Hyperion Essbase™

Version 5

Database Administrator's Guide Volume II

HYPERION®

P/N: 06071-500200-000

US Patent Number 5,359,724

Hyperion Solutions Corporation
1344 Crossman Avenue
Sunnyvale, CA 94089

Publication Number: 40-224-9030

Printed in the USA

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

Table of Contents

Volume I

Introduction

Part I Designing Hyperion Essbase Applications

Part IV Loading Data

Volume II

Part VI Reporting on Your Data

Part VII Managing Multidimensional Essbase Data Storage

Chapter 39 Introducing the Essbase Storage Manager

Part VI Reporting on Your Data

Part VI describes how to create reports on the data in Hyperion Essbase OLAP Server databases, including how to create reports quickly, develop report scripts, optimize reports, and export data to other programs. Part VI contains the following chapters:

- Chapter 34, Quick Start to Report Scripts, introduces you to the basic concepts behind reports and describes how to create them using the Report Writer.
- Chapter 35, Developing Report Scripts, describes how to create complex report scripts, including page layout and formatting information, how to select and sort members, how to restrict and order data values, how to convert data to a different currency, and how to generate reports using the C, Visual Basic, and Grid APIs.
- Chapter 36, Examples of Report Scripts, contains detailed examples of report scripts.
- Chapter 37, Optimizing Your Reports, describes ways to generate your reports more quickly.
- Chapter 38, Copying Data Subsets and Exporting Data to Other Programs, describes how to move data from Hyperion Essbase OLAP Server databases to other programs by extracting an output file of the data to move using the Report Writer.

Chapter 34 Quick Start to Report Scripts

A Hyperion Essbase Application Manager report lets you retrieve formatted summaries from a database.

There are several ways that you can report on the data in your database:

- Use the Application Manager's Report Writer to create a report script and run your report, as explained in this chapter.
- Generate database reports through a spreadsheet. You can use report commands in a spreadsheet in Free-Form mode or in Template Retrieval mode. For information on reporting through your spreadsheet interface, see your Essbase *Spreadsheet Add-in User's Guide*.
- Use Hyperion Essbase APIs. See Chapter 35, Developing Report Scripts, for more information about using report APIs or the online *API Reference* in your `DOCS` directory for syntax and technical descriptions.
- Use third-party reporting tools such as Crystal Info for Essbase (sold separately from Essbase).

Use the Report Writer when you need to create large-scale reports consisting of many pages of multidimensional data. Reports on this scale can often exceed the capabilities of even the most robust spreadsheet. You can use the Application Manager to create report scripts and run reports, or you can schedule report scripts to run in batch mode, using the ESSCMD command-line interface. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Report Writer commands let you define formatted reports, export data subsets from an Essbase database, and produce free-form reports.

This chapter provides fundamental information about reports and report scripts, including:

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

Creating a Simple Report Script

When you combine report commands that include page, row, and column dimension declarations with selected members, you have all the elements of a simple report script.

The following step-by-step example of a report script specifies these elements, dimensions, and member selection commands. It includes comments, which document the behavior of the script, and the **!** output command. This example is based on the Sample Basic database, which is supplied with your Hyperion Essbase OLAP Server installation.

To create a simple report script:

1. Start Application Manager, and connect to your Essbase server.
2. Select the Sample application and the Basic database, and click the Report Scripts  button.

   If you do not have the Sample Basic database installed, contact your Essbase administrator.
3. Click the **New** Button to open the Report Editor.

4. Type the following information, with the exception of the commented (//) lines, which are for your reference:

```
// This is a simple report script example
// Define the dimensions to list on the current page, as below
<PAGE (Market, Measures)

// Define the dimensions to list across the page, as below
<COLUMN (Year, Scenario)

// Define the dimensions to list down the page, as below
<ROW (Product)

// Select the members to include in the report
Sales
<ICHILD Market
Qtr1 Qtr2
Actual Budget Variance
<ICHILD Product

// Finish with a bang
     !
```

5. Choose File | Save, and type `Myrept1` for the report script object name, and save it on the server (the default).
6. Choose Report | Run.

When you run this report against the Sample Basic database, the script produces the following report:

```
                                      East Sales

                           Qtr1                       Qtr2
                  Actual   Budget Variance   Actual   Budget Variance
                ======== ======== ======== ======== ======== ========

100                9,211    6,500    2,711   10,069    6,900    3,169
200                6,542    3,700    2,842    6,697    3,700    2,997
300                6,483    4,500    1,983    6,956    5,200    1,756
400                4,725    2,800    1,925    4,956    3,200    1,756
  Product         26,961   17,500    9,461   28,678   19,000    9,678
```

West Sales

	Qtr1			Qtr2		
	Actual	Budget	Variance	Actual	Budget	Variance
100	7,660	5,900	1,760	7,942	6,500	1,442
200	8,278	6,100	2,178	8,524	6,200	2,324
300	8,599	6,800	1,799	9,583	7,600	1,983
400	8,403	5,200	3,203	8,888	6,300	2,588
Product	32,940	24,000	8,940	34,937	26,600	8,337

South Sales

	Qtr1			Qtr2		
	Actual	Budget	Variance	Actual	Budget	Variance
100	5,940	4,100	1,840	6,294	4,900	1,394
200	5,354	3,400	1,954	5,535	4,000	1,535
300	4,639	4,000	639	4,570	3,800	770
400	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Product	15,933	11,500	4,433	16,399	12,700	3,699

Central Sales

	Qtr1			Qtr2		
	Actual	Budget	Variance	Actual	Budget	Variance
100	9,246	6,500	2,746	9,974	7,300	2,674
200	7,269	6,800	469	7,440	7,000	440
300	10,405	6,200	4,205	10,784	6,800	3,984
400	10,664	5,200	5,464	11,201	5,800	5,401
Product	37,584	24,700	12,884	39,399	26,900	12,499

Market Sales

	Qtr1			Qtr2		
	Actual	Budget	Variance	Actual	Budget	Variance
100	32,057	23,000	9,057	34,279	25,600	8,679
200	27,443	20,000	7,443	28,196	20,900	7,296
300	30,126	21,500	8,626	31,893	23,400	8,493
400	23,792	13,200	10,592	25,045	15,300	9,745
Product	113,418	77,700	35,718	119,413	85,200	34,21

For information about Report Writer command syntax and definitions, see the *online Technical Reference* in your `DOCS` directory. If you are using the Report Editor, choose the Help menu item to consult the Application Manager Help.

To create report scripts in ESSCMD:

Use the REPORTLINE command to execute a single report line. For example,

```
REPORTLINE "<DESC Year !"
```

Use the REPORT command to execute one or more report lines. For example,

```
REPORT <IDESC Year <ICHILD Market
!
```

For information about ESSCMD, see Chapter 43, Performing Interactive and Batch Operations Using ESSCMD.

Parts of Report Scripts and Reports

The Essbase Report Writer consists of three main components:

- *Report Editor* is an ASCII text editor that you use to write the report script. The Report Editor features a text editing window and customized menus. Saved report scripts have the file extension `.REP`.
- *Report Extractor* retrieves the data information from the Essbase database when you run a report script.
- *Report Viewer* displays the complete report. Saved reports have the file extension `.RPT`.

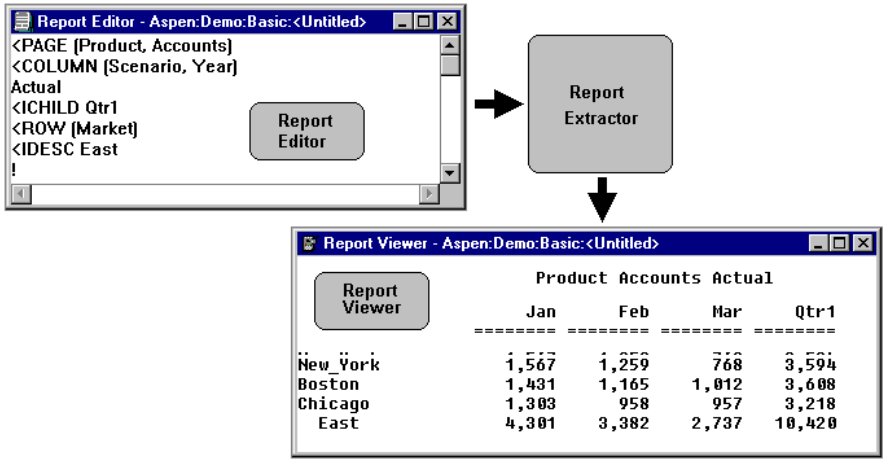

Figure 34-1, Report Writer Components

How the Report Extractor Retrieves Data

The Report Extractor processes the report script and retrieves data in the following order:

1. Composes the member list, based on all possible member combinations. For example, the following command retrieves member East and all of its descendants:

   ```
   <IDESCENDANTS East
   ```

2. Applies member restrictions. For example, the following command refines the member selection:

   ```
   <LINK
   ```

3. Orders the member output. For example, the following command determines the order in which members are sorted:

   ```
   <SORT
   ```

4. Extracts data from the following areas:

   - Local regions
   - Partitioned regions
   - Dynamically calculated data

5. Restricts data. For example, the following command suppresses the display of all rows that contain only missing values:

   ```
   {SUPPRESSMISSINGROWS}
   ```

6. Sorts data. For example, the following command returns rows with the highest values of a specified data column:

   ```
   <TOP
   ```

7. Formats output. For example, the following command skips one or more lines in the final output report:

   ```
   {SKIP}
   ```

The order in which the Report Extractor retrieves data is important when using complex extraction and formatting commands. For example, because the Extractor restricts data (step 5) before sorting data (step 6), if you place conditional retrieval commands in the wrong order, the report output results could be unexpected. Be aware of the data retrieval process when designing your report scripts.

Parts of a Report

Understanding the parts of a report is essential as you plan and design your own reports. A typical report is composed of the following parts:

- *Page Headings* list dimensions represented on the current page. All data values on the page have the dimensions in the page heading as a common attribute.

  ```
  <PAGE (Market, Measures)
  ```

- *Column Headings* list members across a page. You can define columns that report on data from more than one dimension, which results in *nested column headings*.

  ```
  <COLUMN (Year, Scenario)
  ```

- *Row Headings* list members down a page. You can define a member list that includes rows from more than one level within a dimension or from more than one dimension. The rows are indented below the dimension name.

  ```
  <ROW (Product)
  ```

- *Titles* contain user-defined text, date and time stamp, the user ID of the person running the report, page numbers, the name of the source database, or any other descriptive information. Titles are user-generated and optional, whereas page, column, and row headings are automatically generated, because they are necessary to clearly describe the data on the report page.

  ```
  { STARTHEADING
  TEXT   1 "Prepared by:"
        14 "*USERNAME"
         C "The Electronics Club"
        65 "*PAGESTRING"
  TEXT  65 "*DATE"
  SKIP
  ENDHEADING }
  ```

- *Data values* are the values contained in the database cells; they are the lookup results of member combinations or the results of calculations when the report is run through the Report Extractor. Each data value is the combination of the members in the page heading, column heading, and row name.

All data values in a row share the attributes of the row name(s) of that row. A report can have zero or more row name dimensions, each of which produces column of row names, with the innermost row name column cycling the fastest.

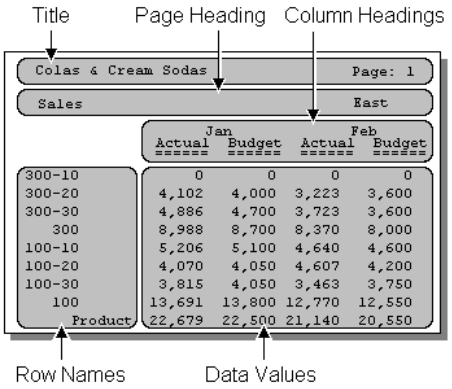

Figure 34-2, Elements of a Typical Report

Parts of a Report Script

A report script consists of a series of Essbase Report Writer commands, terminated by the bang (!) report output command.

You can enter one or more report scripts in a report script file. A report script file is an ASCII text file that you create with the Essbase Report Editor or any text editor.

To build a report script, enter commands that define the layout, member selection, and format in the Essbase Report Editor.

The commands in Report Writer perform two functions: data extraction and formatting.

- *Extraction commands* deal with the selection, orientation, grouping, and ordering of raw data extracted from the database. These commands begin with the left angle bracket (<).
- *Formatting commands* allow for customization of the report format and appearance, the creation of new columns, and calculation of columns and rows. These commands are generally contained within curly braces ({}), although some do begin with the left angle bracket (<).

Additionally, the *bang character* (!) terminates a series of commands and requests information from the database. You must terminate a report script with a bang character, or you can use several bang characters within the script. See Chapter 37, Optimizing Your Reports, for more information about the ! character.

See the online *Technical Reference* in your `DOCS` directory for detailed information about the various report commands that you can use.

Planning a Report

Report design is an important part of presenting your information. Designing a report is easy if you include the proper elements and arrange information in an attractive, easy-to-read layout.

To plan your report:

1. Consider your reporting needs and the time it will take to generate the report.
2. Make a rough sketch of the report that includes:
   - Report layout
   - Number of columns
   - Members to include
   - Titles, if applicable
   - Format of the data values
3. Review the sketch; if you need to add additional data or formatting to the report, this is often apparent at this stage.
4. Determine ways to optimize the run time of the report. See Chapter 37, Optimizing Your Reports, for suggestions about optimizing your report script.

Note: As you plan your report, minimize your use of numeric row names. To avoid ambiguity, give the rows names that describe their content.

Security and Multi-User Issues

Because you run the Report Editor from the Application Manager menu, you must have access to the Application Manager in order to use the Report Editor to create or modify a report script. You can also use any text editor to create script files. If you use the Application Manager's Report Editor, it lets you create and modify report scripts stored on your desktop machine, as well as the Essbase server. To modify report scripts stored on the server, you must have Application Designer or Database Designer access.

Essbase supports concurrent, multi-user database access. As in most multi-user environments, Essbase protects your critical data with a security system. Users can read or update data only if they have the correct privileges.

When you execute a report script, the Essbase security system verifies that you have Read or higher access level to all data members specified in the report. In a filtering process identical to the one for retrieving members into a spreadsheet, Essbase filters any member from the output for which you have insufficient privileges.

To users who are only reporting data, locks placed by other users are transparent. Even if a user has locked and is updating part of the data required by your report, the lock does not interfere with your report in any way. The data in the report reflects the data in the database at the time you run the report. Running the same report later reflects any changes made after your last report ran.

See Chapter 16, Managing Security at Global and User Levels, for more information about the Essbase security system.

Creating and Editing Report Scripts

You can create your report script in the Report Editor or with any ASCII text editor. Once you create your script, you can choose to save it to either the Essbase server or your desktop machine.

You can modify your script using text editing features that let you cut, copy, paste, find, and replace. You can also undo most commands and changes with the Edit | Undo command.

The Report Editor uses familiar text editing commands, such as Edit | Cut, Edit | Copy, and Edit | Paste. These commands are available from the Application Manager menu or through accelerator key (hot key) combinations.

Creating New Report Scripts

Before you can create a report script, you must connect to an Essbase server and open a database.

To create a new report script:

1. Choose File | New | Report Script from the Application Manager menu.

   The Report Editor appears.

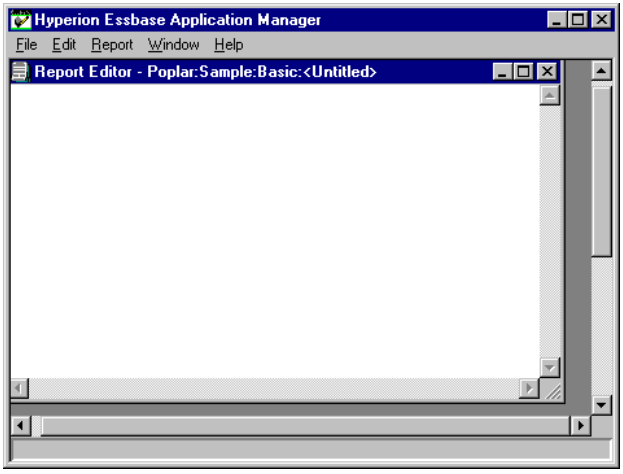

Figure 34-3, New Report Editor Window

2. Begin typing your report script.

   The name of the report script is `Untitled` when it first appears. When you save the script, the Application Manager prompts you for a file name.

Saving Report Scripts

You can save a report script as:

- A file on your client machine.
- An object on the Essbase server. If you want other users to have access to the report script, you need to save it on the Essbase server. You can associate the script object with:
  - An application and all the databases within the application, which lets you run the script from any of the databases in the application.
  - A database, which lets you run the script from the specified database.

Report scripts have a `.REP` extension by default. If you run a report script from the Application Manager it must have a `.REP` extension.

To save a report script:

1. Choose File | Save.
2. Choose to save the script on the Essbase server or to your desktop machine.

To save the script on the Essbase server:

1. Click the **Server** button in the **Location** box.
2. Type the name to give to the file in the **Object Name** box.
3. Choose the server where you want to save the script in the **Server** list box.
4. Choose the application where you want to save the script in the **Application** list box.
5. Choose the database where you want to save the script in the **Database** list box, or choose **(all dbs)** to make the report script available to all databases within the chosen application.
6. Choose **Report Scripts** in the **List Files of Type** box.
7. Click OK.

To save the script on your desktop machine:

1. Type the file name in the **Object Name** box.
2. Choose the application where you want to save the script in the **Application** list box.
3. Choose the database where you want to save the script in the **Database** list box, or choose **(all dbs)** to make the report script available to all databases within the chosen application.

4. Choose **Report Scripts** in the **List Files of Type** box.
5. Click OK.

   By default, the file is saved to the `\ESSBASE\CLIENT\SAMPLE` directory on your desktop machine.

Opening Report Scripts from the Application Manager

To open a report script from the Application Manager:

1. From the Application Manager menu, choose File | Open.

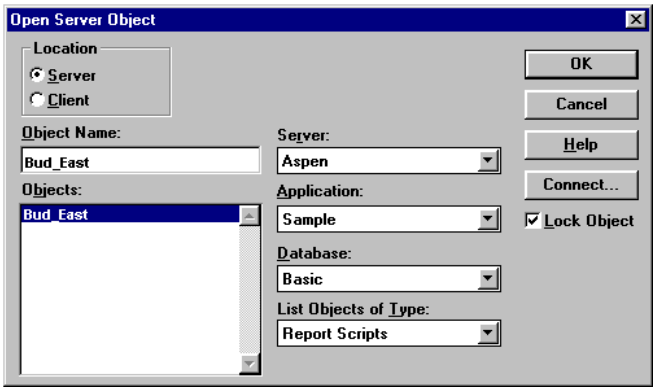

Figure 34-4, Open Server Object Dialog Box

2. From the **Location** group, select the storage location for the report script.
   - Click the **Server** option to search the Essbase Server.
   - Click the **Client** option to search your desktop machine. The dialog box title changes to **Open Client Object**.
3. From the **List Objects of Type** list box, choose **Report Scripts**.

