relational data sources the dimension and member names needed to build an associated Essbase outline. When the Essbase outline is complete, Essbase Integration Server extracts data from the relational data sources, performs the operations specified in the associated metaoutline, and loads the data into the Essbase database. For more information about Essbase Integration Server, see the *Essbase Integration Services System Administrator's Guide*.

Essbase Integration Server includes the following subcomponents, as illustrated in Figure 2 on page 19:

- **OLAP Metadata Catalog:** *OLAP Metadata Catalog* is a Structured Query Language (SQL) relational database that contains the following information:
  - Metadata describing the nature, source, location, and type of data to retrieve from the relational data sources
  - Metadata describing the information required to generate an Essbase outline
  - OLAP models and metaoutlines

    You can create more than one OLAP Metadata Catalog to store OLAP models and metaoutlines. Using XML Import/Export, you can move OLAP models and metaoutlines to a different OLAP Metadata Catalog after you have created and saved them to a specific catalog.

  The OLAP Metadata Catalog is a data source that is configured for Open Database Connectivity (ODBC). If you do not know how to create an ODBC data source, see the *Essbase Integration Services Installation Guide* or the ODBC user documentation.

- **Essbase Integration Services Shell:** *Essbase Integration Services Shell* is a command-line tool used to access Essbase Integration Server to load members and data to an Essbase database. For more information, see the *Essbase Integration Services System Administrator's Guide*.

Workflow for Using Essbase Integration Services

Figure 2 on page 19 provides an overview of the Essbase Integration Services components that you use to prepare the relational data for OLAP reporting through Essbase OLAP Server.

➤ To create an Essbase database from a relational data source:

1. Build an OLAP model that is based on the tables in the appropriate relational data sources.

   Essbase Integration Server stores the OLAP model and the information necessary to retrieve the relevant tables in OLAP Metadata Catalog.

2. Create a metaoutline from the OLAP model.

   Essbase Integration Server stores the metaoutline in OLAP Metadata Catalog.

3. Load members and data into the Essbase database.

4. Update the Essbase database with new members and data.

About OLAP Models

An OLAP model contains a star schema. OLAP models are based on the idea that values in a relational database can be categorized as either facts or dimensions of facts. Facts are the numeric, variable values in the database, such as sales figures and numbers of units sold. Associated with facts are related data values that provide additional information, such as store locations and product IDs of units sold. An OLAP model contains a fact table, one or more dimension tables, and one or more dimension branches. An OLAP model may also contain time and accounts dimensions.

Unlike other integration products, Essbase Integration Services creates an OLAP model that is a logical model, not a physical star schema. The OLAP model is a logical representation of the data values that you select from the tables in the appropriate relational databases and that you want to report in Essbase.

The sample application provided with Essbase Integration Services includes a sample OLAP model named TBC Model.

About Using OLAP Models

Use an OLAP model to create one or more metaoutlines. A metaoutline contains the basic structure required to build an Essbase outline and to load data into Essbase. You can use one OLAP model as the basis for another OLAP model by saving the original OLAP model under a different name and editing it as needed to meet reporting requirements. You can create any number of OLAP models for use in building metaoutlines. However, each metaoutline is based on one, specific OLAP model.

OLAP models have the following features:

- They are reusable. You can use the same OLAP model as the basis for multiple metaoutlines.
- They provide a layer of abstraction that insulates the Essbase database outline from changes in the relational database.
- They enable you to create hierarchies to structure and summarize data from a relational database. You can use the hierarchies in multiple metaoutlines.

About the Fact Table

The *fact table* serves as a container for all numeric facts (for the measures data values that vary over time). In the OLAP model included with the sample application, the fact table consists of the SALES relational table: a table that contains sales figures, cost of goods sold figures, opening and ending inventory quantities, and other data columns of variable measures.

About Dimension Tables

A *dimension table,* such as a Product dimension table, serves as a container for relational tables. Each dimension table contains data that is related to the facts in the fact table. For example, the Product dimension table in the sample OLAP model contains a relational table in which the PRODUCTID is related to the sales figures and inventory quantities in the SALES table.

When a dimension table joins to the fact table, the dimension table and all dimension tables joined to it form a dimension. A dimension in an OLAP model represents a dimension that you want to report in Essbase. For more information on Essbase dimensions, see “About Multidimensional Databases” on page 15.

When you create a dimension in an OLAP model, the dimension becomes available for use in creating a dimension in an associated metaoutline. You can drag a predefined OLAP model dimension directly into the metaoutline to create a dimension. The newly created metaoutline dimension then becomes a dimension in the Essbase outline that you create when you perform a member or data load.

When you build a metaoutline, you can create user-defined dimensions that do not exist in the associated OLAP model. Use this option when you need a dimension, such as Scenario, that does not exist in the relational data source. If you create a Scenario dimension, you can add both Actual and Budget members to track measures, such as actual and projected revenue and profit, that might not be stored in the relational data source.