   The **Objects** list box displays the names of all available report scripts in the selected application and database.

Note: To choose another application to search, select it from the **Application** list box. To choose another database to search, select it from the **Database** list box.

4. You can also locate a locally stored report script through the Windows file system. Click the **File System** button in the **Client Object** dialog box.

   The **Open Client File** dialog box appears. The file name mask `*.REP` appears in the **File Name** text box and all report script files in the current directory appear in the list box.

5. To search other directories, use the **Directories** list box.
6. When the report script you want to open appears in the **Objects** list of the **Open Server Object**, **Open Client Object**, or **Open Client File** dialog box, you can:
   - Click the report script name. When the name appears in the **Object Name** text box, click OK.
   - Enter the name in the **Object Name** text box and click OK.
   - Double-click on the report name in the **Objects** list.

   The Application Manager starts the Report Editor and loads the selected report script.

Opening Report Scripts from an Application or Database Directory

You can open report scripts stored in applications and databases from the application directory.

To open a report script from an application folder:

1. Double-click the application name to open the application folder and select a database from the **Databases** list that contains a report script, or select **(all dbs)** from the **Databases** list if you want to open a report script stored at the application level.
2. Click the **Report Scripts** button.
3. From the **Reports** list, choose the report script you want to open.
4. Click **Open** to start the Report Editor and load the selected report script.

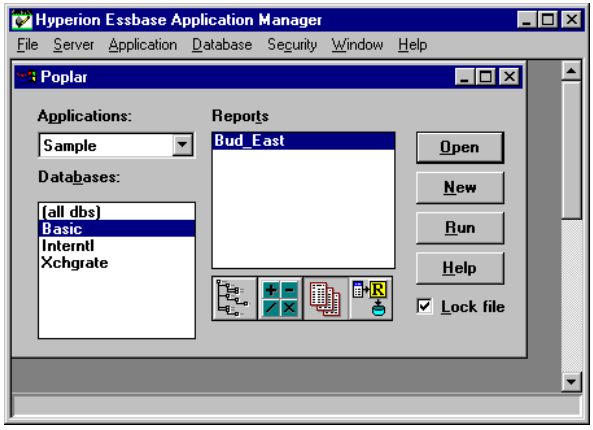

Figure 34-5, Database with a Report Script

Finding Text

You can use the Find menu command to search for text in the Report Editor. Find locates all occurrences of the specified text in your report script.

To find text in a report script:

1. Choose Edit | Find to open the **Find** dialog box.

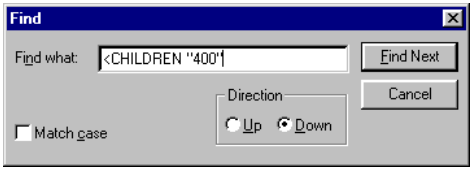

Figure 34-6, Find Dialog Box

2. Type the text you want to find in the **Find what** text box.
3. Use the **Direction** group box to choose the search direction.
   - Click the **Down** option button to search from the cursor position forward to the end of the report script.
   - Click the **Up** option button to search backward from the cursor position to the beginning of the report script.

   For example, Figure 34-6 shows a dialog box that finds the next occurrence of `<CHILDREN "400"` between the current cursor position and the end of the report script.
4. If you want the search to be case-sensitive, check the **Match case** check box. The search matches uppercase and lowercase letters exactly as they appear in the **Find what** text box.

   In the example, the **Match case** check box is checked, so the search finds `<CHILDREN "400"` but not `<Children "400"`.
5. Click **Find Next**.

   If the search finds the text you entered in the **Find what** text box, it highlights the text in the Report Editor.
6. To edit the selection, click Cancel to close the **Find** dialog box.

7. To search for other occurrences of the text, click **Find Next** again.

   When Application Manager reaches the end of the report script, or if it cannot find the selected text to replace, the following dialog box appears:

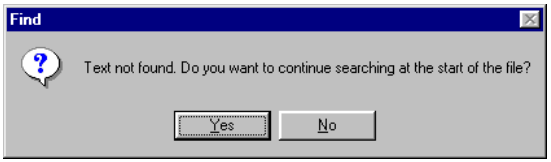

Figure 34-7, Continue Find Dialog Box

   - Click **Yes** to continue searching the file.
   - Click **No** to close the dialog box and return to the **Find** dialog box.

8. When you are finished, click Cancel to close the **Find** dialog box.

Replacing Text

The Replace menu command replaces one or all instances of specified text with different text.

To replace text in a report script:

1. Choose Edit | Replace to open the **Replace** dialog box.

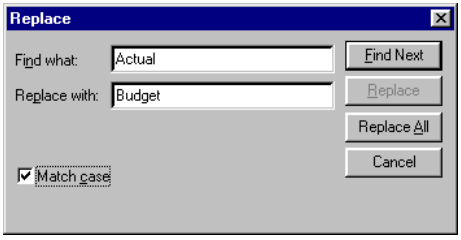

Figure 34-8, Replace Dialog Box

2. Enter the text you want to find in the **Find what** text box.

   If you want the search to be case-sensitive, check the **Match case** check box. The search matches uppercase and lowercase letters exactly as they appear in the **Find what** text box.

   In the example, the **Match case** check box is checked, so the search finds `Actual` but not ACTUAL.

3. Enter the text in the **Replace with** box that will replace one or all instances of the contents of the **Find what** box.
4. Click **Find Next**.

   If the search finds the text you entered in the **Find what** text box, it highlights the text in the Report Editor.
5. Click **Replace** to replace the highlighted text with the information in the **Replace with** box.
6. Click **Replace All** to replace all instances of the text in the **Find what** box with the information in the **Replace with** box.

   When the search reaches the end of the report script, or if it cannot find the selected text to replace, the following dialog box appears:

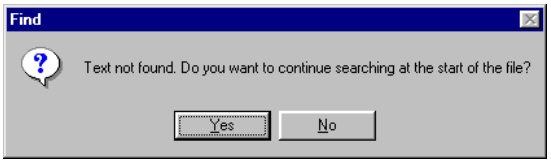

Figure 34-9, Continue Find Dialog Box

   - Click **Yes** to continue the searching the file.
   - Click **No** to close the dialog box and return to the **Replace** dialog box.
7. When you are finished, click Cancel to close the **Replace** dialog box.

Cutting, Copying, and Pasting Text

Cutting, copying, and pasting are basic text editing features that you use to modify your report script. These features work with the desktop's Clipboard, allowing you to take information from the Report Editor (by copying or cutting) and then bring it back into the Report Editor (by pasting). You can copy, cut, and paste between different scripts in the Report Editor window. For example, you can copy text from Script1, open Script2, and paste the text into Script2.

To copy text:

1. In the Report Editor window, select the text to copy.
2. Choose Edit | Copy.

   The text is still displayed in the Report Editor window, and a copy of the information is stored in the desktop's Clipboard.

To cut text:

1. In the Report Editor window, select the text to cut.
2. Choose Edit | Cut.

   The text is removed from the Report Editor window, and is stored in the desktop's Clipboard.

To paste text:

1. In the Report Editor window, place your cursor where you want the text to appear.
2. Choose Edit | Paste.

   The Paste command is disabled if there is nothing stored in the Clipboard.

Deleting Text

To remove text from the Report Editor:

1. In the Report Editor window, select the text to delete.
2. Choose Edit | Delete, or use the `Delete` key on your keyboard.

Running Report Scripts

The Report menu in Report Editor features commands that you can use to choose a destination for the report, select a database against which to run the report script, and run the report script to generate a final report. The following sections describe these commands

Choosing the Report Output

The Output Options menu command opens the **Report Output Options** dialog box that lets you choose three output destinations for your report:

- **Window** displays your report in a window using the default system font.
- **Printer** sends your report to the currently selected printer and lets you choose the font to use for the report. If **Window** is also selected, window output appears in the same font.
- **File** sends your report to a `.RPT` file.

You can select as many output options as you like, but you must select at least one.

Sending the Report Output to a Window

To send your report output to a window:

1. Choose Report | Output Options from the Report Editor.

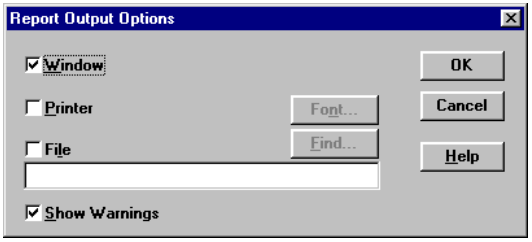

Figure 34-10, Report Output Options Dialog Box with Window Check Box Selected

   The **Window** check box is selected as the default.

   If you also check the **Printer** check box, you can select a printer font for Essbase to use as the screen font. See "Sending Report Output to a Printer" on page 34-20 for details.

2. The **Show Warnings** check box is selected as the default. This option displays any warning messages about report processing in a window on the screen. If you want to turn off this display, click the check box.

3. To display your report in a window, click OK.

   The **Report Output Options** dialog box closes. You are ready to select a database and run the report.

Note: You must check at least one output option to run a report, and you can check any combination of output options.

Sending Report Output to a Printer

To send your report output to a printer:

1. Choose Report | Output Options from the Report Editor menu.
2. Click the **Window** check box to deselect it if you do not want the report to be displayed in a window.
3. Check the **Printer** check box. Application Manager enables the **Font** button.

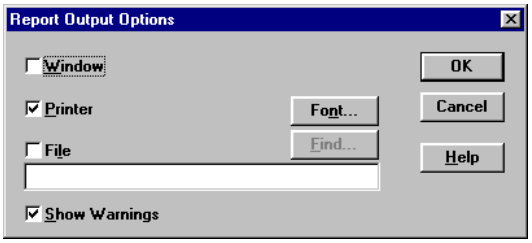

Figure 34-11, Report Output Options Dialog Box with Printer Check Box Selected

4. Click the **Font** button to select a printer font for the report.

   The fonts, font styles, and font sizes available on your system are shown in the **Font** dialog box.

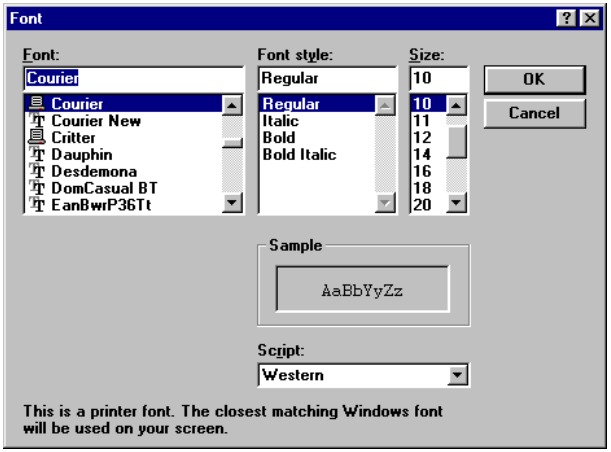

Figure 34-12, Font Dialog Box

5. From the **Font** list, select the font type to use in the report.

6. If you want to change the style of the font, choose a different style from the **Font Style** list.
7. From the **Size** list, choose the point size of the font. A sample of the font appears in the **Sample** box.
8. When you have finished configuring font options, click OK.

   The **Font** dialog box closes, and you return to the **Report Output Options** dialog box.
9. In the **Report Output Options** dialog box, the **Show Warnings** check box is selected as the default. This option displays any warning messages about report processing in a window on the screen. If you want to turn off this display, click the check box.
10. Click OK to prepare the report for the printer.

    The **Report Output Options** dialog box closes. You are ready to select a database and run the report.

Note: You must check at least one output option to run a report, and you can check any combination of output options.

Sending Report Output to a File

To send your report output to a file:

1. Choose Report | Output Options from the Report Editor.
2. Click the **Window** check box to deselect it if you do not want the report to be displayed in a window.
3. Check the **File** check box.

   Application Manager enables the **Find** button.

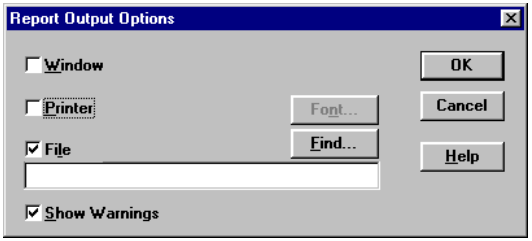

Figure 34-13, Report Output Options Dialog Box with File Check Box Selected

4. If you want to save the report to your desktop machine's file system, click **Find** to choose a directory location.

The file name mask for report output, *.RPT, appears in the **File Name** text box. Any report scripts already in the current directory appear in the list box.

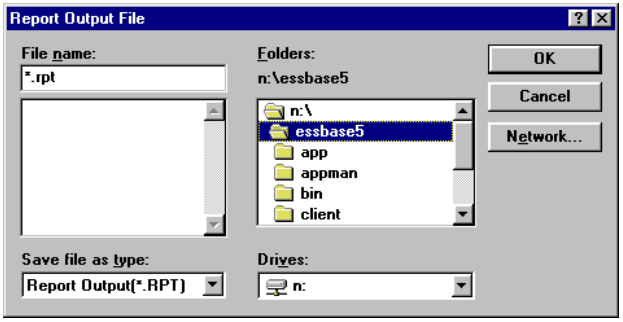

Figure 34-14, Report Output File Dialog Box

5. In the **File name** text box, enter a name for the report script, while retaining the .RPT extension.

   Report script files follow the same naming conventions as DOS files.

6. In the **Folders** list box, select the directory location for the report file.

7. Click OK to save the name and location. The **Report Output File** dialog box closes and returns to the **Report Options** dialog box. The **File** text box displays the file name and directory location of the report script.

8. Click OK to prepare the report script.

9. The **Show Warnings** check box is selected as the default. This option displays any warning messages about report processing in a window on the screen. If you want to turn off this display, click the check box.

   The **Report Output Options** dialog box closes. You are ready to select a database and run the report.

Note: You must check at least one output option to run a report, and you can check any combination of output options.

Choosing Report Databases

The **Select Database** dialog box lets you select a database against which to run your report script.

To choose a report database:

1. Choose Report | Database from the Report Editor.

   The **Select Database** dialog box appears.

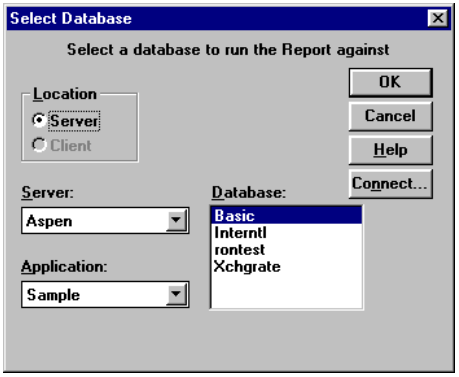

Figure 34-15, Select Database Dialog Box

2. Click the **Server** or **Client** option to specify the location of the database.
3. From the **Server** list box, select the server where the database resides.

   If no server names appear in the list, click the **Connect** button to log onto the server.
4. Select the application where the database resides from the **Application** list box.

   Application Manager lists the databases that reside on the selected server and in the selected application.
5. Select the database from the **Database** list.
6. Click OK.

When you run the report, it reports on the selected database.

Note: If the report is already associated with a database or application other than the one you specify, the Select | Database command does not change that association but simply overrides the default database for this Report Editor session only.

Running the Report

The Report | Run menu command sends your report to the screen and any other output destination you have selected. See "Choosing the Report Output" on page 34-18 to learn how to choose a report output.

To run the report script, choose Report | Run.

Running Multiple Report Script Files

You can use the RUNREPT command in ESSCMD to run one or more report script files. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Executing a Report String Without Creating a Report Script

You can use the REPORTLINE command in ESSCMD to run a single report string. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

You can use the REPORT command in ESSCMD to run multiple report strings. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Developing Free-Form Reports

Free-form reports are often easier to create than structured reports. The free-form reporting style is ideal for ad hoc reporting in the Application Manager's Report Editor window.

A free-form report does not include PAGE, COLUMN, or ROW commands, and instead gathers this information from a series of internal rules that are applied to the report script by the Report Extractor when you run the report.

The following example script and report illustrate free-form reporting:

```
Sales Colas
Jan Feb Mar
Actual Budget
Illinois
Ohio
Wisconsin
Missouri
Iowa
Colorado
{UCHAR}
Central
     !
```

This example produces the following report:

```
                              Sales 100

                 Jan                 Feb                 Mar
           Actual   Budget    Actual  Budget    Actual  Budget
           =======  =======   ======  ======    ======  ======

Illinois       829      700      898     700       932     700
Ohio           430      300      397     300       380     300
Wisconsin      490      300      518     400       535     400
Missouri       472      300      470     300       462     300
Iowa           161        0      162       0       162       0
Colorado       643      500      665     500       640     500
========       ===      ===      ===     ===       ===     ===
  Central    3,025    2,100    3,110   2,200     3,111   2,200
```

You can use formatting commands to add specific formats to a free-form report. The rest of the report is automatically produced in a format similar to that of any other report. When PAGE, COLUMN, and ROW commands are omitted, Essbase formats free-form reports according to the following rules:

1. The Report Extractor finds the last member or members of a single dimension defined in the report specification (before the report output operator !). This dimension becomes the ROW dimension for the report. All remaining selections become PAGE or COLUMN dimensions, as defined by rules 2 and 3.
2. The Report Extractor searches for any single-member selections. If a single member is found that does not satisfy rule 1, that dimension becomes a PAGE dimension.

3. The Report Extractor searches for all remaining dimension members that do not satisfy rules 1 or 2. If any remaining members are found, those dimensions become COLUMN dimensions. COLUMN dimensions are nested in the order of selection in the free-form script.
4. The Report Extractor searches the database outline for any dimensions not specified in the report specification. If any unspecified dimensions are found, they become PAGE dimensions (the default for single-member selections, as defined in rule 2).
5. A subsequent selection of one or more consecutive members from a given dimension overrides any previous selection for that dimension.

For example, the following report recognizes California, Oregon, Washington, Utah, Nevada, and West as members of Market.

```
Sales
Jan Feb Mar
Actual Budget
Apr May Jun
California
Oregon
Washington
Utah
Nevada
{UCHAR}
West
    !
```

The Report Extractor applies free-form formatting rules to this report as follows:

1. Because California, Oregon, Washington, Utah, Nevada, and West are listed last, the Report Extractor treats them as if `ROW (Market)` had been specified (according to rule 1).
2. Sales is a single-member selection from dimension Measures. The Report Extractor treats this member as if `PAGE (Measures)` had been specified (according to rule 2).
3. After searching the remaining members, the Report Extractor finds members of dimensions Year and Scenario, which it treats as `COLUMN (Year, Scenario)`, according to rule 3.
4. The Report Extractor searches the Database Outline and finds that dimension Product is not specified in the report specification. Since Product is a single-member selection, the Report Extractor treats this member as if `PAGE (Product)` had been specified (according to rule 4).