About Metaoutlines

A metaoutline is a template containing the structure and rules for creating a Hyperion Essbase outline. A metaoutline is based on the structure of an OLAP model you specify. Metaoutlines have the following features:

- They are reusable. You can use a metaoutline as the basis for more than one Essbase outline.
- They are centralized. You can define a metaoutline at a central location and use it to create multiple Essbase outlines in multiple locations.
- They enable you to create an Essbase outline that is based on criteria that you specify.
- They enable you to create Essbase databases on demand. You can create or update an Essbase database whenever it is necessary, either immediately or by scheduling periodic updates.
- They enable you to view sample Essbase outlines before you build them.
- They automatically generate the SQL necessary to retrieve data from an external data sources.
- They enable you to filter the data from the relational data sources before you build the associated Essbase outline.
- They enable you to transform data as you create an Essbase outline. You can transform member names as you create an Essbase outline or transform data as you load it into an Essbase database.

Components of a Metaoutline

A metaoutline contains all the following components:

- **One or more measures.** The Essbase database calculates data values, or *measures*, for each dimension intersection of the associated metaoutline. Measures include items such as SALES or COGS (cost of goods sold). Every metaoutline that you use to build an Essbase outline must include at least one measure. For more information, refer to the Essbase Integration Services Console online help.
- **Two or more dimensions.** A dimension in a metaoutline creates a dimension in the associated Essbase outline. For example, if you include a Product dimension in a metaoutline, the Essbase outline that you generate from that metaoutline also provides a Product dimension. You can drag dimensions from the OLAP model or create your own dimensions. For more information, refer to the Essbase Integration Services Console online help.
- **One or more member levels.** A *member level* in a metaoutline creates one or more members at the same level in the associated Essbase outline. For example, if the Product dimension of a metaoutline contains a PRODUCT_DESC member level, the Product dimension in the Essbase outline contains members, such as Birch Beer and Caffeine Free Cola, that correspond to the values in the PRODUCT_DESC member in the relational database.

  You can define a set of filters that determine which members of a member level Essbase Integration Services will add to the associated Essbase outline. You can also define transformation rules that determine what transformations, if any, Essbase Integration Services will perform on the members of a member level as it builds the associated Essbase outline. For more information, refer to the Essbase Integration Services Console online help.

  You can arrange member levels into a hierarchy. A *hierarchy* defines the organizational structure of a group of member levels. For example, a standard time hierarchy in a metaoutline might contain the Year, Quarter, and Month member levels. This hierarchy creates an Essbase outline with members such as 1999, Quarter1, and January.
- **Optional attribute dimensions.** Attribute dimensions in the metaoutline are based on attribute-enabled columns in the OLAP model. After an attribute dimension and member have been created, you can define attribute properties, such as Boolean and numeric ranges, that enable you to view business data in finer detail than would otherwise be easily available.

- **Optional Hybrid Analysis lower level members.** Hybrid Analysis integrates a relational database with an Essbase multidimensional database so that applications and reporting tools can directly retrieve data within both databases. By enabling Hybrid Analysis for dimension members in a metaoutline, users of spreadsheets and report writer interfaces can access data contained in the Essbase database and drill down to the data accessed directly from the relational data source.
- **Optional Drill-Through Report members.** Essbase Integration Services enables you to define drill-through reports. A drill-through report is based on an *intersection level* (member combination) that spreadsheet users can double-click to start the drill-through process. Then spreadsheet users view or customize pre-defined drill-through reports that retrieve the relevant detail columns from the relational data source. Because Essbase Integration Server captures the metadata necessary to create the Essbase outline, it returns the drill-through report in the context of the data that spreadsheet users are viewing.

You can also create hierarchies for the metaoutline as you create an OLAP model. Refer to the Essbase Integration Services Console online help for more information on hierarchies.

The sample application provided with Essbase Integration Services includes a sample metaoutline, TBC Metaoutline. You can see the dimensions named Year, Market, Measures, Product, Scenario, and Supplier. The Year dimension contains a hierarchy—Quarter and Month.

Chapter

2

Preparing Relational Data Sources

An OLAP model is a dimensional model of relational data in the form of a star schema. Before you can create an effective OLAP model, you need to understand and define the business needs and data sources that you have available. You may need to modify some data sources to make the transition from relational to multidimensional as easy and efficient as possible.

For information about creating an OLAP model, refer to the Essbase Integration Services Console Help.

This chapter contains the following topics to help you prepare relational data:

Defining User Needs

Before you start to design an OLAP model, consider the questions in this topic. By carefully constructing answers to these questions, you begin to create an effective OLAP model.

Remember that when you create an OLAP model, the ultimate goal is to create a multidimensional Essbase database. This topic assumes that you are familiar with the design principles for a multidimensional database. For more information, see "About OLAP" on page 14 and the *Essbase Database Administrator's Guide*.

What information (data) do users want to see in the Essbase database?

Examples of types of information are orders, sales, invoices, and the general ledger. Select data by combining an understanding of the business with an understanding of what data is available. After reviewing the data that you select, you decide whether to create one or multiple Essbase databases.

What is the maximum level of detail of data that users want to see in the Essbase database?

Select the level of detail by combining an understanding of the business with an understanding of the performance requirements of the Essbase database. (In general, the less detail stored in the Essbase database, the faster the performance.) For example, you can opt to store data that is aggregated by quarter or by month. The level of detail that you select is the basic level of detail that you represent in the fact table. In an OLAP model, the fact table is where the numeric measurements of the business are represented.

You can also use drill-through reports or Hybrid Analysis to offer users at the spreadsheet level alternative views of data and direct access to the relational source data.

What dimensions apply to each fact table row?

A dimension is a data category. Typical dimensions are Product, Market, and Time. Base the selection of dimensions on an understanding of how users want to view data. Each dimension has a number of members. For example, the Market dimension can include members representing New York, Boston, San Francisco, and Los Angeles. Each row in the fact table represents a combination of members, one from each dimension. For example, a row in the fact table can store the sales for Product A in New York in February.

What measures do you want to represent in the fact table?

Measures are numeric quantities that vary over time. Examples of measures are quantity sold, cost of goods sold, and profit. Not all numeric values are measures. For example, although product size is a numeric value, you probably prefer to represent it as a member of a product dimension rather than as a member of a measures dimension because users do not want to compare product size over time.

Defining Data Sources

Now that you have defined what data users want to see in the Essbase database, you need to define the data sources. Consider the following questions:

- Is the data clean? Consider the quality and integrity of the source data. For more information, see "Cleansing Data" on page 32.
- Is the data calculated by a procedure and not stored; for example, a discount figure that is calculated for a specific product at a specific time? You need to create such information as tables in the chosen relational database management systems (RDBMS). Consider creating a staging area to perform this task. See "Deciding Whether to Create a Staging Area" on page 30.
- Is the data in a single structured query language (SQL) database or in multiple SQL data sources? Essbase Integration Server can access multiple SQL sources; however, you can consolidate SQL tables into a single SQL data source for each chosen RDBMS.
- Is the data in flat files? Essbase Integration Server does not access flat files; you need to create flat files as tables in the chosen RDBMSs. Consider creating a staging area to perform this task. See "Deciding Whether to Create a Staging Area" on page 30. Alternatively, after you build the Essbase database, you can use Essbase Application Manager to load data. For more information, see the *Essbase Database Administrator's Guide*.

Consolidating Data into a Single Data Source

When you create an OLAP model, the data that you need can come from multiple SQL data sources.

However, if you want to consolidate your data into a single data source, you must take the following steps:

1. Select a target RDBMS to connect to the server running Essbase Integration Server.

   For a list of supported RDBMSs, see the *Essbase Integration Services Installation Guide*.

2. Create any data that does not exist as an SQL data source; for example, create relational tables in the chosen RDBMS for any information that is currently in flat files.

3. Consolidate the data source tables into a single database in the chosen RDBMS.

When you consolidate the data, you may want to create a staging area.

Deciding Whether to Create a Staging Area

A *staging area* is an RDBMS database that you create to meet the needs of a specific application. Typically, a staging area is small (a maximum of 1 to 2 GB) in comparison to a data warehouse or online transaction processing (OLTP) application. It is a snapshot of data; that is, it is not constantly updated with new data but is refreshed periodically from source data.

Using a staging area provides advantages, because you can use a staging area to perform any of the following tasks:

- Combine data from disparate or heterogeneous platforms without changing the original data in the RDBMS

  For example, you can combine data from various RDBMSs and flat files.

- Fine-tune data for a specific application

  For example, you can calculate a subset of the RDBMS data and store the calculated data in a staging area. You can then run faster queries on the precalculated data in the staging area.

- Create tables or views to denormalize the data so that it maps more easily to an OLAP model (see “Creating Views and Tables” on page 32)

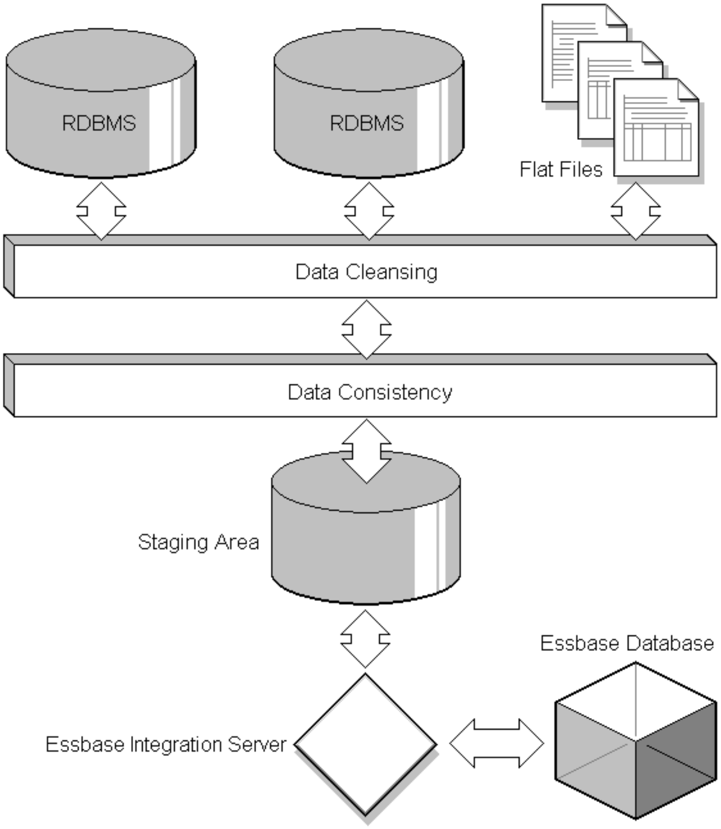
- Transform data that is not consistently described (see "Transforming Data" on page 40)
- Prepare data in the staging area for issues related to the year 2000.