5. Finally, the Report Extractor finds that Apr May Jun is from the same dimension as Jan Feb Mar and appears on a subsequent line of the script. The Report Extractor discards the first specification (Jan Feb Mar) and uses the second (Apr May Jun).

As a result, the report example produces the following report:

```
                              Product Sales

                       Actual                         Budget
              Apr       May       Jun       Apr       May       Jun
         ======== ======== ======== ======== ======== ========


California  3,814     4,031     4,319     3,000     3,400     3,700
Oregon      1,736     1,688     1,675     1,100     1,000     1,100
Washington  1,868     1,908     1,924     1,500     1,600     1,700
Utah        1,449     1,416     1,445       900       800       800
Nevada      2,442     2,541     2,681     1,900     2,000     2,100
======      =====     =====     =====     =====     =====     =====
  West     11,309    11,584    12,044     8,400     8,800     9,400
```

Note: You cannot use substitution variables in free-form mode.

Chapter 35 Developing Report Scripts

Once you understand the basics of creating report scripts, you can create more complex reports.

You create a report using *extraction commands* which specify member combinations for pages, columns, and rows. You use *formatting commands* to determine the visual design of the report and to control some of the data values' display. Formatted data values are displayed in the report when you run the script, based on the combined extraction and report commands.

Extraction commands do the following:

- Determine the selection, orientation, grouping, and ordering of raw data records extracted from the database. Extraction commands are based on either dimension or member names, or keywords. Their names begin with the left angle bracket (<) .
- Apply to the report from the line on which they occur until the end of the report. If another extraction command occurs on a subsequent line of the report, it overrides the previous command.

Formatting commands do the following:

- Let you customize the format and appearance of a report and create report-time calculations. Formatting commands are generally enclosed inside left and right curly braces ({}), although there are several formatting commands that begin with the < character.
- Are either applied globally within the report script or are specific to a member.

This chapter provides information about creating complex report scripts, including:

For fundamental information about reports and report scripts, see Chapter 34, Quick Start to Report Scripts.

Syntax Guidelines

To build a report, you enter commands that define the layout, member selection, and format you want in the Application Manager's Report Editor. When you write a report script, follow these guidelines:

- Separate commands with at least one space, tab, or new line. Report processing is not affected by extra blank lines, spaces, or tabs.
- Enter commands in either uppercase or lowercase. Commands are not case-sensitive. If the database outline is case-sensitive, the members in the report script must match the outline.
- To start report processing, enter the ! (bang) report output command or one or more consecutive numeric values. You can place one or more report scripts, each terminated by its own ! command, in the same report file.
- You can group more than one format command within a single set of curly braces. For example, these formats are synonymous:

  ```
  {UDATA SKIP}
  {UDATA} {SKIP}
  ```

- You can use command abbreviations to speed up command entry. For example, instead of <SORTMBRNAMES, enter <SORTMBRN. Use only the abbreviations documented in the online *Technical Reference* in your `DOCS` directory. If you use other abbreviations, the Report Writer may execute the wrong command.
- Enclose member names that begin with an ampersand, or contain spaces or the word Default, in double quotes; for example, "&Product", "Cost of Goods Sold", "Default".

- If a formatting command is preceded by three or more underscore, equal sign, or hyphen characters, respectively, the Report Extractor assumes that the characters are extraneous underline characters and ignores them. For example, ===<SKIP 1>.
- Use // (double slash) to indicate a comment. Everything on the line following a comment is ignored by the Report Writer. Each line of a comment must start with a double slash, so you can include multi-line comments.

Designing the Page Layout

Reports are two-dimensional views of multidimensional data. You can use page layout commands to incorporate additional dimensions that are defined as nested groups of columns or rows on a page, or additional pages in the report.

The page layout is composed of headings that make up the columns and rows of a page. You define the basic layout of a report using page, row, and column data extraction commands combined with specific member selections.

Each component of page layout has a different extraction command:

<PAGE **<COLUMN** **<ROW**

In addition, the <ASYM and <SYM commands override the default method of interpreting the column dimension member lists, and produce either an asymmetric or symmetric report format.

For information about formatting the page, column, or row headings, see "Formatting Page, Column, and Row Headings" on page 35-8.

Creating Page, Column, and Row Headings

To design the report headings:

1. Type <PAGE (*dimensionname*, *dimensionname*), where *dimensionname* lists dimensions represented on the current page. All data values on the page have the dimensions in the page heading as a common attribute. For example,

   ```
   <PAGE (Measures, Market)
   ```

2. Press Enter.
3. Type <COLUMN (*dimensionname*), where *dimensionname* equals the name of each dimension to display across the page. For example,

   ```
   <COLUMN (Year)
   ```

   You can add dimension names to create nested column headings.

4. Press Enter.
5. Type `<ROW (`*`dimensionname`*`)`, where *`dimensionname`* equals the name of each dimension to display down the page. For example,

   ```
   <ROW (Market)
   ```

6. Press Enter.

Note: You can select additional members to associate to the heading commands. If you type a member name as a parameter for the PAGE, COLUMN, or ROW commands, the Report Extractor automatically associates the member with the appropriate dimension.

The following report script is based on the Sample Basic database:

```
<PAGE (Product, Measures)
<COLUMN (Scenario, Year)
Actual
<ICHILD Qtr1
<ROW (Market)
<IDESC East
!
```

This script produces the following report:

```
               Product Measures Actual

                     Jan      Feb      Mar     Qtr1
                ======== ======== ======== ========

New York             512      601      543    1,656
Massachusetts        519      498      515    1,532
Florida              336      361      373    1,070
Connecticut          321      309      290      920
New Hampshire         44       74       84      202
  East             1,732    1,843    1,805    5,380
```

Modifying Headings

You can perform the following modifications to headings in the report:

To Do This...	Use This Report Command
Create a custom page heading in place of the default heading, which is displayed at the top of each page in the report or immediately following a HEADING command. Use the ENDHEADING command to specify the end of the custom heading.	STARTHEADING
Display the page heading, either the default heading or the heading as defined with the STARTHEADING and ENDHEADING commands. Use this command to re-enable the page heading display if the SUPHEADING command has been used.	HEADING
Force the immediate display of the heading without waiting for the next non-suppressed data row, when heading suppression commands are in use.	IMMHEADING
Automatically turn on the display of the column header.	COLHEADING
Display the page heading before the next data output row.	PAGEHEADING

For information about suppressing the display of headings in the report, see "Suppressing Page, Column, and Row Formatting" on page 35-11.

Creating Symmetric and Asymmetric Reports

Essbase reports can contain symmetric or asymmetric column groups. Essbase determines the symmetry of column groups automatically, based on the members you select.

A *symmetric report*, shown below, is characterized by repeating, identical groups of members.

```
                East                                West
      Budget          Actual              Budget        Actual
Q1   Q2   Q3      Q1   Q2   Q3         Q1   Q2   Q3   Q1   Q2   Q3
```

An *asymmetric report*, shown below, is characterized by groups of nested members that differ by at least one member in the nested group. There can be a difference in the number of members or the names of members.

```
             East                                    West
    Budget          Actual                          Budget
 Q1  Q2  Q3      Q1  Q2  Q3                      Q1  Q2  Q3
```

By default, Essbase creates a symmetric report unless you select the same number of members for all column dimensions.

See Chapter 36, Examples of Report Scripts, for an example of an asymmetric report.

The Essbase evaluation of symmetry versus asymmetry takes place prior to any ordering, restriction on columns, or application of the effects of calculated columns.

Overriding Default Column Groupings

You can override the default column grouping that Essbase selects for reports with the `<SYM` and `<ASYM` commands. `<SYM` and `<ASYM` affect the member selection commands that follow them in a report.

1. Use the `<SYM` command when the selection of column members meets the requirements of the rule for asymmetry, but you want to produce a symmetric report. The `<SYM` command always produces a symmetric report, creating all combinations of each column dimension.
2. Turn off the symmetric format and restore the rules for asymmetric reports with the `<ASYM` command.

Changing Column Headings

If you only need to change the column headings rather than the symmetry of the report, the `<PYRAMIDHEADERS` and `<BLOCKHEADERS` formatting commands are useful.

- Use the `<BLOCKHEADERS` formatting command to change the pyramid-style headers used in symmetric reports to block-style headers like those used in asymmetric reports. A symmetric report uses the `<PYRAMIDHEADERS` mode of column layout by default.
- Use the `<PYRAMIDHEADERS` formatting command to change the block-style headers used in asymmetric reports to pyramid-style headers like those used in symmetric reports. An asymmetric report uses the `<BLOCKHEADERS` mode of column layout.

Formatting

Formatting commands define the format of data and labels in the final report. These commands are generally enclosed in left and right curly braces ({}).

The two types of formatting commands are *global* and *member-specific* commands.

- *Global commands* are executed when they occur in the report script file, and stay in effect until the end of the report file or until another global command replaces them.

  For example, the `{SUPMISSINGROWS}` command suppresses all rows in the report script file that contain only missing values.

- *Member-specific commands* are executed as they are encountered in the report script, usually the next member in the report script, and affect only that member. A format attached to a member is executed before that member is processed.

  For example, the `{SKIP}` command skips the specified number of rows between row dimensions in a report script. If you want a additional rows to skip lines, you must use the SKIP command again.

Formatting Report Pages

There are a number of formatting commands that you can use to design the look of your final report pages.

See Chapter 36, Examples of Report Scripts, for report formatting examples.

Setting the Page Length, Width, and Centering

You can set the following page specifications in the report script:

To Do This...	Use This Report Command
Specify the column widths in a report.	WIDTH
Set the report's left margin.	LMARGIN
Set the center of the page.	SETCENTER

Inserting Page Breaks

You can set the following types of page breaks in the report script:

To Do This...	Use This Report Command
Set the number of lines for each page.	PAGELENGTH
Force a page break regardless of how many lines have been generated for the current page.	NEWPAGE
Insert a page break whenever a member from the same dimension as the member in the command changes from one line in the report to the next. Use the NOPAGEONDIMENSION command to turn off this function.	PAGEONDIMENSION
Enable page breaks in a report when the number of lines on a page is greater than the current PAGELENGTH setting.	FEEDON

Formatting Page, Column, and Row Headings

Column and row formatting commands make up a special type of format setting commands.

Specifying Column Formats

Specifications for column formatting commands can precede or follow the columns to which they apply, depending on the desired format.

For example, in the following script, based on the Sample Basic database, the first `{DECIMAL}` command is processed after only two columns are set up, `Actual` and `Budget`. The `{DECIMAL}` command, however, refers to a column three, which does not yet exist. Essbase responds to this command by dynamically expanding the report to three columns. When the report specification expands to six columns, the `{DECIMAL}` formatting applies to columns three *and* six (and all multiples of three).

Essbase performs this pattern extension on the assumption that when another dimension is added, causing repetitions of previous column dimension groups, the formatting should repeat as well. The second {DECIMAL} formatting command is then applied to columns 1 and 4 only, as it occurs after the creation of six columns.

```
<PAGE (Measures, Market)
Texas Sales
     <COLUMN (Scenario, Year)
     Actual Budget
{DECIMAL 2 3 }
     Jan Feb Mar
{DECIMAL 1 1 4 }
<ROW (Product)
<DESC "100"
     !
```

This script produces the following report:

```
                          Sales Texas

                Actual                        Budget
          Jan       Feb       Mar       Jan       Feb       Mar
          ===       ===       ===       ===       ===       ===

100-10    452.0      465     467.00     560.0      580      580.00
100-20    190.0      190     193.00     230.0      230      240.00
100-30 #Missing  #Missing  #Missing  #Missing  #Missing   #Missing
```

The following scripts demonstrate two approaches to column formatting that produce identical results. In the first script, the first two {DECIMAL} commands are positioned to format every first and third column by distributing the formats when Jan Feb appears *after* processing the {DECIMAL} command. These examples are based on the Sample Basic database.

```
//Script One: Format Columns by Distributing the Formats

<PAGE (Measures, Market)
California Sales
     <COLUMN (Scenario, Year)
     Actual Budget Variance
{DECIMAL 1 1 }
{DECIMAL 2 3 }
     Jan Feb
     //     {DECIMAL 1 1 4 }   These lines are commented; the
     //     {DECIMAL 2 3 6 }   Report Extractor ignores them.
<ROW (Product)
<DESC "100"
       !
```

The two {DECIMAL} commands are positioned to format the individual columns 1, 3, 4, and 6.

```
//  Script Two: Format Columns by Direct Assignment

<PAGE (Measures, Market)
California Sales
     <COLUMN (Scenario, Year)
     Actual Budget Variance
     //     {DECIMAL 1 1 }     These lines are commented; the
     //     {DECIMAL 2 3 }     Report Extractor ignores them.
     Jan Feb
{DECIMAL 1 1 4 7 }
{DECIMAL 2 3 6 9 }
<ROW (Product)
<DESC "100"
       !
```

Both scripts produce the following report:

```
                          Sales California

              Actual           Budget            Variance
            Jan     Feb      Jan       Feb      Jan       Feb
          =====    ====     ====      ====    =====      ====

 100-10    678.0     645    840.00    800.0     (162)  (155.00)
 100-20    118.0     122    140.00    150.0      (22)   (28.00)
 100-30    145.0     132    180.00     160.0       (35)   (28.00)
```

Accommodating Long Column and Row Names

Member names that are too long to fit into the column are automatically truncated; the tilde character (~) signifies that part of the name is missing. Long member names are common when using aliases in the report.

There are a several ways to modify your columns to display the entire member name.

To Do This...	Use This Report Command
Define the column width for all row members in the column.	NAMEWIDTH
Change where the row member column appears, and to shift the remaining columns left or right to make room fro the long member names.	NAMESCOL

Suppressing Page, Column, and Row Formatting

You can suppress the display of page heading, columns, and rows in your report by using various SUPPRESS commands.

To Suppress the Display of...	Use This Report Command
The default column heading in the report.	SUPCOLHEADING
Rows that have only zero or missing values.	SUPEMPTYROWS
All rows that contain missing values. Use INCMISSINGROWS, INCZEROROWS, or INCEMPTYROWS to display rows that are empty, or have missing data or zeros.	SUPMISSINGROWS
The page member heading whenever a heading is generated.	SUPPAGEHEADING
The page and column headings, all member names, page breaks, commas, and brackets in the final report. To turn on the display of column(s) of row member names, use the NAMESON command. To turn on the use of commas and brackets, use the COMMAS and BRACKETS commands.	SUPALL
The default heading (page header and column headers) or custom header, if defined, at the top of each page.	SUPHEADING
Row member names in the final report. Use the NAMESON command to include row member names in the report.	SUPNAMES

To Suppress the Display of...	Use This Report Command
All output while continuing to process all operations, such as calculations, format settings, and so forth. Use the OUTPUT command to reverse the actions of SUPOUTPUT.	SUPOUTPUT
Currency information when you use the CURRENCY command to convert the data values in your report to a specified currency.	SUPCURHEADING

Repeating Row Names

To repeat the row member names on every line of the report, use the `ROWREPEAT` command. Use the `NOROWREPEAT` command to prevent row member names from being repeated on each line of the report if the row member name does not change on the next line. `NOROWREPEAT` is enabled by default.

Formatting Reports with Tab Delimiters

You can place tabs between columns rather than spaces in your report scripts. This is useful when you want to export report output into another form.

To replace spaces with tab delimiters:

Type `{TABDELIMIT}` anywhere in the report script.

When you save the report script, Essbase automatically replaces the spaces with tabs. When you view the report in the Report Viewer, black squares indicate tab marks.

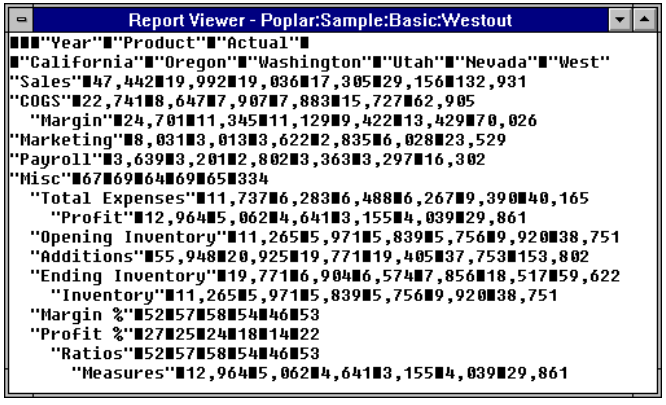

Figure 35-1, Tab-Delimited Report

Adding Totals and Subtotals

Column and row calculations let you create additional calculations that are not defined as part of the database outline. For example, you can use column and row calculations to create extra columns or rows in a report, based upon selected data members, and perform calculations on these or existing columns and rows.

For examples of report scripts that contain column and row calculations, see Chapter 36, Examples of Report Scripts.

Totaling Columns

The CALCULATE COLUMN command lets you create a new report column, perform on-the-fly calculations, and display the calculation results in the newly created column.

The following table summarizes column calculation commands:

To Do This...	Use This Report Command
Create a new report column, perform Dynamic Calculations, and display the calculation results in the newly created column. Use the OFFCOLCALCS command to temporarily disable column calculations in the report, and ONCOLCALCS to re-enable calculations.	CALCULATE COLUMN
Remove all column calculation definitions from the report.	REMOVECOLCALCS

CALCULATE COLUMN adds up to 499 ad hoc column calculations to a report. Each new calculated column is appended to the right of the existing columns in the order in which it is created, and given the next available column number. These columns calculate the sum of data across a range of columns or an arithmetic expression composed of simple mathematical operators.

The CALCULATE COLUMN command supports the standard mathematical operations. For syntax and parameter descriptions, see the online *Technical Reference* in your DOCS directory.

If you use the same name for more than one column, Essbase creates only the last column specified in the CALCULATE COLUMN command. Use a leading space with the second name (and two leading spaces with the third name, and so on) to create a unique column name.

Alternately, you can add descriptive text far enough to the right that it is truncated to the column width. You could, for instance, use the names "Q1 Actual" and "Q1 Budget" to distinguish similar column names without affecting the appearance of the report. Column names are printed with right justification until the column header space is filled. Excess characters are then truncated to the right.

Divide lengthy column name labels into two or more lines. The maximum number of lines across which you can divide a label is equal to the number of column dimensions designated in the report specification. To do this, insert the tilde character (~) in the name at the point where you want the break. You must also specify at least two members for each column dimension to use the maximum number of lines.