  Year-2000 issues may occur if the RDBMS stores the year as two digits (for example, 99) and not four digits (for example, 1999).

The disadvantage of using a staging area is that the data in the staging area is a snapshot and may not represent the current data.

Figure 3 shows a staging area as part of a data warehouse. Data is copied into the staging area from the RDBMS source data and converted from flat files and other RDBMS sources.

Figure 3: Data Warehouse with Staging Area



Cleansing Data

Essbase Integration Server does not cleanse invalid data for you. Data needs cleansing if it is inconsistent. Inconsistent data may include values that are incorrect, are not of the correct data type, or do not match integrity constraints (containing rows that do not have entries for the required key columns). Also, data may be inconsistent because the same value is entered in different forms. Such an inconsistency occurs, for example, when data is drawn from more than one source or when users enter data incorrectly.

If source data is inconsistent, you must fix, or cleanse, the data. Cleansing data can be a simple process, such as making suspect data into nulls, or a more complex process that requires the use of a data-cleansing tool. Similar data requires similar cleansing rules.

Creating Views and Tables

Essbase Integration Server does not distinguish between the tables and the views of an RDBMS; you can use either tables or views to create an OLAP model. You may want to create views or tables specifically for use with Essbase Integration Server to provide security and convenience and to make the transition from relational to multidimensional as efficient as possible.

Consider creating views or new tables if you want to reach either of these results:

- To build an Essbase hierarchy down to a specific level in a recursive table, see "Building Essbase Hierarchies from Recursive Tables" on page 34.
- To create aliases in the Essbase database from data stored in multiple columns in the source data tables, see "Creating Aliases for Dimension and Member Names" on page 45.

Also consider creating views or tables if the source tables meet any of the following criteria:

- The source tables contain unions. Essbase Integration Server does not generate SQL for unions. See "Removing Unions" on page 36.
- The source tables have columns that you want to transpose to rows. See "Transposing Columns and Rows" on page 36.

- The source tables are in a packaged application. If the source tables are in a packaged application, you may not know which tables contain the data that you need because the table names provided by the application may not be meaningful in terms of your business needs. You will probably need to ask an application specialist to create the required tables and views (with meaningful names) in a staging area in the target RDBMS. See "Deciding Whether to Create a Staging Area" on page 30.
- The source tables are highly normalized. Normalized data is appropriately grouped and does not include redundant data. Consider denormalizing data to facilitate the transition of the data to an Essbase database. See "Denormalizing Source Data Tables" on page 38.

Deciding Whether to Create a View or a New Table

In most cases, you create views of the relational data (instead of new tables), because views ensure that data is current and efficiently stored. Also, with views, you do not need to maintain two sets of the same data.

When deciding whether to create a view or a new table consider the following questions:

- Does the data already exist? If the data does not exist, you need to create a new table. See "Consolidating Data into a Single Data Source" on page 29.
- Do you want to index columns that are not indexed in the source RDBMS table? You need to create a new table because you cannot index columns in a view. See "Creating Indexes" on page 40.
- Do you want to index columns that contain data that you need to transform? Because many RDBMSs ignore indexes on columns that have transformations, you probably need to create a new table. See "Transforming Data" on page 40.

If you are using a staging area, you create tables and views in the staging area. See "Deciding Whether to Create a Staging Area" on page 30.

Building Essbase Hierarchies from Recursive Tables

A r*ecursive table* contains information in one row that is a parent or child of information in a second row. Essbase Integration Server can build Essbase outline hierarchies from a recursive source table.

When creating an Essbase hierarchy from a recursive table, follow these guidelines:

- To associate aliases or user-defined attributes (UDA) with members created from a recursive table, ensure that the column with which you will associate the alias or UDA is fully defined. See "Creating Aliases or UDAs for Members in a Recursive Table" on page 34.
- When creating an OLAP model, join the recursive table to itself. For more information and for an example of a recursive source table, refer to the Essbase Integration Services Console Help.
- When creating a metaoutline, select the parent or child column that you want to filter on as a member level in the metaoutline. Refer to the online help for information on filters.

Building a Hierarchy Down to a Specific Level

To build the Essbase outline down to a specific level, you must create a view that contains data for only the levels that you want to build.

Creating Aliases or UDAs for Members in a Recursive Table

When you create an Essbase outline, you can associate aliases or UDAs with members in the outline.

If you are working with a recursive source table and you want to associate aliases or UDAs with members created from the recursive source table, you may need to prepare the data. The steps that you must take to associate aliases or UDAs vary, depending on whether the alias or UDA data is contained in the recursive source table or in a separate table.

If the alias or UDA data is in a separate table, you must complete specific steps when creating an OLAP model. The steps that you complete vary, depending on whether the column, with which you are associating the alias or UDA, is fully defined (see Table 1 and Table 4). For more information on the specific steps, refer to the Essbase Integration Services Console Help.