This example is based on the Sample Basic database.

```
{CALCULATE COLUMN "Year to Date~Actual Total" = 1 : 2}

{CALCULATE COLUMN "Year to Date~Budget Total" = 3 : 4}
```

The example produces the following report:

```
                                    Sales East

             Actual   Year to Date      Budget    Year to Date
          Jan    Feb  Actual Total    Jan    Feb  Budget Total
       ====== ====== ============= ====== ====== =============

400-10    562    560         1,122    580    580         1,702
400-20    219    243           462    230    260           722
400-30    432    469           901    440    490         1,391
```

As a rule, in symmetric reports, if a calculated column name has fewer levels than the number of column dimensions, the previous member (to the left) of each of the column dimensions, above the top level supplied in the calculated column name, is attributed to the calculated column. If normal PYRAMIDHEADERS mode is in use, the centering of those higher-level column members shifts to the right to include the calculated column or columns. Column header members on the same level as calculated column names are not applied to the new calculated column or columns, and their centering does not shift.

If BLOCKHEADERS mode is in use, that is, if every member applying to a column is repeated above that column, the same rules apply, except that instead of shifting column header member centering, they are repeated in the appropriate higher levels of the calculated column name.

Asymmetric reports do not have groups of columns that share a member attribute. These reports still allow multiple-level column names up to the number of column levels defined, but member attributes from preceding columns are not automatically shared and used for those levels that are not defined.

In cases where there are fewer column header dimensions than the number of levels that you want, you can create multi-line column labels. In this case, use `TEXT`, `STARTHEADING`, `ENDHEADING`, and other formatting commands to create a custom heading.

For the syntax and definitions of column calculation commands, see the online *Technical Reference* in your `DOCS` directory.

Column Numbering

If the number of regular (non-calculated) columns varies in the report because multiple sections in the report have different numbers of columns, the column numbers used to identify the calculated columns shift accordingly. For example:

- If the first section of a report has 12 columns (including row name columns), and 3 calculated columns are declared, column numbers 0-11 are the regular columns, and columns 12-14 are the calculated columns.
- If a second section of the report reduces the number of regular columns to 6, then the regular columns are columns 0-5, and the same calculated columns are columns 6-8.
- Likewise, if the number of regular columns is increased, the numbering of the calculated columns starts at a higher number.

In the example, **CC1**, **CC2**, and **CC3** represent the names of three calculated columns names. The column numbering for a report with two different sections with varying numbers of regular columns would be:

```
internal
col # s: 0    1      2      3      4      5      6      7
            Jan    Feb    Mar    Apr    CC1    CC2    CC3
            ===    ===    ===    ===    ===    ===    ===
   Sales      1      3      5      3     22     55     26
   Expense    1      2      5      3     23     65     33

                 same report- new section

internal
col # s: 0    1      2      3      4      5
           Qtr1    YTD    CC1    CC2    CC3
            ===    ===    ===    ===    ===
   Sales      2      9     22     57     36
   Expense    4      8     56     45     33
```

If you do not want the calculated columns in the second section, or if you need a different set of column calculation, use the command REMOVECOLCALCS to clear the old ones out. You can then define new column calculations.

This example assumes that all three column calculations had no references to regular columns other than columns 1 and 2. If **CC3**'s calculation were = 1 + 3 + 6, when the second section of the report starts, an error would occur stating that the column calculation referred to a nonexistent column (6).

Totaling Rows

Row calculations create summary rows in a report. You can use *summary rows* to calculate the sum of data across a range of rows or to calculate an arithmetic expression composed of simple mathematical operators.

The following table summarizes row calculation commands:

To Do This...	Use This Report Command
Create a new row and associate it with a row name or label. This is similar to declaring a variable. You can also perform simple calculations with CALCULATE ROW. For more complex calculations, use SETROWOP. See also OFFROWCALCS and ONROWCALCS.	CALCULATE ROW
Temporarily disable row calculations in the report. See also CALCULATE ROW and ONROWCALCS.	OFFROWCALCS
Re-enable calculations after using OFFROWCALCS. See also CALCULATE ROW and OFFROWCALCS.	ONROWCALCS
Define complex calculations for the row specified in CALCULATE ROW. SETROWOP defines a calculation operator to be applied to all subsequent output data rows. You can display the calculation results in the newly created row with the PRINTROW command.	SETROWOP
Immediately display the row specified in CALCULATE ROW to the report.	PRINTROW
Reset the value of the calculated row to `#Missing`. See also CLEARALLROWCALC.	CLEARROWCALC
Reset the value of all calculated rows after using the CLEARROWCALC command.	CLEARALLROWCALC
Create a new calculated row with captured data. See also SAVEANDOUTPUT.	SAVEROW
Capture data and output the result after using the SAVEROW command.	SAVEANDOUTPUT

For the syntax and definitions of row calculation commands, see the online *Technical Reference* in your DOCS directory.

Commands that designate columns must use valid data column numbers, as determined by the original order of the columns.

- Precede and follow all operators in an expression with a single space.
- Essbase does not support nested (parenthetical) expressions.
- Essbase supports integer and floating-point constants in expressions as single entries or members of an array.

The CALCULATE ROW command can specify an operation (+, -, *, /, or OFF) as an equation consisting of constants, other calculated rows, and operators. Equations are evaluated at the time of declaration.

If you specify an operator, it applies to subsequent output rows and stores the result in the calculated row. This is useful for aggregating a series of rows to obtain a subtotal or total. To reset the operator, use SETROWOP. If the CALCULATE ROW command does not specify either an equation or an operator, the + operator is assumed.

The CALCULATE ROW command supports the standard mathematical operations. For syntax and parameter descriptions, see the online *Technical Reference* in your DOCS directory.

This example is based on the Sample Basic database.

```
{ CALC ROW "Total Sales" = "Sales..Group1"
     + "Sales..Group2" }
```

The example creates "Total Sales" based on two other calculated rows.

Changing How Data Displays

You can use a variety of formatting commands to customize how data displays in your final report.

Underlining

Use underlining as a visual aid to break up blocks of information in a report.

To Do This...	Use This Report Command
Set the default underline character that will display in the report.	UNDERLINECHAR
Underline all non-blank characters in the preceding row.	UCHARACTERS
Underline all the columns in the preceding row.	UCOLUMNS
Underline all the data columns for a row, while not underlining the row name columns.	UDATA
Underline all the row name columns in the preceding row while not underlining the data columns.	UNAME
Underline the row member names in a row whenever a member from the same dimension as the member in the command changes. Use the NOUNAMEONDIM command to turn off underlining for new rows.	UNAMEONDIMENSION

Suppressing Data Formatting

You can suppress data that you do not want displayed in your final report by using various SUPPRESS commands.

To Suppress the Display of...	Use This Report Command
Brackets around negative numbers. Use the BRACKETS command to re-enable brackets.	SUPBRACKETS
Commas in numbers greater than 999. Use the COMMAS command to re-enable commas.	SUPCOMMAS
Rows that have only zero data values. Use INCZEROROWS or INCEMPTYROWS to re-enable the display.	SUPZEROROWS
The European method for displaying numbers (2.000,01 whereas the non-European equivalent is 2,000.01). Use the EUROPEAN command to re-enable European number display.	SUPEUROPEAN
The automatic insertion of a page break whenever the number of lines on a page exceeds the current PAGELENGTH setting.	SUPFEED
Formats that produce output such as underlines and skips. Use INCFORMATS to re-enable the display.	SUPFORMATS
Text masks that were defined in the report using the MASK command. Use INCMASK to re-enable the display.	SUPMASK

See “Suppressing Page, Column, and Row Formatting” on page 35-11 for information about suppressing the display of page, column, and row formats.

Indenting

Use indenting to provide visual clues to row levels of the script.

To Do This...	Use This Report Command
Shift the first row names column in column output order by a specified number of characters.	INDENT
Indent subsequent row members in the row names column based on the generation in the database outline. Use the NOINDENTGEN command to left-justify row member names based on generation name.	INDENTGEN

Inserting Custom Titles

Titles are user-generated and optional, in contrast to the automatically generated page and column headings and row names, which describe the data on the report page.

Titles repeat at the top of each report page, and provide such valuable information about a report as:

- A date and time stamp
- The user ID of the person running the report
- Page numbers
- The name of the source database
- Any other descriptive information

To add a title to the report, use the TEXT command, combined with:

- Any of the predefined keywords that automatically display information in the report
- A text string that you define

Note: You can also use the TEXT command at the bottom of the report to provide summary information.

See the online *Technical Reference* in your `DOCS` directory for the syntax and definitions of Report Writer commands.

Replacing Missing Text or Zeros with Labels

When you run a report, there are often many empty data cells where no data was applicable to the retrieval, or cells where the value is zero.

The report displays the default #MISSING label in the data cell when no data values are found.

To replace the #MISSING label with a text label:

At the point in the script where you want to replace the #MISSING label with a text label, type

```
{MISSINGTEXT ["text"]}
```

where *text* is any text string that you want to display in the data cells.

You can place the MISSINGTEXT command at any point the report script; the command applies throughout the script.

Note: You can also suppress #MISSING labels from appearing in the report. See "Suppressing Data Formatting" on page 35-20 for information about suppressing labels, or see the online *Technical Reference* in your `DOCS` directory for the syntax and definitions of Report Writer commands.

To replace zeros with a text label:

At the point in the script where you want to replace zeros with a text label, type

```
{ZEROTEXT ["text"]}
```

where *text* is any text string that you want to display in the data cells.

Adding Blank Spaces

Adding blank spaces in a report draws the reader's eye to key information such as totals.

To Do This...	Use this Report Command
Add one or more blank lines in the final report.	SKIP
Add a blank line when a member from the same dimension as the specified member in the command changes on the next line in the report. Use the NOSKIPONDIMENSION command to turn off insertion of a new line.	SKIPONDIMENSION

Changing How Data Values Display

You can use the following commands to change how data values display in your final report:

To Do This...	Use this Report Command
Turn on the display of commas for numbers greater than 999 after commas have been suppressed with either a SUPCOMMAS or SUPALL command.	COMMAS
Turn on the display of brackets around negative numbers instead of negative signs, after using the SUPBRACKETS command earlier in the script.	BRACKETS
Include a percentage sign or other character after the data values.	AFTER
Include a dollar sign or other character before the data values.	BEFORE
Use the European method for displaying numbers where decimal points are used as the thousands separator character while commas separate the decimal portion of the number from the integer portion (2.000,01, whereas the non-European equivalent is 2,000.01).	EUROPEAN
Overwrite text in each output row with a specified characters and position.	MASK

Selecting and Sorting Members

The data that is displayed in the final report is based upon the members that you select and the order in which you display them. In addition, you can use conditional retrievals to further refine the selection and sorting of members.

Selecting Members

Member selection commands are extraction commands that select ranges of members based on database outline relationships, such as sibling, generation, and level. Using member selection commands ensures that any changes to the outline are automatically reflected in your report, unless you change the member name on which the member selection command is based.

To Do This...	Use This Report Command
Select all the members from the same dimension as the dimension member.	ALLINSAMEDIM
Include all the siblings of the specified member.	ALLSIBLINGS
Include all the ancestors of the specified member.	ANCESTORS
Select all members in the level immediately below the specified member.	CHILDREN
Include the descendants of the specified member to the report, excluding the dimension top.	DESCENDANTS
Select the Level 0 members at the bottom of the dimension.	DIMBOTTOM
Include the top member of the dimension.	DIMTOP
Include a member and its ancestors.	IANCESTORS
Select the specified member and all members in the level immediately below it.	ICHILDREN
Include the specified member and its descendants.	IDESCENDANTS
Include the specified member and its parent.	IPARENT
Include all members from the same dimension and generation as the specified member.	OFSAMEGEN
Include all members from the same dimension and on the same level as the specified member.	ONSAMELEVELAS
Include the parent of the specified member to the report.	PARENT

Selecting Members Using Generation and Level Names

Generation and level name selection commands identify a specific level or generation of members, based on either:

- The default generation or level name in the outline
- The user-defined generation or level name in the outline

When you use generation and level names, changes to the outline are automatically reflected in your report. You can either define your own generation and level names, or you can use the default names provided by Essbase. For information on generations and levels, see Chapter 12, Introducing Dynamic Dimension Building.

Using generation or level names whenever possible makes your report easier to maintain. Because you do not have to specify a member name in your report, you do not need to change the report if the member name is changed or deleted from the database outline.

Note: Generation and level names are stand-alone commands. You cannot use them in place of member names in report extraction or formatting commands. For example, you *cannot* use them with the <DESCENDANTS or <CHILDREN commands.

To use default level names:

At the point in the script where you want to select a member by the default level name, use the following format:

Lev*n*,*DimName*

where *n* is the level number.

DimName is the name of the dimension from which you want to select the members.

Note: Do *not* leave a space after the comma.

For example, Lev1,Year selects all the level 1 members of the Year dimension.

To use default generation names:

At the point in the script where you want to select a member by the default generation name, use the following format:

Gen*n*,*DimName*

where *n* is the generation number.

DimName is the name of the dimension from which you want to select the members.

Note: Do *not* leave a space after the comma.

For example, Gen2,Year selects all the generation 2 members of the Year dimension.

Note: These default generation and level names do not appear in the Outline Editor.

The following example is based on the Sample Basic database. It uses the default generation name Gen2,Year to generate a report that includes the members Qtr1, Qtr2, Qtr3, and Qtr4 from the Year dimension.

```
<PAGE(Product)
<COLUMN(Year)
<ROW (Measures)
{OUTALTNAMES}
Cola
Gen2,Year
Sales Profit
       !
```

The report script produces the following report:

```
                        Cola Market Scenario
                   Qtr1     Qtr2     Qtr3     Qtr4
                ======== ======== ======== ========
Sales             14,585   16,048   17,298   14,893
    Profit         5,096    5,892    6,583    5,206
```

Selecting Dynamic Time Series Members

You create and identify dynamic members in the database outline; they are members that are only calculated during user retrieval requests, such as generating a report script. The Time dimension contains a special Dynamic Time Series tag which has reserved generation names that you can define in the outline alias table.

Generation Name	Reserved Names	Explanation
History	H-T-D	History-to-Date
Year	Y-T-D	Year-to-Date
Season	S-T-D	Season-to-Date
Period	P-T-D	Period-to-Date
Quarter	Q-T-D	Quarter-to-Date
Month	M-T-D	Month-to-Date
Week	W-T-D	Week-to-Date
Day	D-T-D	Day-to-Date

See Chapter 7, Creating Applications and Databases, for information about creating and maintaining Dynamic Time Series generation names in the database outline.

Note: The outline's database header message identifies the number of dynamic members that are enabled in the current outline.

To select a Dynamic Time Series member:

At the point in the script where you want to select a Dynamic Time Series member, use either of the following formats:

```
<LATEST memberName
```

where *memberName* is the name of the member in the Time dimension.

The LATEST command is a global command that is applied to the entire report script, and is an aggregation based on the lowest level member within the dimension.

```
reservedName(memberName)
```

where *reservedName* is the reserved Dynamic Time Series generation name, and the *memberName* is the name of the member in the Time dimension.

If you use this syntax to specify a Dynamic Time Series, the time series name is associated only to the member listed in the argument.

When you run the report script, the members are dynamically updated, and the information is incorporated into the final report.

Notes:

- You must type the Dynamic Time Series string exactly as it appears in the database outline; you cannot create your own string and incorporate it into the final report.
- You can create an alias table for the Dynamic Time Series members in the database outline, and use the aliases instead of the predefined generation names.

Selecting Members Using Boolean Operators

Boolean operators let you specify precise member combinations within your report, which is particularly useful when dealing with large outlines. Use the AND, OR, and NOT Boolean operators, combined with extraction commands, to refine your member selections within the report script.

- Use the AND operator when all conditions must be met.
- Use the OR operator when one condition of several must be met.
- Use the NOT operator to choose the inverse of the selected condition.

To create a Boolean expression using operators:

At the point in the script where you want to use linking, enter the following format:

```
<LINK (extractionCommand [operator extractionCommand])
```

where *extractionCommand* is the member selection command to retrieve data from, and *operator* is either the AND or OR operator.

Note: You must select members from the same dimension, and all extraction command arguments must be enclosed in parentheses, as in the example above. NOT can only be associated with an extraction command, and does not apply to the entire expression.

You can use Boolean operators with member selection commands, such as UDA and wildcards. See the online *Technical Reference* in your DOCS directory for a list of all valid extraction commands that can be used in conjunction with the LINK command.

Examples:

```
<LINK ((<IDESC("100") AND <UDA(Product,Sweet)) OR <LEV (product,0))
```

selects sweet products from the "100" sub-tree, plus all level 0 products.

```
<LINK ((<IDESC("100") AND NOT <UDA (Product,Sweet)) OR <LEV
(product,0))
```

selects non-sweet products from the "100" sub-tree, plus all level 0 products.

See Chapter 36, Examples of Report Scripts, for other examples of narrowing member selection criteria.

Selecting Members Using Substitution Variables

Substitution variables act as global placeholders for information that changes regularly; you set the substitution variables on the server through Application Manager or ESSCMD, and assign a value to each variable. You can then change the value at any time, reducing manual changes to a report script. You do need Database Designer privileges to set substitution variables.

For example, many reports are dependent on reporting periods; if you generate a report based on the current month, you would have to manually update the report script every month. With a substitution variable set on the server, such as CurMnth, you can change the assigned value each month to the appropriate time period. Essbase dynamically updates the information when you run the final report.

See Chapter 7, Creating Applications and Databases, for information about creating and changing substitution variables in the database outline. See the online *Technical Reference* in your DOCS directory, for information about the leading & character.

You can set substitution variables at the following levels:

- Server, providing access to the variable from all applications and databases on the server
- Application, providing access to the variable from all databases within the application
- Database, providing access to the specified database

To use a substitution variable:

The substitution variable must be accessible from the application and database against which you are running the report.

At the point in the script where you want to use the variable, use the following format:

```
&variablename
```

where *variablename* is the same as the substitution variable set on the server. For example,

```
<ICHILD &CurQtr
```

becomes

```
<ICHILD Qtr1
```

Note: The variable name can be an alphanumeric combination up to 80 characters in length, including underscore characters. You cannot use spaces or punctuation in the variable name.

When you run the report script, Essbase replaces the variable name with the substitution value and that information is incorporated into the final report.

Selecting Members with User-Defined Attributes

User-defined attributes (UDA) let you select and report on data based on a common attribute; they are particularly useful when performing member selections from an outline with an unbalanced hierarchy. You can set UDAs on the server for attributes such as color, size, gender, flavor, or any other common member attribute. You must have Database Designer privileges to set UDAs on the server.