If the alias or UDA data is in the recursive source table, you must complete specific steps when creating the OLAP model. The column with which you associate the alias or UDA *must* be fully defined, and all alias or UDA information *must* relate to the fully defined column.

If you want to associate an alias or UDA with the parent column of a recursive table, the parent column *must* be fully defined. A recursive table parent column is fully defined when the parent column contains every value (every member proposed for the Essbase hierarchy). Thus the parent column contains the lowest-level value in the hierarchy with a NULL value in the child column. In Table 1, the GEO_PARENT column is fully defined because the GEO_PARENT column contains the lowest-level value, 01010, with a NULL child in the GEO_CHILD column.

Table 1: Fully Defined Parent Column

GEO_PARENT	GEO_CHILD
USA	East
East	Maine
Maine	Bangor
Bangor	01010
01010	<NULL>

If you want to associate an alias or UDA with the child column in a recursive table, the child column *must* be fully defined. A recursive table child column is fully defined when the child column contains every value (every member proposed for the Essbase hierarchy). Thus the child column contains the highest-level value in the hierarchy, with a NULL value in the parent column.

In Table 2, the GEO_CHILD column is fully defined because the GEO_CHILD column contains the highest-level value, USA, with a NULL parent in the GEO_PARENT column.

Table 2: Fully Defined Child Column

GEO_PARENT	GEO_CHILD
<NULL>	USA
USA	East
East	Maine
Maine	Bangor
Bangor	01010

For more information on the specific steps that you must perform, refer to the Essbase Integration Services Console Help.

Removing Unions

Essbase Integration Server does not generate SQL for unions. A *union* is a type of join that combines the results of two SELECT statements. A union is often used to merge lists of values that are contained in two tables. If the source tables use unions, you must create a view (or views) of the data that does not use the unions before you can start to work with the data in Essbase Integration Services. For more information on unions, see the RDBMS documentation.

Transposing Columns and Rows

If you want to transpose columns and rows so that you can more easily transition data from various data sources to an Essbase database, transpose the columns before you start to work with the data in Essbase Integration Server. For example, if you want to create multiple Essbase members from a single relational database column, you can create a new table or view that transposes row values to column values.

Consider the following example in which you want to create multiple Essbase measures (Init_Sales, Subsequent_Sales, and Return_Sales) from a single database column (SALESTYPE).

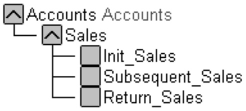
The SALESACTUALS table contains the following columns:

Table 3: SALESACTUALS Table

PRODID	MKTID	TIMEID	INITSALES	SALESTYPE	PRODID
100	2	01-01-2000	100.00	Sales	100
100	2	01-02-2000	10.00	Returns	100
100	2	01-03-2000	50.00	Subsequent	100
100	2	01-04-2000	20.00	Returns	100
100	2	01-01-2000	100.00	Sales	100
:	:	:	:	:	:

You want to create the Essbase outline in the hierarchy illustrated in Figure 4:

Figure 4: Accounts Hierarchy



```
Accounts Accounts
└─ Sales
   ├─ Init_Sales
   ├─ Subsequent_Sales
   └─ Return_Sales
```

You want each SALESTYPE value to form the Essbase member: SALES to form the Init_Sales member, RETURNS to form the Return_Sales member, and SUBSEQUENT to form the Subsequent_Sales member. To achieve these goals, create a view or a new table that transposes row values into column values. This example produces the following table:

Table 4: View of SALESACTUALS Table with Transposed Columns

PRODID	MKTID	TIMEID	INITSALES	RETURNS	SUBSEQUENT
100	2	01-01-2000	100.00	0.00	0.00
100	2	01-02-2000	0.00	10.00	0.00
100	2	01-03-2000	0.00	0.00	50.00
100	2	01-04-2000	0.00	20.00	0.00
:	:	:	:	:	:

The following example of Oracle SQL defines the view shown in Table 4:

```
)
Create view SalesActuals_vw as
    (Select ProdId, MktId, TimeId,
         decode (SalesType, 'Sales', Sales, 0) "InitSales",
         decode (SalesType, 'Returns', Sales, 0) "Returns",
         decode (SalesType, 'Subsequent', Sales, 0)"Subsequent"
            from SalesActuals
)
```

For more information on using SQL to define views, see the documentation for the RDBMS that you use.

When you have defined a new table or view, use it to create an OLAP model.

Denormalizing Source Data Tables

If data is highly normalized, it may not map clearly to an OLAP model. Normalized data is appropriately grouped and does not include redundant data. Even though you can use normalized source tables to create an OLAP model by specifying joins (refer to the Essbase Integration Services Console Help), it may be more efficient to create a new table of denormalized data in the RDBMS.

Consider the following example. In the first three tables, data is highly normalized so that redundant data is minimized:

Table 5: Normalized Product Family Data

FAMILYID	FAMILYDESC
100	Colas
200	Root Beer

Table 6: Normalized Product Data

PRODID	FAMILYID
100-10	100
100-20	100

Table 7: Normalized Product Description Data

PRODID	PRODDESC
100-10	Cola
100-20	Diet Cola

The following table shows the denormalized data in one table:

Table 8: Denormalized Product Data

FAMILYID	FAMILYDESC	PRODID	PRODDESC
100	Colas	100-10	Cola
100	Colas	100-20	Diet Cola
100	Colas	100-30	Caffeine Free Cola
100	Colas	100-10	Cola

If you use the denormalized data source, Essbase Integration Server does not need to compute the joins when it builds the OLAP model.

Joining Tables

In the data source tables, set up primary and foreign keys to join each of the following tables:

- Join tables that form the fact table. See “Selecting Tables for the Fact Table” on page 43.
- Join tables within each dimension branch. See “Selecting Tables for Dimensions” on page 44.
- Join dimension tables to the tables in the fact table.

When you join tables in the RDBMS, Essbase Integration Server automatically detects the join when it builds the OLAP model.

Creating Indexes

Essbase Integration Server detects indexes (including bitmapped indexes) that you have defined on the source tables and uses them to create an Essbase outline and to load data. *Indexes* are pointers that are logically arranged by the values of a key. Indexes optimize access to relational data. Bitmapped indexes are specialized indexes that may improve performance during analysis of numeric data.

You can significantly improve performance by defining indexes on the appropriate data in the source tables.

Consider taking the following actions to improve performance:

- Define indexes on columns that you use to filter data in the RDBMS source or in the OLAP model. For example, if the source database contains columns for city and state and you filter on city or state (for example, `SELECT * FROM Region WHERE State = Ca%`), then index the columns on which you are filtering (in this example, the column State). If you are creating alias names and you filter on alias names, index the column containing alias names. For more information on alias names, see the *Essbase Database Administrator's Guide*.
- Define bitmapped indexes on numeric data that you use to filter the database. For example, if you filter on sales values (for example, `SELECT Product FROM ProdSales GROUPBY Product ORDERBY ProdId HAVING SUM(Sales)>15000`), then consider defining a bitmapped index on sales values. Most RDBMSs support bitmapped indexes. For more information on bitmapped indexes, see the documentation for the RDBMS that you are using.

Transforming Data

You may need to transform some of the data, for example, to change date formats or to ensure that data is described consistently.

Assume that you want to create an OLAP model that combines data from the sales databases and data from the financial databases. If the sales databases specify New York as New_York and the financial databases specify New York as NY, you can transform NY to New_York in the staging area (see "Deciding Whether to Create a Staging Area" on page 30) without changing the data in the original financial databases.

You can do some data transformations in OLAP models and metatoutlines. For a complete list of available transformations, refer to the Essbase Integration Services Console Help. If, however, you must do significant transformations on the data, you may want to use a data transformation tool before you use the data in Essbase Integration Server.

Transformations that you must perform in the RDBMS, and not in Essbase Integration Services, include these:

- The most frequently accessed transformations

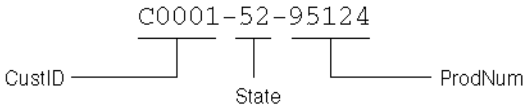
  For example, assume that you want the Essbase database outline to have members for Year, Qtr, and Month and that the data source table contains a column called TRANSDATE. TRANSDATE holds the transaction date for each row. Transform the data ahead of time, creating a new source table that contains the columns YEAR, QTR, and MONTH. The new columns contain the data transformed from the TRANSDATE column. You can then index the YEAR, QTR, and MONTH columns to improve performance.

  Note: Many RDBMSs ignore indexes on columns that have transformations. You may need to transform the data and create a physical table with new columns that can be indexed.

- Transformations to take the intelligence out of key columns

  Some key columns may include categories of information that you want to split out before using the data in Essbase Integration Server. For example, if the source table has a PRODID column containing product identification information, transform the data in the RDBMS so that it maps to the data categories that you want to display in the Essbase outline. In Figure 5, for example, the categories are CUSTID, STATE, and PRODNUM.

Figure 5: Transforming a Product Code Stored in a PRODID Column



Deciding Which Tables Are Available to OLAP Model Creators

When you connect to one or more source databases, Essbase Integration Services Console displays in the left frame all tables to which the RDBMS user ID provides you access (SELECT permission).

For all RDBMSs, if you have partial access to a table, Essbase Integration Server displays the whole table. (For example, if you have SELECT permission for only half of the columns in a table, you can see the whole table.) However, in the OLAP model, do not use a column to which you do not have SELECT permission. Essbase Integration Server cannot filter, load data, or load members in a column for which you do not have SELECT permission.

Note: For DB2, Essbase Integration Server displays all tables that are in the database, regardless of whether you have access to them. If, however, if you do not have access (SELECT permission) to a table, you cannot view the columns in the table or use the table to build an OLAP model.

For added security and ease of administration, you may want to define which tables the users can see by taking one or more of the following actions:

- Create specific RDBMS user IDs for users who are creating OLAP models.
- Create views of some data and provide users who are creating OLAP models with access to the views rather than access to the original tables.
- Use conditions to restrict a view so that users who are creating an OLAP model can see only a subset of the view. For example, you can restrict the view to show data only for the current year although the table contains data for three years. For more information on adding restrictive conditions, see the documentation for the RDBMS that you use.