You can use the UDA command in conjunction with Boolean operators to further refine your report queries. See "Selecting Members Using Boolean Operators" on page 35-27 for more information.

See Chapter 7, Creating Applications and Databases, for information about creating and maintaining UDAs.

To select members based on a user-defined attribute:

At the point in the script where you want to select members based on the UDA, use the following format:

```
<UDA (dimName,"UDAstring")
```

where the *dimName* is the dimension of the member to select, and *UDAstring* is the user-defined attribute that is set on the server. For example:

```
<UDA (product,"Sweet")
```

When you run the report script, Essbase incorporates the UDA members into the final report.

Note: You must type the UDA string exactly as it appears in the database outline; you cannot create your own UDA string and incorporate it into the report script.

Selecting Members Using Wildcards

You can use wildcards to select either members, generation, or level names in a report script. If you use member names, Essbase searches the member and all descendants of that member. If you specify a generation or level name, Essbase only searches members of that generation or level.

Using wildcards reduces the amount of member information to put into a script and simplifies script maintenance.

The following two types of wildcards are supported in the Report Writer:

- Trailing asterisk * characters at the end of the string to search for common member attributes
- Pattern-matching question mark ? characters anywhere in the string to represent any single character member attribute

To select members using a trailing wildcard:

At the point in the script where you want to select members using a trailing wildcard, use the following format:

```
<MATCH mbrName,"charName*")
```

where *mbrName* is the name of the member to select, and *charName* is the beginning character in the following member. Using the Sample Basic database,

```
<MATCH (Year,"J*")
```

returns Jan, Jun, and Jul.

To select members using a pattern-matching wildcard:

At the point in the script where you want to select members using a pattern-matching wildcard, use the following format:

```
<MATCH (mbrName,"???charNames")
```

where *mbrName* is the name of the member to select, and *charNames* are the characters in the following member. Using the Sample Basic database,

```
<MATCH (Product,"???-10")
```

returns 100-10, 200-10, 300-10, and 400-10.

Note: In the Sample Basic database example, three question marks are used to represent the variable three characters in the string. If only two question marks were used in the example, no matches would be found. You can place question mark wildcards anywhere in the match string.

Selecting Members Using Static Member Names

Static member names are non-changing member names, such as Sales and COGS, that you enter directly into the report script. Although they are easy to establish, static member name definitions can be difficult to maintain if your database outline changes. You must change the member name in every script that is used with that database.

In general, report scripts are easier to maintain if you use generation and level names, or member selection commands, rather than static member names whenever possible. For example, when you remove a product, such as Widgets, from the dimension, the member selection command `<child Product` works, but a specific list referencing Widgets as a static member name generates the following error message:

```
"Unknown Member [Widgets]."
```

Static members tend to be particularly difficult to maintain when dealing with distributed OLAP databases, where both source and target databases must be consistent.

Note: These user-defined, static member generation and level names do not appear in the Outline Editor.

The following report script uses a static member called ProductGroups to select all the Level 1, Product member names from the Sample Basic database.

```
<PAGE(Year)
<COLUMN(Product)
<ROW (Measures)
Qtr1
ProductGroups
Sales Profit
!
```

The report script produces the following report:

```
                        Qtr1 Market Scenario
                   100      200      300      400     Diet

               ======== ======== ======== ======== ========
Sales            25,048   26,627   23,997   20,148   25,731
    Profit        7,048    6,721    5,929    5,005    7,017
```

When you run a report that includes static member definitions, the report displays members in order of their definition in the report script by member name. Sort commands have no effect on static member definitions. See "Sorting Members" on page 35-36 for more information about the effects of sorting members.

Suppressing Shared Members

You can suppress the display of duplicate shared members when you extract data for your report. You can only suppress shared member display in conjunction with the following:

- Generation names
- Level names
- DIMBOTTOM command
- OFSAMEGEN command
- ONSAMELEVELAS command

Suppressing shared members is useful when you want to eliminate unnecessary duplication of data within the report.

To suppress shared members:

At the point in the script where you want to suppress a shared member, type

```
<SUPSHARE
```

This suppresses the display of shared members for the duration of the report script. Use the <SUPSHAREOFF command to reset the display of shared members in the script.

See "Suppressing Page, Column, and Row Formatting" on page 35-11 for more information about suppressing data from your final report.

Selecting Alias Names for Members

Aliases make reports easier to read and help your reader focus on the data values rather than the meanings of member names. You can display members in a report by their aliases. For example, you can display page, column, and row names, such as Diet Cola or Caffeine Free Cola, rather than the corresponding member names 100-20 and 100-30.

To Do This...	Use This Report Command
Display the alias set in the database outline's current alias table, without the member name.	OUTALT
Display the alias set in the database outline's current alias table, followed by the member name.	OUTALTMBR
Display the member name, followed by the alias set in the database outline's current alias table.	OUTMBRALT
Display the alias set in the database outline's current alias table, without the member name, for all members in the report.	OUTALTNAMES
Reset the default display of member names after using the OUTALTNAMES command.	OUTMBRNAMES
Include several alias tables within one report script.	OUTALTSELECT

For a complete listing of alias commands, see the online *Technical Reference* in your DOCS directory.

To Display a Member Name and Alias

You can display members in a report as a combination of the member name and its alias. This lets you display more descriptive page, column, and row names, such as Diet Cola 100-20 or 100-30 Caffeine Free Cola.

To display a member name and alias, use the OUTALTNAMES and OUTALTMBR commands together in a report script, as shown in the following example:

```
<PAGE (Product, Measures)
<COLUMN (Scenario, Year)
{OUTALTNAMES}
<OUTALTMBR
Actual
<ICHILD Qtr1
<ROW (Market)
<IDESC "300"
!
```

The report script produces the following report:

```
                  Dark Cream 300-10 Measures Actual
                       Jan      Feb      Mar      Qtr1
                  ======== ======== ======== ========
Market                 800      864      880    2,544

                  Vanilla Cream 300-20 Measures Actual
                       Jan      Feb      Mar      Qtr1
                  ======== ======== ======== ========
Market                 220      231      239      690

                  Diet Cream 300-30 Measures Actual
                       Jan      Feb      Mar      Qtr1
                  ======== ======== ======== ========
Market                 897      902      896    2,695

                     Cream Soda 300 Measures Actual
                       Jan      Feb      Mar      Qtr1
                  ======== ======== ======== ========
Market               1,917    1,997    2,015    5,929
```

Sorting Members

When you sort the members you include in your report, be aware that sorting commands affect members differently depending on whether they are referenced by member selection commands or by static member definitions. Report Writer commands sort members either by member name or data values.

Member selection commands, such as <CHILD and <DESCENDANT, select members in the order specified by the database outline. By default, a report that includes member selection commands displays members in their hierarchical database outline order. You can override this default by specifying a sort order with a sort command.

Because sort commands affect the order of the members selected by the member selection commands, they must precede any member selection commands to which they apply. If you specify a sort command, the sort order is preserved until another sort command overrides it.

Sort commands modify member selection commands, such as <CHILD and <DESCENDANT. Sort commands do not perform any final sorting of rows during formatting. Be careful when you place a sort command in your report script that you do not start the sort too soon, and that you override it to turn it off, if necessary, before the next selection command.

Sort commands have no effect on static member definitions.

To Do This...	Use This Report Command
Sort all members alphabetically by the alias name of the member, if aliases are used in the report script.	SORTALTNAMES
Sort all following members in ascending order starting with either the letter **a** or the lowest generation and moving toward the letter **z** or the highest generation. This is the default sort order.	SORTASC
Sort all following members in descending order starting with either the letter **z** or the highest generation and moving toward the letter **a** or the lowest generation.	SORTDESC
Sort all following members according to the generation of the member in the database outline.	SORTGEN
Sort all following members according to the level of the member in the database outline.	SORTLEVEL
Sort all members alphabetically by member name.	SORTMBRNAMES
Disable all previous sorting commands so that members added to the report follow the normal hierarchical order based on the database outline.	SORTNONE

For a list of sorting commands syntax and descriptions, see the online *Technical Reference* in your DOCS directory.

Restricting and Ordering Data Values

Several Report Writer commands let you perform conditional retrieval and data sorting in your reports.

To Do This...	Use This Report Command
Specify the conditions the column(s) of a data row must satisfy before the row is returned.	RESTRICT
Specify the ordering of the rows of a report, based on the data values of data columns.	ORDERBY
Specify the number of rows to return. These rows must contain the top values of a specific data column.	TOP
Specify the number of rows to return. These rows must contain the lowest values of a specific data column.	BOTTOM

For the syntax and definitions of these commands, see the online *Technical Reference* in your DOCS directory. For detailed examples using these commands, see Chapter 36, Examples of Report Scripts.

Configurable variables are used during conditional retrievals. For information about setting the Report Writer configurable variables, see Chapter 37, Optimizing Your Reports.

Defining the Order of Operation

RESTRICT, ORDERBY, TOP, and BOTTOM can appear anywhere in the report script and in any order. When using these commands, place all global script formatting commands before a Page member or a Column member, or before a PAGE command or COLUMN command that expands into Page or Column members (for example, IDESC, or ICHILD).

Essbase extracts data and applies restrictions and ordering in the following order:

1. Applies RESTRICT and any existing restrictive option such as SUPPMISSING, SUPZEROS, and SUPEMPTYROWS.
2. Applies TOP and/or BOTTOM.
3. Applies ORDERBY.

Essbase then returns rows and displays output.

See Chapter 34, Quick Start to Report Scripts, for more information.

Using TOP, BOTTOM, and ORDERBY with Sorting Commands

TOP, BOTTOM, and ORDERBY commands sort a report output by its data values. Essbase applies TOP and BOTTOM first, followed by ORDERBY. If the report contains a sort command, such as SORTMBRNAMES, which sorts members and not data, Essbase applies the sort command first, followed by TOP and BOTTOM, and then ORDERBY. ORDERBY is the final sort that takes place.

Restricting Data Ranges

The arguments of the RESTRICT command let you specify qualifications for selecting rows. Essbase includes only qualified rows in the resulting report output.

RESTRICT works only on the range of rows that you specify in a row member selection.

Essbase processes the restrictions from left to right, and does not allow grouping with parentheses in the list of arguments.

For example, the following example is *not* a valid syntax:

```
RESTRICT ( ... (@DATACOL(1) > 300 AND @DATACOL(2) < 600 ) ...)
```

Only one RESTRICT is allowed per report, as terminated by the ! command. If a report script contains more than one report, each RESTRICT overwrites the one in the previous report. For example:

```
RESTRICT ( @DATACOL(1) > @DATACOL(2) AND 800 < @DATACOL(3)
 OR @DATACOL(4) <> #MISSING )
```

This RESTRICT command is equivalent in operation to the following syntax:

```
RESTRICT (((@DATACOL(1) > @DATACOL(2)) AND (800<@DATACOL(3))) OR
(@DATACOL(4) <> #MISSING))
```

Ordering Data

The ORDERBY command orders the output rows according to the data values in the specified column(s). You can specify either ascending "ASC" (the default) or descending "DESC." You can specify different sorting directions in different columns of the same report.

To determine the set of rows to be ordered, specify the row grouping dimension in the command. The default row grouping is the innermost row dimension.

Only one ORDERBY is allowed per report, as terminated by the ! command. If a report script contains more than one report, each ORDERBY overwrites the one in the previous report.

Using ORDERBY with Formatting Commands

Follow these guidelines when using the ORDERBY command in a report script:

- Avoid using row formatting commands when you are using ORDERBY in a report. Formatting commands scheduled for a given point in the report may show up unexpectedly because ORDERBY shifted the row that contained the member with formatting.
- In general, avoid using row formatting commands, and place overall script formatting before column members or commands that expand the column members (such as `ICHILD column dimension, <column ..., column member`).

Specifying Rows to Return

The TOP and BOTTOM commands specify the qualified number of rows with the highest or lowest column values, respectively, within a row group to be returned in a report. If the row group member is not specified, the innermost row group dimension is the default row group.

You can use TOP and BOTTOM together in the same report, but only one TOP and one BOTTOM is allowed per report. In this case, the two commands should have the same data column as their argument in order to prevent confusion. The result of the TOP and BOTTOM command is sorted by the value of the data column specified in the command in descending order.

TOP and BOTTOM work only on the range of rows specified in row member selection.

Note: If TOP or BOTTOM occurs with ORDERBY, the ordering column of the ORDERBY does not have to be the same as the data column of the TOP or the BOTTOM.

If any combination of the ORDERBY, TOP, or BOTTOM commands exist together in a report script, the row group member <*rowgroupmemb*> should be the same. This restriction removes any confusion about the sorting and ordering of rows within a row group.

Warning: Essbase discards rows that contain `#MISSING` values in their sorting column from the set of extracted data rows before the applying the TOP or BOTTOM sort.

For example, this command returns two rows with the highest data values in *col2* (Actual, Qtr2) per row group:

```
1- TOP (2, @DATACOL(2) )
```

When you run this command against the Sample Basic database, the row grouping is *Product*, which implies that for FLORIDA, the report returns 100-10 and 100-30 product rows, and for MAINE, the report returns 100-10, 100-40 product rows, and so on.

		Actual		Budget	
		Qtr1	Qtr2	Qtr1	Qtr2
Florida	100-10	570	670	570	650
	100-20	235	345	321	432
	100-30	655	555	455	865
	100-40	342	342	432	234
Maine	100-10	600	800	800	750
	100-20	734	334	734	534
	100-30	324	321	235	278
	100-40	432	342	289	310
New York	100-10	1010	1210	1110	910
	100-20	960	760	650	870
	100-30	324	550	432	321
	100-40	880	980	880	1080
	100-50	#MI	#MI	#MI	#MI

This example returns rows with the highest data values in *col2* (Actual, Qtr2) per report, because the row grouping is the “`market`”.

```
2- TOP("market", 3, @DATACOL(2))
```

This command returns the following rows:

New York	100-10	1010	1210	1110	910
	100-40	880	980	880	1080
Maine	100-10	600	800	800	750

This example returns two rows with the lowest data values in *col2* (Actual, Qtr2) per row group.

```
3- BOTTOM ("market", 2, @DATACOL(2) )
```

This command returns the following rows:

```
Maine          100-20      734        334        734        534
               100-30      324        321        235        278
```

Note: TOP and BOTTOM put an upper limit on the number of (qualified) rows returned after all restrictions are applied. This upper limit is equal to the number of rows in the TOP plus the number of rows in the BOTTOM commands.

How Other Report Configurations Affect Row Specifications

When using the TOP and BOTTOM commands, be aware that some other commands affect their operation. In particular, Essbase treats the SUPPMISSING, SUPZEROS, and SUPEMPTYROWS options as restrictions and applies them to the extracted rows along with the RESTRICT command restrictions. Essbase applies these optional restrictions to the data rows before processing the TOP or BOTTOM commands, and before applying an ORDERBY command.

Using TOP and BOTTOM with Formatting Commands

Whenever a formatting command occurs in a report, it is appended to the member that follows it. For example, in this sequence, {UCOL}, the underline columns command is appended internally to the member that comes next. In Script 1, it is appended to the row member that can be described as "first child of market, assuming FLORIDA."

```
SCRIPT 1              SCRIPT 2
....                  ....
< ROW MARKET          {UCOL}
{UCOL }               < FLORIDA    (row member)
<ICHILD MARKET
< TOP ...             < BOTTOM ....
```

Script 2, appends {UCOL} to the row member FLORIDA. Essbase executes {UCOL} whenever it encounters a row that has row member FLORIDA. If the TOP or BOTTOM command returns a row that does not contain FLORIDA, the formatting commands appended to the rows are never executed.

Because of this, it is a good idea to place all general formatting commands before a COLUMN command, or a command that expands into column members. This guarantees that the formatting is executed. However, you should not use formatting commands that work on rows, because these rows may never be picked up by the TOP or BOTTOM command. Also avoid using SAVEROW and CALCROW with the TOP and BOTTOM commands.

Converting Data to a Different Currency

If your database has a currency partition, you can calculate currency conversions in report scripts. Use the CURRENCY command to set the output currency and currency type. Use the CURHEADING command to display the currency conversion heading.

Note: Currency conversion is not supported across transparent partitions.

For information about creating a currency conversion application, see Chapter 42, Designing and Building Currency Conversion Applications.

For the syntax and definitions of Report Writer commands, see the online *Technical Reference* in your `DOCS` directory.

Generating Reports Using the C, Visual Basic, and Grid APIs

Use the following table to determine the report API calls that you can make:

To Do This...	Use This C API Function	Use This Visual Basic API Function	Use This C Grid API Function
Start sending a report specification to the active database.	ESSBEGINREPORT	ESBBEGINREPORT	ESSGBEGINREPORT
Mark the end of a report specification being sent to the active database.	ESSENDREPORT	ESBENDREPORT	N/A
Send a report specification to the active database as a single string.	ESSREPORT	ESBREPORT	N/A
Send a report specification to the active database from a file.	ESSREPORTFILE	ESBREPORTFILE	ESSGREPORTFILE

See the online *API Reference* in your `DOCS` directory for syntax and descriptions of these API functions.

Chapter 36 Examples of Report Scripts

This chapter includes report scripts that demonstrate report procedures and formats that are most frequently required in business settings. If you examine the techniques in these scripts and the resulting output, you can adapt them for use in your own reports.

The samples use both the Demo Basic and Sample Basic databases provided with your Hyperion Essbase server. Each sample identifies the database used. The scripts for these examples are available in your `\ESSBASE\APP\DEMO\BASIC` directory or your `\ESSBASE\APP\SAMPLE\BASIC` directory.

The sample reports in this chapter demonstrate the following techniques:

For fundamental information about reports and report scripts, see Chapter 34, Quick Start to Report Scripts. For detailed information about using Report Writer commands to write reports and reports scripts, see Chapter 35, Developing Report Scripts. For the syntax and usage of each Report Writer command, see the online *Technical Reference* in your `DOCS` directory.

Sample 1: Creating a Different Format for Each Page

This sample report contains data for Actual Sales. Each report page shows a different Product. The report lists products on the same page until the maximum page length is reached. To place each Product on a separate page, you must use the PAGEONDIMENSION format command, as shown in Sample 2.