Note: When you connect to the source database, Essbase Integration Server provides read-only access to it, regardless of the access provided by the RDBMS user ID. For example, if the RDBMS user ID has read-write access, you have only read-only access when creating OLAP models.

Selecting Tables for the Fact Table

A fact table is a logical container for the relational tables that define the data values (measures) for each dimension intersection in an OLAP model. For more information, refer to the Essbase Integration Services Console Help. The type of Essbase database that you want to create determines which source tables you must include in the fact table. See "Defining User Needs" on page 28.

When you select source tables to include in the fact table, consider both how the data is distributed among the tables and how the data relates to the dimensions in the Essbase database.

Select a table or tables that contain all the measures data that Essbase end users want to see. For example, if the relational database contains the following ORDERS and ORDERDETAILS tables, you must include both tables if you want the fact table to include data from both tables.

Table 9: ORDERS and ORDERDETAILS Tables for the Fact Table

Orders	**ORDERID**	**TRANSDATE**	**SHIPID**	**EMPTYID**

Order Details	**ORDERID**	**PRODID**	**NUMUNITS**	**UTPRICE**

Select a table or tables that represent all the dimensions (data categories) on which the Essbase end users want to analyze data. For example, if end users need to analyze product sales by time, by geography, and by sales channel, select a table or tables that contain key columns for each of these dimensions.

Table 10: Sales Table for the Fact Table

	PRODID	**SALES**	**DATESOLD**	**CITY**	**CHANNEL**	**UNTSSOLD**	**...**	**...**
Sales								:

Note: You can improve performance by setting up primary and foreign keys, and joining the tables that form the fact table. See "Joining Tables" on page 39.

Selecting Tables for Dimensions

A dimension is a data category. Typical dimensions are Product, Geography, and Time. A dimension table is a logical container within an OLAP model. A dimension table includes one or more relational tables that define a potential Essbase dimension. A dimension table can join to other dimension tables, forming a *dimension branch*.

The data that you want to be available for analysis in the Essbase application that you create determines which source tables you use to create dimensions. See "Defining User Needs" on page 28. When you select source tables for each dimension, select a source table or tables that include the maximum amount of information about the data category (dimension). For example, if you have the following PRODUCTS and ORDERS source tables, select the PRODUCTS table to form the Products dimension.

Table 11: ORDERS and PRODUCTS Tables for the Dimension Table

Products	**PRODID**	**PRODNAME**	**SIZE**	**COLOR**

Orders	**PRODID**	**PRODNAM**	**ORDERID**	**DATE**

The ORDERS table contains product information for only those products that have been sold. The PRODUCTS table contains data for all products. Thus, Essbase end users can use the PRODUCTS table information to analyze products that are or are not selling.

Creating Aliases for Dimension and Member Names

An *alias* is an alternative name for a member or dimension in an Essbase outline. An alias is often a more easily identifiable product name or product description for a column in a relational data source. For example, in the Essbase Sample Basic database, the 200-20 member in the Product dimension has the alias Diet Root Beer.

If the source data contains information that you want to use to create aliases in an Essbase outline, include the relevant data as a column in the associated OLAP model.

When you create a metaoutline, you can retrieve alias information from only one column in the OLAP model. If the alias information in the source data is in more than one column, complete one of the following data preparation procedures:

- In the RDBMS, create a view or a new table to *concatenate* (and, if necessary, transform) the alias information into a single column. Creating a view or new table may be the easiest method, especially if the alias information is stored in three or more columns.
- In the OLAP model, concatenate (and, if necessary, transform) the columns that contain the alias information. For more information on concatenating and transforming columns, refer to the Essbase Integration Services Console Help.
- In the OLAP model, use the pass-through feature to run a procedure that uses RDBMS logic to retrieve alias information from the column or columns that you specify. For example, you can use the pass-through feature to run a procedure in DB2. The DB2 procedure uses CASE logic to retrieve alias information from two different columns; if a field in one column is empty, the procedure tells the RDBMS to retrieve the alias information for that field from a different column.

  For more information on pass-through transformations, refer to the Essbase Integration Services Console Help. For more information on procedures and CASE logic, see the RDBMS documentation.

Note: You can associate one alias table for each Essbase database that you create using Essbase Integration Services.

Creating Time and Accounts Dimensions

In an OLAP model, you can create time and accounts dimensions that carry over directly to the Essbase outline as the dimensions tagged as time and accounts. The *time dimension* table contains date-related columns from the relational data source. The *accounts dimension* table contains measures columns that duplicate the columns contained in the fact table.

Preparing Data for Time Analysis

If the business application requires time-related analysis, such as examining trends over time or making seasonal sales comparisons, include a time dimension in the Essbase outline. A time dimension includes members for the time periods that you report on; for example, monthly, quarterly, and yearly sales.

To support a time dimension, the data source should include the following elements:

- The data source should include one or more time-related columns associated with measures data. For example, if you want to report on sales per month and per quarter, one of the tables that you include in the fact table needs a date column to reflect sales by month and quarter.
- If needed, the data source should include one or more tables containing user-defined calendars. A user-defined calendar maps the time-related column in the fact table to a specific business calendar.

Associating Time Data with Measures Data

To enable Essbase Integration Server to map the data that you are measuring (such as sales or costs) to specific time periods (such as month, quarter, or year), the source data for the fact table must contain one or more time-related columns. For example, from a transaction date column, you can determine the day, week, and month in which the associated transaction occurred. Or, the transaction data can include separate columns for each time period. These time periods are potential members of the dimension tagged as time.

Working with Summary Data

If the measures columns in the source data contain summary data, the time-related column must indicate the time period for which the data is summarized. For example, assume that the SALESINVACT table contains measures columns SALES and COGS; a date-related column, TRANSDATE; and STATE and PRODCODE columns that categorize by sales date, state, and product code.

Table 12: TBC_MD Data in the SALESINVACT Table

STATE	PRODCODE	TRANSDATE	SALES	COGS	...
IL	100-10-C001-S0002-P001	Jan 4 2000 12:00AM	672.00	217.00	
IL	100-10-C001-S0002-P001	Feb 4 2000 12:00AM	241.00	70.00	
IL	100-10-C001-S0002-P001	Mar 3 2000 12:00AM	287.00	84.00	
IL	100-10-C001-S0002-P001	Apr 7 2000 12:00AM	295.00	85.00	
IL	100-10-C001-S0002-P001	May 3 2000 12:00AM	702.00	207.00	

Each row in the SALESINVACT table summarizes product sales and costs for a particular date for customers from a particular state. Instead of summarizing data by day, a row can summarize data by other periods; for example, by week or by month.

Formatting the Date

Essbase Integration Server supports three *data types*: string, numeric, and datetime. The basic Open Database Connectivity (ODBC) data types map to four data types within Essbase Integration Services: text, numeric, Boolean, and datetime. The datetime data type is a timestamp that includes the year, month, day, and time of day.

The data type of the source data column that contains time-related data does not matter. For example, the date can be a timestamp in a column with a datetime data type, or it can be keyed data in a numeric column (for example, `09232000` or `20000923`). A string column can contain dates with or without separating characters such as slashes or periods (for example, `09/23/2000` or `23.09.2000 or 09232000`). Portions of the date (such as the day, week, month, or quarter) can be included in the source data as separate string or numeric columns.

It is important that the values in the time-related columns can be mapped to the time periods that you want in the time dimension. If dates are stored as datetime data types and you want quarterly totals, Essbase Integration Server can determine the quarter. If dates are stored in numeric or string columns and you want quarterly totals, you must have an explicit column containing the associated quarter for the data that you want to consolidate.

Working with Data in Similar Time Periods

In certain applications, such as general ledger and legacy systems, tables may be organized so that columns identify similar time periods. For example, a single row may include all months of a year. In this case, each subsequent row includes a measure category as a data value; for example, a specific account number.

Table 13: Example of a Table Where Columns Define the Time Periods

PRODCODE	STATE	ACCOUNT	JAN	FEB	MAR	...
100-10-C001-S002-P001	AZ	Sales	672.00	241.00	287.00	
100-10-C001-S002-P001	AZ	COGS	403.00	132.00	177.00	
100-10-C001-S002-P001	CA	Sales	401.00	143.00	378.00	
100-10-C001-S002-P001	CA	COGS	260.00	101.00	226.00	

To create a dimension of the type time, Essbase Integration Server requires that all time identifiers be in one column and that accounts identifiers be in separate columns, as shown in Table 14. If, for example, data is stored with each month as an individual column, as shown in Table 13, you should restructure the data. Individual months should be data values in a MONTH column, and the measure categories, such as SALES and COGS, should be separate columns.

Table 14: Example of a Table Where Time Periods Are Stored as Data Values

PRODCODE	STATE	MONTH	SALES	COGS
100-10-C001-S002-P001	AZ	1	672.00	403.00
100-10-C001-S002-P001	AZ	2	241.00	132.00
100-10-C001-S002-P001	AZ	3	287.00	177.00

Table 14: Example of a Table Where Time Periods Are Stored as Data Values

PRODCODE	STATE	MONTH	SALES	COGS
100-10-C001-S002-P001	CA	**1**	**401.00**	**260.00**
100-10-C001-S002-P001	CA	**2**	**143.00**	**101.00**
100-10-C001-S002-P001	CA	**3**	**378.00**	**226.00**
100-10-C001-S002-P001	AZ	1	672.00	403.00

You can write a program to convert the data to a table that uses the appropriate format or use an RDBMS to transpose the columns (see “Transposing Columns and Rows” on page 36).

Accessing Tables in the OLAP Metadata Catalog

If you log on with a particular user ID when you create OLAP Metadata Catalog and then log on with a different user ID, you cannot access the tables in the OLAP Metadata Catalog unless you create an alias for the user ID (Sybase, SQL Server) or synonyms for the tables (DB2, Oracle). For more information, see the *Essbase Integration Services Installation Guide*.

You must perform a different procedure when time measures are stored as columns that include accounts information, as described in the Essbase Integration Services Console Help.

Index