Because none of the cities in South sell Stereo or Compact_Disc, the data values indicate `#MISSING`. You can represent missing values by suppressing the row or substituting a replacement text string, such as "N/A." See "Sample 2: Handling Missing Values" on page 36-4 for an example of substituting page breaks and labels for missing values.

```
                        Sales Actual Stereo

                    Qtr1     Qtr2     Qtr3     Qtr4
                ======== ======== ======== ========

East               7,839    7,933    7,673   10,044
West              11,633   11,191   11,299   14,018
South            #Missing #Missing #Missing #Missing
  Market          19,472   19,124   18,972   24,062

                     Sales Actual Compact_Disc

                    Qtr1     Qtr2     Qtr3     Qtr4
                ======== ======== ======== ========

East              10,293    9,702    9,965   11,792
West              14,321   14,016   14,328   17,247
South            #Missing #Missing #Missing #Missing
  Market          24,614   23,718   24,293   29,039

                           Sales Actual Audio

                    Qtr1     Qtr2     Qtr3     Qtr4
                ======== ======== ======== ========

East              18,132   17,635   17,638   21,836
West              25,954   25,207   25,627   31,265
South            #Missing #Missing #Missing #Missing
  Market          44,086   42,842   43,265   53,101
```

Use the following script to create Sample 1:

```
<PAGE (Accounts, Scenario, Product)
Sales
Actual
<IDESC Audio

    <COLUMN (Year)
    <CHILD Year

<ROW(Market)
<ICHILD Market
      !
```

The ! report output command is required to generate the report.

Because the IDESC selection command is used for Audio, the report selects all three members. Only a single member is selected from the other page dimensions, Sales and Actual. As a result, the script creates three report pages. They appear as one long report page unless you use the PAGEONDIMENSION format command, as shown in Sample 2.

This report script, `ACTSALES.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 2: Handling Missing Values

This report has the same layout and member selection as Sample 1, and shows you how to use page breaks and labels for missing values.

```
                    Sales Actual Stereo

                    Qtr1     Qtr2     Qtr3     Qtr4
                ======== ======== ======== ========

East               7,839    7,933    7,673   10,044
West              11,633   11,191   11,299   14,018
South                N/A      N/A      N/A      N/A
  Market          19,472   19,124   18,972   24,062
```

```
                    Sales Actual Compact_Disc

                    Qtr1     Qtr2     Qtr3     Qtr4
                ======== ======== ======== ========

East              10,293    9,702    9,965   11,792
West              14,321   14,016   14,328   17,247
South                N/A      N/A      N/A      N/A
  Market          24,614   23,718   24,293   29,039
```

```
                          Sales Actual Audio

                     Qtr1     Qtr2     Qtr3     Qtr4
                 ======== ======== ======== ========

East               18,132   17,635   17,638   21,836
West               25,954   25,207   25,627   31,265
South                 N/A      N/A      N/A      N/A
  Market           44,086   42,842   43,265   53,101
```

Use the following script to create Sample 2:

```
<PAGE (Accounts, Scenario, Product)
Sales
Actual
<IDESC Product
{ PAGEONDIMENSION Product }
{ MISSINGTEXT "N/A" }

      <COLUMN (Year)
      <CHILD Year

<ROW(Market)
<ICHILD Market
      !
```

The PAGEONDIMENSION format command creates a page break whenever a member from the specified dimension changes. Because the report selects eight Product members, this creates an eight-page report.

The MISSINGTEXT format command substitutes any strings enclosed within double quotes into the `#MISSING` string. To suppress missing values, use the SUPMISSINGROWS command.

You can also combine format commands within one set of braces:

```
{ PAGEONDIMENSION Product MISSINGTEXT "N/A" }
```

This report script, `MISS_LBL.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 3: Nesting Columns

Each page produced by this report sample contains Sales information for a given Market. The report has two groups of columns across the page. The Actual and Budget members are the nested column group below Year members.

Note that the Actual and Budget members are on the same line in the report. You can put multiple commands on one line, but report commands are easier to read if they are spread out.

Sales East

	Jan		Feb		Mar		Qtr1	
	Actual	Budget	Actual	Budget	Actual	Budget	Actual	Budget
Stereo	2,788	2,950	2,482	2,700	2,569	2,700	7,839	8,350
Compact_Disc	3,550	3,450	3,285	3,250	3,458	3,250	10,293	9,950
Audio	6,338	6,400	5,767	5,950	6,027	5,950	18,132	18,300
Television	5,244	4,800	4,200	4,300	3,960	4,300	13,404	13,400
VCR	4,311	4,200	3,734	3,700	3,676	3,700	11,721	11,600
Camera	2,656	2,850	2,525	2,670	2,541	2,670	7,722	8,190
Visual	12,211	11,850	10,45	10,670	10,177	10,670	32,847	33,190
Product	18,549	18,250	16,226	16,620	16,204	16,620	50,979	51,490

Sales West

	Jan		Feb		Mar		Qtr1	
	Actual	Budget	Actual	Budget	Actual	Budget	Actual	Budget
Stereo	4,102	4,000	3,723	3,600	3,808	3,600	11,633	11,200
Compact_Disc	4,886	4,700	4,647	4,400	4,788	4,400	14,321	13,500
Audio	8,988	8,700	8,370	8,000	8,596	8,000	25,954	24,700
Television	5,206	5,100	4,640	4,600	4,783	4,600	14,629	14,300
VCR	4,670	4,650	4,667	4,200	4,517	4,200	13,854	13,050
Camera	3,815	4,050	3,463	3,750	3,478	3,750	10,756	11,550
Visual	13,691	13,800	12,770	12,550	12,778	12,550	39,239	38,900
Product	22,679	22,500	21,140	20,550	21,374	20,550	65,193	63,600

```
                                            Sales South

                 Jan                  Feb                  Mar                Qtr1
            Actual  Budget      Actual  Budget      Actual  Budget      Actual   Budget
          ======== =======   ======= ======= = ====== ======= ======== ========

Television  3,137   3,400     2,929   3,100     2,815   3,100     8,881    9,600
VCR         3,225   3,400     3,206   3,100     3,120   3,100     9,551    9,600
Camera      2,306   2,400     2,167   2,400     2,168   2,400     6,641    7,200
  Visual    8,668   9,200     8,302   8,600     8,103   8,600    25,073   26,400
    Product 8,668   9,200     8,302   8,600     8,103   8,600    25,073   26,400
```

```
                                          Sales Market

                    Jan                  Feb                  Mar                  Qtr1
               Actual  Budget      Actual  Budget      Actual  Budget      Actual  Budget
               ====== =======    ======= ======= ======== ======= ========   ======

Stereo          6,890   6,950      6,205   6,300      6,377   6,300     19,472  19,550
Compact_Disc    8,436   8,150      7,932   7,650      8,246   7,650     24,614  23,450
  Audio        15,326  15,100     14,137  13,950     14,623  13,950     44,086  43,000
Television     13,587  13,300     11,769  12,000     11,558  12,000     36,914  37,300
VCR            12,206  12,250     11,607  11,000     11,313  11,000     35,126  34,250
Camera          8,777   9,300      8,155   8,820      8,187   8,820     25,119  26,940
  Visual       34,570  34,850     31,531  31,820     31,058  31,820     97,159  98,490
    Product    49,896  49,950     45,668  45,770     45,681  45,770    141,245 141,490
```

Use the following script to create Sample 3:

```
<PAGE (Accounts, Market)
Sales
<ICHILD Market
{ PAGEONDIMENSION Market }
{ SUPMISSINGROWS }

      <COLUMN (Year, Scenario)
      <ICHILD Qtr1
       Actual Budget

<ROW(Product)
<IDESC Product
      !
```

The report selects four Markets because the <ICHILD command is applied to Market. Only Sales is selected from the other page dimension, so the report has four pages.

For the South, all the rows of Product data do not appear. Recall that the cities in the South do not sell every Product. The report uses the SUPMISSINGROWS format command to suppress the output of any member rows with all missing values.

This report script, `COLGROUP.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 4: Grouping Rows

Each page of this report contains Sales information for a given Market. The report page contains members for both Product and Year as groups of rows down the page. This script creates a four-page report because the page dimensions and their member selections are the same as in Sample 3. The row/column layout is switched because the row and column dimensions are different.

		Sales East		
		Actual	Budget	Variance
Stereo	Qtr1	7,839	8,350	(511)
	Qtr2	7,933	8,150	(217)
	Qtr3	7,673	8,350	(677)
	Qtr4	10,044	10,400	(356)
	Year	33,489	35,250	(1,761)
Compact_Disc	Qtr1	10,293	9,950	343
	Qtr2	9,702	9,750	(48)
	Qtr3	9,965	10,050	(85)
	Qtr4	11,792	12,550	(758)
	Year	41,752	42,300	(548)
Audio	Qtr1	18,132	18,300	(168)
	Qtr2	17,635	17,900	(265)
	Qtr3	17,638	18,400	(762)
	Qtr4	21,836	22,950	(1,114)
	Year	75,241	77,550	(2,309)
Television	Qtr1	13,404	13,400	4
	Qtr2	12,115	12,900	(785)
	Qtr3	15,014	14,200	814
	Qtr4	17,861	17,300	561
	Year	58,394	57,800	594
VCR	Qtr1	11,721	11,600	121
	Qtr2	10,999	11,100	(101)
	Qtr3	13,217	11,800	1,417
	Qtr4	14,386	14,900	(514)
	Year	50,323	49,400	923
Camera	Qtr1	7,722	8,190	(468)
	Qtr2	7,581	8,210	(629)
	Qtr3	8,181	8,630	(449)
	Qtr4	10,853	11,550	(697)
	Year	34,337	36,580	(2,243)
Visual	Qtr1	32,847	33,190	(343)
	Qtr2	30,695	32,210	(1,515)
	Qtr3	36,412	34,630	1,782
	Qtr4	43,100	43,750	(650)
	Year	143,054	143,780	(726)
Product	Qtr1	50,979	51,490	(511)
	Qtr2	48,330	50,110	(1,780)
	Qtr3	54,050	53,030	1,020
	Qtr4	64,936	66,700	(1,764)
	Year	218,295	221,330	(3,035)

		Sales West		
		Actual	Budget	Variance
Stereo	Qtr1	11,633	11,200	433
	Qtr2	11,191	11,050	141
	Qtr3	11,299	11,650	(351)
	Qtr4	14,018	14,500	(482)
	Year	48,141	48,400	(259)
Compact_Disc	Qtr1	14,321	13,500	821
	Qtr2	14,016	13,500	516
	Qtr3	14,328	14,300	28
	Qtr4	17,247	16,700	547
	Year	59,912	58,000	1,912
Audio	Qtr1	25,954	24,700	1,254
	Qtr2	25,207	24,550	657
	Qtr3	25,627	25,950	(323)
	Qtr4	31,265	31,200	65
	Year	108,053	106,400	1,653
Television	Qtr1	14,629	14,300	329
	Qtr2	14,486	13,800	686
	Qtr3	14,580	14,000	580
	Qtr4	20,814	19,400	1,414
	Year	64,509	61,500	3,009
VCR	Qtr1	13,854	13,050	804
	Qtr2	13,156	12,600	556
	Qtr3	15,030	13,750	1,280
	Qtr4	18,723	17,950	773
	Year	60,763	57,350	3,413
Camera	Qtr1	10,756	11,550	(794)
	Qtr2	10,573	11,400	(827)
	Qtr3	10,735	11,550	(815)
	Qtr4	13,906	15,000	(1,094)
	Year	45,970	49,500	(3,530)
Visual	Qtr1	39,239	38,900	339
	Qtr2	38,215	37,800	415
	Qtr3	40,345	39,300	1,045
	Qtr4	53,443	52,350	1,093
	Year	171,242	168,350	2,892
Product	Qtr1	65,193	63,600	1,593
	Qtr2	63,422	62,350	1,072
	Qtr3	65,972	65,250	722
	Qtr4	84,708	83,550	1,158
	Year	279,295	274,750	4,545

		Sales South		
		Actual	Budget	Variance
Television	Qtr1	8,881	9,600	(719)
	Qtr2	8,627	9,300	(673)
	Qtr3	8,674	9,300	(626)
	Qtr4	12,919	12,600	319
	Year	39,101	40,800	(1,699)
VCR	Qtr1	9,551	9,600	(49)
	Qtr2	9,049	9,300	(251)
	Qtr3	9,998	10,000	(2)
	Qtr4	12,923	13,600	(677)
	Year	41,521	42,500	(979)
Camera	Qtr1	6,641	7,200	(559)
	Qtr2	6,765	7,350	(585)
	Qtr3	6,798	7,500	(702)
	Qtr4	9,486	10,200	(714)
	Year	29,690	32,250	(2,560)
Visual	Qtr1	25,073	26,400	(1,327)
	Qtr2	24,441	25,950	(1,509)
	Qtr3	25,470	26,800	(1,330)
	Qtr4	35,328	36,400	(1,072)
	Year	110,312	115,550	(5,238)
Product	Qtr1	25,073	26,400	(1,327)
	Qtr2	24,441	25,950	(1,509)
	Qtr3	25,470	26,800	(1,330)
	Qtr4	35,328	36,400	(1,072)
	Year	110,312	115,550	(5,238)

		Sales Market		
		Actual	Budget	Variance
Stereo	Qtr1	19,472	19,550	(78)
	Qtr2	19,124	19,200	(76)
	Qtr3	18,972	20,000	(1,028)
	Qtr4	24,062	24,900	(838)
	Year	81,630	83,650	(2,020)
Compact_Disc	Qtr1	24,614	23,450	1,164
	Qtr2	23,718	23,250	468
	Qtr3	24,293	24,350	(57)
	Qtr4	29,039	29,250	(211)
	Year	101,664	100,300	1,364
Audio	Qtr1	44,086	43,000	1,086
	Qtr2	42,842	42,450	392
	Qtr3	43,265	44,350	(1,085)
	Qtr4	53,101	54,150	(1,049)
	Year	183,294	183,950	(656)
Television	Qtr1	36,914	37,300	(386)
	Qtr2	35,228	36,000	(772)
	Qtr3	38,268	37,500	768
	Qtr4	51,594	49,300	2,294
	Year	162,004	160,100	1,904
VCR	Qtr1	35,126	34,250	876
	Qtr2	33,204	33,000	204
	Qtr3	38,245	35,550	2,695
	Qtr4	46,032	46,450	(418)
	Year	152,607	149,250	3,357
Camera	Qtr1	25,119	26,940	(1,821)
	Qtr2	24,919	26,960	(2,041)
	Qtr3	25,714	27,680	(1,966)
	Qtr4	34,245	36,750	(2,505)
	Year	109,997	118,330	(8,333)
Visual	Qtr1	97,159	98,490	(1,331)
	Qtr2	93,351	95,960	(2,609)
	Qtr3	102,227	100,730	1,497
	Qtr4	131,871	132,500	(629)
	Year	424,608	427,680	(3,072)
Product	Qtr1	141,245	141,490	(245)
	Qtr2	136,193	138,410	(2,217)
	Qtr3	145,492	145,080	412
	Qtr4	184,972	186,650	(1,678)
	Year	607,902	611,630	(3,728)

Use the following script to create Sample 4:

```
<PAGE (Accounts, Market)
Sales
<ICHILD Market
{ PAGEONDIMENSION Market }
{ SUPMISSINGROWS }

      <COLUMN (Scenario)
      <CHILD Scenario

<ROW(Product3, Year)
<ICHILD Year
<IDESC Product
      !
```

This report script, `ROWGROUP.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 5: Reporting on Different Combinations of Data

Each page represents a different combination of Product, Market, and Budget data. The total number of pages is determined by the number of Market and Product members. This section shows a representative part of the output.

Some data values have four decimal places. The number of decimal places, by default, is output to the true number of decimal values of the data cell. "Sample 6: Formatting Different Combinations of Data" on page 36-16 uses the DECIMAL format command to define a specific number of places.

The member selection commands select three Product members and fourteen Market members. This produces a 42-page report. The number of report pages is determined by multiplying the number of members selected from each page dimension.

Budget Audio New_York

	Qtr1	Qtr2	Qtr3	Qtr4	Year
Sales	6,400	6,400	6,700	8,350	27,850
Cost_of_Goods_Sold	3,012	3,012	3,146	3,973	13,143
Margin	3,388	3,388	3,554	4,377	14,707
Marketing	525	515	475	555	2,070
Payroll	1,950	1,950	1,950	1,950	7,800
Miscellaneous	0	0	0	0	0
Total_Expenses	2,475	2,465	2,425	2,505	9,870
Profit	913	923	1,129	1,872	4,837
Profit_%	14	14	17	22	17
Margin_%	53	53	53	52	53

Budget Audio Boston

	Qtr1	Qtr2	Qtr3	Qtr4	Year
Sales	6,050	5,750	5,900	7,350	25,050
Cost_of_Goods_Sold	2,829	2,695	2,762	3,413	11,699
Margin	3,221	3,055	3,138	3,937	13,351
Marketing	410	400	400	520	1,730
Payroll	1,590	1,590	1,590	1,590	6,360
Miscellaneous	0	0	0	0	0
Total_Expenses	2,000	1,990	1,990	2,110	8,090
Profit	1,221	1,065	1,148	1,827	5,261
Profit_%	20	19	19	25	21
Margin_%	53	53	53	54	53

```
                          Budget Product Market

                    Qtr1     Qtr2     Qtr3     Qtr4     Year
                ======== ======== ======== ======== ========

Sales             141,490  138,410  145,080  186,650  611,630
Cost_of_Goods_Sold 55,860   54,579   57,379   73,276  241,093
  Margin           85,630   83,831   87,702  113,374  370,537
Marketing          10,555   10,680   10,780   13,915   45,930
Payroll            43,234   43,248   43,248   43,248  172,978
Miscellaneous           0        0        0        0        0
  Total_Expenses   53,789   53,928   54,028   57,163  218,908
    Profit         31,841   29,903   33,674   56,211  151,629
    Profit_%           23       22       23       30       25
    Margin_%           61       61       60       61       61
```

Use the following script to create Sample 5:

```
<PAGE (Scenario, Product, Market)
Budget
<ICHILD Product
<IDESC Market
{ PAGEONDIMENSION Product }   // New page at each new Product
{ PAGEONDIMENSION Market }    // New page at each new Market
       <COLUMN (Year)
       <ICHILD Year

<ROW(Accounts)
<DESC Accounts
      !
```

This report script, `COMBO1.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 6: Formatting Different Combinations of Data

This report uses the same layout and member selection as Sample 5, and adds more formatting in the report body. Note the use of line formatting.

Budget Audio New_York

	Qtr1	Qtr2	Qtr3	Qtr4	Year
Sales	6,400	6,400	6,700	8,350	27,850
Cost_of_Goods_Sold	3,012	3,012	3,146	3,973	13,143
Margin	3,388	3,388	3,554	4,377	14,707
Marketing	525	515	475	555	2,070
Payroll	1,950	1,950	1,950	1,950	7,800
Miscellaneous	0	0	0	0	0
Total_Expenses	2,475	2,465	2,425	2,505	9,870
Profit	913	923	1,129	1,872	4,837
Profit_%	14.27	14.42	16.85	22.42	17.37
Margin_%	52.94	52.94	53.04	52.42	52.81

```
                        Budget Audio Boston

                    Qtr1     Qtr2     Qtr3     Qtr4     Year
                ======== ======== ======== ======== ========

Sales              6,050    5,750    5,900    7,350   25,050
Cost_of_Goods_Sold 2,829    2,695    2,762    3,413   11,699
                -------- -------- -------- -------- --------
  Margin           3,221    3,055    3,138    3,937   13,351

Marketing            410      400      400      520    1,730
Payroll            1,590    1,590    1,590    1,590    6,360
Miscellaneous          0        0        0        0        0
                -------- -------- -------- -------- --------
  Total_Expenses   2,000    1,990    1,990    2,110    8,090

    Profit         1,221    1,065    1,148    1,827    5,261
                ======== ======== ======== ======== ========
    Profit_%       20.18    18.52    19.46    24.86    21.00
    Margin_%       53.24    53.13    53.19    53.56    53.30
```

Use the following script to create Sample 6:

```
<PAGE (Scenario, Product, Market)
{ PAGEONDIMENSION Product PAGEONDIMENSION Market }
Budget
<ICHILD Product
<IDESC Market

      <COLUMN (Year)
      <ICHILD Year

<ROW(Accounts)

{ SUPBRACKET DECIMAL 0 }
Sales
Cost_of_Goods_Sold

{ UDATA "-" }      //line formatting command
Margin

{ SKIP }
Marketing
Payroll
Miscellaneous
```

```
{ UDATA "-" }      //line formatting command
Total_Expenses

{ SKIP }
Profit

{ UDATA DECIMAL 2 }    //line formatting command
Profit_%
Margin_%
    !
```

Format commands apply to members that follow the commands. The report begins each new page with the formats in place at the end of the previous report page. For example, if a report page ends with two decimal places, the following page begins with two decimal places. This report demonstrates the use of several important format commands:

- DECIMAL. The script for this report specifies the DECIMAL 0 format command before the Sales member.
- SUPBRACKET. By default, negative numbers are enclosed in brackets, ( ). The SUPBRACKET format command causes negative numbers to be output with a minus sign.
- UDATA. The UDATA command places underline characters under data columns. The character is specified within double quotes. The default is a double underline.

This report script, `COMBO2.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 7: Using Alternate Names

This report outputs members in the middle of a page and uses alternate names or aliases. The default row member indentation is turned off.

```
                          Stereo Market

       Qtr4                                    Year
  Actual   Budget                          Actual   Budget
======== ========                        ======== ========

  24,062   24,900 Sales                    81,630   83,650
  13,937   14,442 COGS                     47,654   48,517
-------- --------                        -------- --------
  10,125   10,458 Margin                   33,976   35,133

   1,438    1,600 Marketing                 4,933    5,465
   7,110    6,840 Payroll                  28,440   27,360
    -200        0 Misc.                     -143        0
-------- --------                        -------- --------
   8,348    8,440 Total_Expenses           33,230   32,825

   1,777    2,018 Profit                      746    2,308
======== ========                        ======== ========
    7.39     8.10 Profit_%                   0.91     2.76
   42.08    42.00 Margin_%                  41.62    42.00
```

```
                          Compact_Disc Market

       Qtr4                                  Period
    Actual   Budget                       Actual   Budget
  ======== ========                     ======== ========

    29,039   29,250 Sales                101,664  100,300
    10,830   11,115 COGS                  38,120   38,114
  -------- --------                     -------- --------
    18,209   18,135 Margin                63,544   62,186
     1,669    1,780 Marketing              6,067    5,975
     5,721    5,415 Payroll               22,200   21,660
      -226        0 Misc.                     97        0
  -------- --------                     -------- --------
     7,164    7,195 Total_Expenses        28,364   27,635

    11,045   10,940 Profit                35,180   34,551
  ======== ========                     ======== ========
     38.04    37.40 Profit_%               34.60    34.45
     62.71    62.00 Margin_%               62.50    62.00
```

Use the following script to create Sample 7:

```
<PAGE (Product, Market)
{ PAGEONDIMENSION Product }
{ PAGEONDIMENSION Market }
<IDESC Product
{ DECIMAL 0 }
<SYM
     <COLUMN (Year, Scenario)
     Qtr4 Year
     Actual Budget
<ROW(Accounts)
{ SUPBRACKET OUTALTNAMES NOINDENTGEN ORDER 1,2,0,3,4 }
Sales Cost_of_Goods_Sold
{ UDATA "-" }
Margin
{ SKIP }
Marketing Payroll Miscellaneous
{ UDATA "-" }
Total_Expenses
{ SKIP }
Profit
{ UDATA DECIMAL 2 }
Profit_%
Margin_%
     !
```

The SYM command forces the report to output symmetric column groups. The default is to display two columns: one for Qtr4 Actual and one for Year Budget. Because the report calls for Actual and Budget under both Qtr4 and Year, the SYM command is required. Alternatively, repeat the Actual and Budget names under Qtr4 and Year.

The OUTALTNAMES format command causes the report to use alternate names or aliases instead of member names.

The NOINDENTGEN format command causes row members to not be indented. By default, members are indented two spaces for each level.

The ORDER command moves specified output columns to new locations. The row name is considered column 0.

The FIXCOLUMNS format command restricts the number of output columns. Reports often require both ORDER and FIXCOLUMNS. You can use ORDER to remove unwanted columns, and FIXCOLUMNS to stop these columns from appearing after the report columns.

This report script, `MIDDLE.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 8: Creating Custom Headings and % Characters

This report displays custom headings and percent sign (%) characters after each data value. This section shows a representative part of the output.

Prepared by: Admin The Electronics Club Page: 1
09/21/95

Profit_% Actual Stereo

	Jan	Feb	Mar	Apr	May	Jun
New_York	1.43%	-10.00%	-3.51%	-2.22%	1.14%	-6.18%
Boston	-0.34%	-2.51%	-4.44%	-4.89%	-7.02%	-13.15%
Chicago	-0.65%	-0.72%	-2.28%	-3.53%	-6.33%	-10.79%
East	0.18%	-4.47%	-3.39%	-3.41%	-3.60%	-9.70%
San_Francisco	1.43%	-1.87%	4.42%	2.15%	-1.26%	0.66%
Seattle	0.95%	-5.66%	1.42%	-6.82%	-11.47%	-12.34%
Denver	3.03%	-1.11%	-5.88%	-6.52%	-5.17%	-13.83%
Los_Angeles	-1.50%	-3.94%	-2.86%	-3.29%	3.12%	-2.51%
West	0.98%	-2.95%	-0.13%	-2.81%	-2.62%	-5.61%
Dallas	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Houston	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Phoenix	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
South	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Market	0.65%	-3.56%	-1.44%	-3.06%	-3.03%	-7.29%

Prepared by: Admin The Electronics Club Page: 2
09/21/95

Profit_% Actual Compact_Disc

	Jan	Feb	Mar	Apr	May	Jun
New_York	32.51%	29.95%	35.30%	32.70%	30.45%	31.73%
Boston	33.42%	27.92%	33.98%	30.74%	27.45%	30.85%
Chicago	34.29%	30.48%	26.33%	28.83%	28.11%	33.76%
East	33.35%	29.50%	32.30%	30.92%	28.77%	32.09%
San_Francisco	37.77%	35.02%	33.41%	33.23%	35.32%	37.95%
Seattle	40.41%	38.33%	38.89%	37.06%	37.01%	38.29%
Denver	31.93%	32.10%	34.82%	29.15%	32.71%	30.85%
Los_Angeles	31.65%	30.22%	30.22%	31.45%	27.06%	33.20%
West	35.51%	33.94%	34.21%	32.77%	33.16%	35.25%
Dallas	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Houston	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Phoenix	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
South	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Market	34.60%	32.10%	33.41%	32.01%	31.35%	33.97%

Prepared by: Admin The Electronics Club Page: 3
09/21/95

Profit_% Actual Audio

	Jan	Feb	Mar	Apr	May	Jun
New_York	19.35%	13.64%	18.64%	16.55%	16.70%	14.65%
Boston	18.34%	14.44%	18.94%	14.94%	12.14%	12.42%
Chicago	18.50%	16.67%	13.18%	14.12%	12.70%	13.74%
East	18.76%	14.88%	17.09%	15.32%	14.05%	13.68%
San_Francisco	20.32%	17.38%	18.92%	18.03%	18.23%	20.57%
Seattle	23.36%	21.40%	23.37%	20.17%	18.82%	19.04%
Denver	18.36%	17.25%	18.88%	13.43%	15.84%	12.14%
Los_Angeles	17.15%	14.76%	15.44%	15.76%	15.10%	17.07%
West	19.75%	17.53%	19.00%	16.88%	17.01%	17.52%
Dallas	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Houston	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Phoenix	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
South	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Market	19.34%	16.45%	18.21%	16.24%	15.78%	15.96%

```
Prepared by: Admin                The Electronics Club            Page: 8
                                                                 09/21/95

                              Profit_% Actual Product

                     Jan      Feb      Mar      Apr      May      Jun
                  =======  =======  =======  =======  =======  =======

New_York           22.71%   21.43%   13.11%   10.54%    9.73%   13.16%
Boston             24.98%   23.25%   19.95%   18.00%   17.03%   18.62%
Chicago            22.01%   17.94%   18.14%   15.45%   18.70%   16.01%
  East             23.19%   20.84%   16.89%   14.42%   14.94%   15.78%
San_Francisco      23.71%   20.60%   21.93%   20.45%   21.44%   19.98%
Seattle            21.06%   21.05%   21.24%   19.00%   21.72%   15.13%
Denver             21.61%   16.01%   19.79%   14.81%   20.66%   13.89%
Los_Angeles        17.54%   15.51%   17.03%   14.33%   17.59%   16.09%
  West             21.02%   18.35%   19.99%   17.26%   20.30%   16.61%
Dallas             15.67%   16.50%   15.32%   13.93%   20.36%   15.49%
Houston            20.01%   20.29%   20.62%   15.87%   23.60%   12.38%
Phoenix            20.01%   16.12%   17.18%   16.50%   21.39%   15.22%
  South            18.39%   17.53%   17.59%   15.36%   21.66%   14.46%
    Market         21.37%   19.09%   18.46%   15.92%   18.67%   15.93%
```

Use the following script to create Sample 8:

```
<PAGE (Accounts, Scenario, Product)
{ PAGEONDIMENSION Product }       // New page when Product changes
Profit_%
Actual
<IDESC Product
      <COLUMN (Year)
      Jan Feb Mar Apr May Jun
<ROW(Market)
{ STARTHEADING
TEXT   1 "Prepared by:"
      14 "*USERNAME"
       C "The Electronics Club"
      65 "*PAGESTRING"
TEXT  65 "*DATE"
SKIP
ENDHEADING }
{ Decimal 2 AFTER "%" SUPBRACKET }   // Place % at end and
                                     // suppress bracket
<IDESC Market
      !
```

Each data value in the report has a percent sign, %. This label is defined with the `AFTER "%"` format command. You can specify any character within double quotes.

This report has custom headings at the top of each page. All format commands specified between the STARTHEADING and ENDHEADING format commands appear at the top of each report page.

TEXT format commands define text labels. The report generator provides *dynamic* text with **options*. See the online *Technical Reference* in your `DOCS` directory for a full list of the available options. This report uses the following options:

- *USERNAME, which outputs the user name used when connecting to Essbase
- *PAGESTRING, which outputs the current page number of the report
- C, which centers the report title

This report script, `HEADING1.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 9: Creating Custom Page Headings

This report builds on Sample 8 by adding custom page headings. By default, page dimension members are output at the top center of a report page.

Prepared by :admin — The Electronics Club — Page: 1
Actual Profit by Product — 12/12/95

Product: Stereo

	Jan	Feb	Mar	Apr	May	Jun
New York	1.43%	-10.00%	-3.51%	-2.22%	1.14%	-6.18%
Boston	-0.34%	-2.51%	-4.44%	-4.89%	-7.02%	-13.15%
Chicago	-0.65%	-0.72%	-2.28%	-3.53%	-6.33%	-10.79%
San Francisco	1.43%	-1.87%	4.42%	2.15%	-1.26%	0.66%
Seattle	0.95%	-5.66%	1.42%	-6.82%	-11.47%	-12.34%
Denver	3.03%	-1.11%	-5.88%	-6.52%	-5.17%	-13.83%
Los Angeles	-1.50%	-3.94%	-2.86%	-3.29%	3.12%	-2.51%
Dallas	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Houston	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Phoenix	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
East	0.18%	-4.47%	-3.39%	-3.41%	-3.60%	-9.70%
West	0.98%	-2.95%	-0.13%	-2.81%	-2.62%	-5.61%
South	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Market	0.65%	-3.56%	-1.44%	-3.06%	-3.03%	-7.29%

Prepared by :admin — The Electronics Club — Page: 2
Actual Profit by Product — 12/12/95

Pageduct:Compact Disc

	Jan	Feb	Mar	Apr	May	Jun
New York	32.51%	29.95%	35.30%	32.70%	30.45%	31.73%
Boston	33.42%	27.92%	33.98%	30.74%	27.45%	30.85%
Chicago	34.29%	30.48%	26.33%	28.83%	28.11%	33.76%
San Francisco	37.77%	35.02%	33.41%	33.23%	35.32%	37.95%
Seattle	40.41%	38.33%	38.89%	37.06%	37.01%	38.29%
Denver	31.93%	32.10%	34.82%	29.15%	32.71%	30.85%
Los Angeles	31.65%	30.22%	30.22%	31.45%	27.06%	33.20%
Dallas	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Houston	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Phoenix	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
East	33.35%	29.50%	32.30%	30.92%	28.77%	32.09%
West	35.51%	33.94%	34.21%	32.77%	33.16%	35.25%
South	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Market	34.60%	32.10%	33.41%	32.01%	31.35%	33.97%

Prepared by :admin — The Electronics Club — Page: 3
Actual Profit by Product — 12/12/95

Product: Audio

	Jan	Feb	Mar	Apr	May	Jun
New York	19.35%	13.64%	18.64%	16.55%	16.70%	14.65%
Boston	18.34%	14.44%	18.94%	14.94%	12.14%	12.42%
Chicago	18.50%	16.67%	13.18%	14.12%	12.70%	13.74%
San Francisco	20.32%	17.38%	18.92%	18.03%	18.23%	20.57%
Seattle	23.36%	21.40%	23.37%	20.17%	18.82%	19.04%
Denver	18.36%	17.25%	18.88%	13.43%	15.84%	12.14%
Los Angeles	17.15%	14.76%	15.44%	15.76%	15.10%	17.07%
Dallas	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Houston	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Phoenix	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
East	18.76%	14.88%	17.09%	15.32%	14.05%	13.68%
West	19.75%	17.53%	19.00%	16.88%	17.01%	17.52%
South	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
Market	19.34%	16.45%	18.21%	16.24%	15.78%	15.96%

Prepared by :admin The Electronics Club Page: 4
Actual Profit by Product 12/12/95

Product:Television

	Jan	Feb	Mar	Apr	May	Jun
New York	19.45%	28.80%	3.29%	-4.14%	-1.98%	7.73%
Boston	34.53%	40.00%	24.92%	19.20%	15.00%	22.31%
Chicago	20.61%	12.86%	15.29%	7.36%	19.23%	10.29%
San Francisco	16.53%	12.59%	17.52%	10.74%	18.70%	7.43%
Seattle	9.22%	12.29%	14.89%	18.39%	22.94%	7.87%
Denver	17.69%	0.24%	16.39%	10.75%	21.05%	12.36%
Los Angeles	10.41%	6.25%	10.48%	3.00%	14.52%	5.34%
Dallas	2.24%	6.23%	5.89%	5.38%	15.48%	4.75%
Houston	11.04%	8.65%	11.52%	0.54%	14.38%	2.19%
Phoenix	11.09%	5.81%	8.58%	7.82%	12.47%	10.94%
East	24.69%	26.95%	13.96%	6.89%	10.43%	13.14%
West	13.41%	8.23%	14.57%	10.31%	18.95%	7.91%
South	7.55%	6.73%	8.38%	4.90%	14.19%	5.97%
Market	16.41%	14.54%	12.86%	7.86%	14.93%	9.28%

Prepared by :admin The Electronics Club Page: 5
Actual Profit by Product 12/12/95

Product:VCR

	Jan	Feb	Mar	Apr	May	Jun
New York	34.65%	32.38%	19.50%	20.96%	14.14%	21.57%
Boston	31.44%	28.43%	22.33%	29.27%	25.86%	27.89%
Chicago	31.97%	29.00%	30.95%	27.85%	28.81%	28.26%
San Francisco	32.28%	31.47%	29.50%	31.81%	30.89%	26.42%
Seattle	31.06%	33.42%	29.99%	26.21%	26.86%	16.40%
Denver	31.18%	29.86%	30.39%	27.87%	29.66%	19.05%
Los Angeles	29.14%	29.11%	28.16%	26.27%	27.31%	29.36%
Dallas	32.17%	31.32%	27.11%	27.78%	27.55%	29.86%
Houston	34.07%	35.53%	35.29%	31.06%	32.90%	20.63%
Phoenix	32.91%	29.86%	30.39%	27.87%	29.66%	19.05%
East	32.87%	29.99%	24.43%	25.95%	22.99%	25.79%
West	30.88%	30.94%	29.42%	28.19%	28.76%	23.61%
South	32.99%	32.22%	30.83%	28.84%	29.89%	23.71%
Market	32.14%	30.99%	28.19%	27.62%	27.19%	24.37%

Prepared by :admin — The Electronics Club — Page: 6
Actual Profit by Product — 12/12/95

Product:Camera

	Jan	Feb	Mar	Apr	May	Jun
New York	16.35%	12.30%	6.72%	4.26%	4.66%	6.67%
Boston	10.55%	6.52%	10.66%	4.99%	17.87%	10.74%
Chicago	17.57%	12.68%	13.67%	12.95%	16.71%	13.10%
San Francisco	30.97%	25.36%	25.64%	24.43%	21.37%	25.96%
Seattle	16.69%	11.96%	12.33%	7.91%	20.53%	13.56%
Denver	22.36%	12.31%	12.19%	7.77%	20.53%	13.56%
Los Angeles	14.96%	13.19%	15.65%	11.15%	14.95%	11.30%
Dallas	11.68%	8.99%	11.41%	4.72%	17.62%	10.47%
Houston	12.07%	10.99%	9.83%	11.79%	21.17%	13.15%
Phoenix	14.36%	11.24%	10.13%	13.31%	22.56%	15.70%
East	15.06%	10.77%	10.07%	7.41%	12.67%	9.99%
West	22.31%	16.92%	17.63%	14.32%	19.47%	17.24%
South	12.71%	10.38%	10.47%	9.94%	20.45%	13.11%
Market	17.59%	13.28%	13.39%	11.04%	17.68%	13.92%

Prepared by :admin — The Electronics Club — Page: 7
Actual Profit by Product — 12/12/95

Product:Visual

	Jan	Feb	Mar	Apr	May	Jun
New York	24.40%	25.71%	9.65%	6.67%	5.23%	12.31%
Boston	28.51%	28.19%	20.58%	19.75%	19.82%	21.89%
Chicago	23.86%	18.63%	20.75%	16.17%	21.97%	17.24%
San Francisco	25.96%	22.72%	24.00%	22.11%	23.63%	19.59%
Seattle	19.41%	20.82%	19.77%	18.23%	23.54%	12.48%
Denver	23.89%	15.13%	20.45%	15.73%	23.82%	15.05%
Los Angeles	17.77%	15.97%	17.99%	13.38%	19.15%	15.46%
Dallas	15.67%	16.50%	15.32%	13.93%	20.36%	15.49%
Houston	20.01%	20.29%	20.62%	15.87%	23.60%	12.38%
Phoenix	20.01%	16.12%	17.18%	16.50%	21.39%	15.22%
East	25.49%	24.13%	16.77%	13.89%	15.46%	16.94%
West	21.85%	18.89%	20.65%	17.52%	22.43%	16.00%
South	18.39%	17.53%	17.59%	15.36%	21.66%	14.46%
Market	22.26%	20.27%	18.58%	15.76%	19.98%	15.91%

```
Prepared by :admin                The Electronics Club            Page: 8
                                 Actual Profit by Product        12/12/95

 Product:Product

                       Jan       Feb       Mar       Apr       May       Jun


  New York          22.71%    21.43%    13.11%    10.54%     9.73%    13.16%
  Boston            24.98%    23.25%    19.95%    18.00%    17.03%    18.62%
  Chicago           22.01%    17.94%    18.14%    15.45%    18.70%    16.01%
  San Francisco     23.71%    20.60%    21.93%    20.45%    21.44%    19.98%
  Seattle           21.06%    21.05%    21.24%    19.00%    21.72%    15.13%
  Denver            21.61%    16.01%    19.79%    14.81%    20.66%    13.89%
  Los Angeles       17.54%    15.51%    17.03%    14.33%    17.59%    16.09%
  Dallas            15.67%    16.50%    15.32%    13.93%    20.36%    15.49%
  Houston           20.01%    20.29%    20.62%    15.87%    23.60%    12.38%
  Phoenix           20.01%    16.12%    17.18%    16.50%    21.39%    15.22%
 East               23.19%    20.84%    16.89%    14.42%    14.94%    15.78%
 West               21.02%    18.35%    19.99%    17.26%    20.30%    16.61%
 South              18.39%    17.53%    17.59%    15.36%    21.66%    14.46%
Market              21.37%    19.09%    18.46%    15.92%    18.67%    15.93%
```

Use the following script to create Sample 9:

```
<PAGE (Accounts, Scenario, Product)
<IDESC Product
<SORTLEVEL
{ PAGEONDIMENSION Product }
{ STARTHEADING
TEXT    1 "Prepared by:"
       14 "*USERNAME"
        C "The Electronics Club"
       65 "*PAGESTRING"
SUPPAGEHEADING
UNDERLINECHAR " "
TEXT    C "Actual Profit by Product"
       65 "*DATE"
TEXT    1 "Product:"
       10 "*PAGEHDR 3"
SKIP
ENDHEADING }
Profit_%
```

```
Actual
      <COLUMN (Year)
      Jan Feb Mar Apr May Jun
<ROW(Market)

{ DECIMAL 2 AFTER "%" SUPBRACKET UNDERSCORECHAR " " }
{ INDENTGEN 1 }
<IDESC Market
     !
```

The SUPPAGEHEADING format command suppresses the default page headings from output.

The *PAGEHDR command customizes the location of page member labels. The Sample 9 script uses page heading number 3, Product, because this is the third page dimension.

You may have also noticed that member names do not have underscores. The UNDERSCORECHAR format command blanks out underscores.

Another difference is the underlining of column headings. The UNDERLINECHAR format command causes the underlining to character to change to the character in quotes.

The report rows are also sorted according to their levels in the database outline. Sort commands, such as SORTLEVEL, do not affect individual members selected in reports. Instead, these commands work in conjunction with member selection commands.

Note: You can use only one sort command in a report.

Sample 9 reverses the indentation of levels from previous reports. The INDENTGEN command indents members to the specified number of characters.

This report script, `HEADING2.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 10: Using Formulas

Column calculation formulas manipulate the column value of a particular row or a constant. In this report sample, each % column represents the quarterly values as a percent of Sales for the respective quarter. In addition, the Avg column represents an average value for the two quarters.

```
                              Actual Product Market

                        Qtr1      %     Qtr2     %       Avg
                    ======== ====== ======== ====== ========

Sales                141,245 100.00  136,193 100.00  138,719
Cost_of_Goods_Sold    58,104  41.14   56,281  41.32   57,193
  Margin              83,141  58.86   79,912  58.68   81,527
Marketing             11,211   7.94   11,302   8.30   11,257
Payroll               43,817  31.02   43,827  32.18   43,822
Miscellaneous            302   0.21    1,859   1.36    1,081
  Total_Expenses      55,330  39.17   56,988  41.84   56,159
    Profit            27,811  19.69   22,924  16.83   25,368
    Profit_%              20   0.01       17   0.01       18
    Margin_%              59   0.04       59   0.04       59
```

Use the following script to create Sample 10:

```
// This report performs column calculations based on values in a
// report row.

<PAGE (Scenario, Product, Market)
Actual

      <COLUMN (Year)
      Qtr1 Qtr2

{ DECIMAL 2 3 4 }
{ NAMEWIDTH 22 WIDTH 7 3 4 }
{ ORDER 0 1 3 2 4 5 }

<ROW (Accounts)
{ SAVEROW } Sales
     !

{ CALCULATE COLUMN "%" = 1 % "Sales" 1 }
{ CALCULATE COLUMN "% " = 2 % "Sales" 2 }
{ CALCULATE COLUMN "Avg" = 1 + 2 / 2. }

<DESC Accounts
     !
```

Note: You can include comments in the report by preceding the text with //. The Report Extractor ignores everything that follows the double slash. You can use comments to explain report processing.

The SAVEROW command reserves space for a row member that the CALCULATE COLUMN command calculates. In this case, the calculation affects Sales. The ! is required after the member name.

The CALCULATE COLUMN command allows column numbers, row names, or constants in formulas. You can read the first calculation this way: "% equals column 1 as a percent of Sales in column 1."

Each calculated column label must be unique. Note how the second calculated column label has a blank space after the % sign.

To specify a constant, define a number followed by a period. You can use a constant in either a column or row calculation. The last column calculation takes the sum of columns 1 and 2 and divides by the value 2. This formula is interpreted as (1+2)/2, *not* 1 + (2/2).

As noted in Sample 7, the ORDER command arranges columns in the specified order. By default, calculated columns are added to the end of existing columns retrieved from the database. In this example, columns 0-2 are automatically retrieved, based on selected members. Columns 3-5 are the calculated columns. The ORDER command applies to both retrieved and calculated columns.

This report script, `COLCALC1.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 11: Placing Two Page Layouts on the Same Page

This sample report has two different page layouts on the same page.

Year Profit_% Actual

	East	West	South	Market
Stereo	-0.52%	1.91%	0.00%	0.91%
Compact_Disc	32.60%	36.00%	0.00%	34.60%
Audio	17.86%	20.81%	0.00%	19.60%
Television	20.40%	16.57%	13.50%	17.21%
VCR	30.81%	32.43%	33.70%	32.24%
Camera	16.66%	21.66%	17.83%	19.07%
Visual	23.16%	23.56%	22.27%	23.09%
Product	21.34%	22.50%	22.27%	22.04%

Sales Actual Product

	Qtr1	Qtr2	Qtr3	Qtr4	Year
New_York	$18,631	$17,681	$19,923	$24,403	$80,638
Boston	$15,812	$15,050	$16,716	$19,159	$66,737
Chicago	$16,536	$15,599	$17,411	$21,374	$70,920
East	$50,979	$48,330	$54,050	$64,936	$218,295
San_Francisco	$19,761	$19,019	$20,722	$24,807	$84,309
Seattle	$13,766	$13,546	$14,204	$19,034	$60,550
Denver	$13,800	$13,588	$13,838	$18,232	$59,458
Los_Angeles	$17,866	$17,269	$17,208	$22,635	$74,978
West	$65,193	$63,422	$65,972	$84,708	$279,295
Dallas	$ 9,226	$ 9,175	$ 9,481	$12,700	$40,582
Houston	$ 7,690	$ 7,363	$ 7,646	$10,785	$33,484
Phoenix	$ 8,157	$ 7,903	$ 8,343	$11,843	$36,246
South	$25,073	$24,441	$25,470	$35,328	$110,312
Market	$141,245	$136,193	$145,492	$184,972	$607,902

Use the following script to create Sample 11:

```
<PAGE (Year, Accounts, Scenario)

      <COLUMN (Market)
      <ICHILD Market

<ROW(Product)
<IDESC Product

Actual
{ DECIMAL 2 WIDTH 10 SUPBRACKET AFTER "%" }
Profit_%
     !

<PAGE (Accounts, Scenario, Product)
Actual
Sales
Product

      <COLUMN(Year)
      <ICHILD Year

<ROW(Market)
{ DECIMAL 0 After " " BEFORE "$" }
<IDESC Market
     !
```

In a single report, you can select multiple dimension layouts and members. To define a multiple layout report, define reports as you normally would. Separate the commands with exclamation marks as shown above. Whenever the column, row, or page dimensions change between ! output commands, new headings are automatically generated to match the new layout.

The BEFORE format command places a character in front of data values. The AFTER format command turns off the percent signs from the first report layout.

This report script, `2LAYOUTS.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 12: Formatting for Data Export

This sample creates a report with a member name in each column. This format is required when you export Essbase data to another product, such as an SQL database, with a flat file.

```
New York        Stereo          Sales              1000.0    950.0
New York        Stereo          Cost of Goods Sold  580.0    551.0
New York        Stereo          Margin              420.0    399.0
New York        Stereo          Marketing            80.0     80.0
New York        Stereo          Payroll             340.0    340.0
New York        Stereo          Miscellaneous         0.0      0.0
New York        Stereo          Total Expenses      420.0    420.0
New York        Stereo          Profit                0.0    -21.0
New York        Stereo          Profit %              0.0     -2.2
New York        Stereo          Margin %             42.0     42.0
New York        Compact Disc    Sales              1200.0   1150.0
New York        Compact Disc    Cost of Goods Sold  456.0    437.0
New York        Compact Disc    Margin              744.0    713.0
New York        Compact Disc    Marketing            95.0     95.0
New York        Compact Disc    Payroll             310.0    310.0
New York        Compact Disc    Miscellaneous         0.0      0.0
New York        Compact Disc    Total Expenses      405.0    405.0
New York        Compact Disc    Profit              339.0    308.0
New York        Compact Disc    Profit %             28.3     26.8
New York        Compact Disc    Margin %             62.0     62.0
New York        Audio           Sales              2200.0   2100.0
New York        Audio           Cost of Goods Sold 1036.0    988.0
New York        Audio           Margin             1164.0   1112.0
New York        Audio           Marketing           175.0    175.0
New York        Audio           Payroll             650.0    650.0
New York        Audio           Miscellaneous         0.0      0.0
New York        Audio           Total Expenses      825.0    825.0
New York        Audio           Profit              339.0    287.0
New York        Audio           Profit %             15.4     13.7
New York        Audio           Margin %             52.9     53.0
New York        Television      Sales              1800.0   1600.0
```

Use the following script to create Sample 12:

```
<PAGE(Scenario)

<COLUMN(Year)

<ROW (Market, Product, Accounts)
<CHILD East
<DESC Product

{ DECIMAL 1
WIDTH 9
SUPBRACKET
SUPCOMMA
MISSINGTEXT " "
UNDERSCORECHAR " "
SUPHEADING
NOINDENTGEN
SUPFEED
ROWREPEAT

Budget
      Jan Feb

<DESC Accounts
     !
```

The ROWREPEAT command produces rows of data that have the member names repeat for each row dimension.

The SUPFEED command suppresses page feeds. A page feed automatically occurs when the report output reaches the default page length of 66 rows, unless you enter the PAGELENGTH command to change this setting. When a large flat file is created, you can use this command to prevent page breaks (blank rows) from appearing in the report every time output reaches a logical page length.

This report script, `FLAT2SQL.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 13: Creating Asymmetric Columns

Asymmetric columns make up this report. Typically, a report contains symmetric columns. That is, when multiple dimensions appear across the page as column groups, each level of nested columns has the same number of members nested below. Because Actual has only one nested column, Jan, and Budget has three nested columns, this report is considered asymmetric.

Some rows in the report use names other than the member names from the database. In addition to allowing alternate names, as in Sample 7, you can rename a row name in the reporter.

	Product Market			
	Actual Jan	Budget Jan	Budget Feb	Budget Mar
Revenue	49,896	49,950	45,770	45,770
Cost of Goods	20,827	19,755	18,058	18,047
Gross Margin	29,069	30,196	27,712	27,723
Marketing	3,560	3,515	3,525	3,515
Payroll	14,599	14,402	14,416	14,416
Miscellaneous	249	0	0	0
Total Expenses	18,408	17,917	17,941	17,931
Profit	10,661	12,279	9,771	9,792

Use the following script to create Sample 13:

```
<PAGE (Product, Market)

      <COLUMN (Scenario, Year)
      Actual   Budget Budget Budget
         Jan   Jan    Feb    Mar

<ROW (Accounts)

{ RENAME "Revenue" } Sales
{ RENAME "Cost of Goods" } Cost_of_Goods_Sold
{ RENAME "Gross Margin" } Margin

{ SKIP UNDERSCORECHAR " " }
<ICHILD Total_Expenses

{ SKIP }
Profit
!
```

To create an asymmetric report, you must specify the member name of each column. Because the report output has two column groupings, Scenario and Year, you must specify a member from each dimension for each column. If you do not specify each column member, the resulting report format is symmetric.

The RENAME command redefines a member name when the report is output. This is useful when you do not want to use an alternate names table.

This report script, `ASYMM.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 14: Calculating Columns

This section contains two examples of CALCULATE COLUMN scripts and the reports they produce. CALCULATE COLUMN supports standard mathematical operations.

Sample 14-A: Basic Calculated Columns

East

Actual						Budget				Var
Jan	Feb	Mar	Qtr1			Jan	Feb	Mar	Q1	Q1
1,295	1,132	553	2,980	Tele~	Profit	1,240	950	950	3,140	(160)
25	27	14	66		Profit_%	26	22	22	70	(4)
56	62	59	177		Margin_%	60	60	60	180	(3)
1,417	1,120	898	3,435	VCR	Profit	1,466	1,161	1,161	3,788	(353)
33	30	24	87		Profit_%	35	31	31	98	(10)
61	61	62	183		Margin_%	63	63	63	189	(6)
400	272	256	928	Cam~	Profit	528	360	360	1,247	(319)
15	11	10	36		Profit_%	19	13	13	45	(10)
70	70	70	211		Margin_%	71	71	71	213	(2)
3,112	2,524	1,707	7,343	Visu~	Profit	3,234	2,471	2,471	8,175	(832)
25	24	17	66		Profit_%	27	23	23	74	(7)
61	63	63	187		Margin_%	64	64	64	191	(4)

Use the following script to create Sample 14-A:

```
<PAGE(Market)
East Actual

      <COLUMN (Scenario, Year)
      Actual  Budget
      Jan Feb Mar

{ CALC COL "Qtr1"  = 2 : 4
  CALC COL "Q1" = 5 : 7
  CALC COL "Var~Q1" = 8 - 9

  ORDER 2,3,4,8,0,1,5,6,7,9
  WIDTH 7 WIDTH 10 0 1
}

<ROW (Product, Accounts)
<ICHILD Visual

<CHILD Accounts
     !
```

This report script, `COLCALC2.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 14-B: Asymmetric Columns

The following sample has two regular columns defined in *asymmetric* mode. For more information on asymmetric columns, see "Sample 13: Creating Asymmetric Columns" on page 36-38.

East

Budget Jan			Actual Jan	Actual % Sales
1,200	Television	Payroll	1,236	25%
440		Marketing	365	9%
1,240		Profit	1,295	26%
4,800		Sales	5,244	100%
1,030	VCR	Payroll	1,044	25%
150		Marketing	156	4%
1,466		Profit	1,417	35%
4,200		Sales	4,311	100%
1,195	Camera	Payroll	1,167	42%
300		Marketing	288	11%
528		Profit	400	19%
2,850		Sales	2,656	100%
3,425	Visual	Payroll	3,447	29%
890		Marketing	809	8%
3,234		Profit	3,112	27%
11,850		Sales	12,211	100%

Use the following script to create Sample 14-B:

```
<PAGE(Market)
East

     <COLUMN(Scenario, Year)
     Budget  Actual
     Jan     Jan

{ ORDER 2,0,1,3,4 WIDTH 12 0 1 NOINDENTGEN AFTER "%" 4
  SKIPONDIM Product LMARGIN 10 }

<ROW(Product, Accounts)

{ CALC ROW "Sales" OFF }
{ CALC COL "Actual~% Sales" = 2 % "Sales" 2 }

<ICHILD Visual

{ SAVEROW }Sales
     Payroll
     Marketing
     Profit
<DUPLICATE   Sales
     !
```

This report script, `COLCALC3.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 15: Calculating Rows

The sample reports in this section demonstrate CALCULATE ROW scripts and the reports they produce. The samples use the abbreviation CALC ROW instead of CALCULATE ROW.

Sample 15-A: Basic Calculated Row

This sample report demonstrates the basic form of the CALCULATE ROW command.

```
                    Audio Actual Sales

                      Jan      Feb      Mar
                 ======== ======== ========

Boston              1,985    1,801    1,954
New_York            2,310    2,082    2,259
Chicago             2,043    1,884    1,814

Total Sales         6,338    5,767    6,027
Avg Sales           2,113    1,922    2,009
```

Use the following script to create Sample 15-A:

```
     Audio Actual Sales
     Jan Feb Mar

{ CALC ROW "Total Sales" }      //create new calculated row

Boston
New_York
Chicago

{ SKIP
  CALC ROW "Avg Sales" = "Total Sales" /3
  PRINTROW "Total Sales"
  PRINTROW "Avg Sales" }
      !
```

This report script, `ROWCALC1.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 15-B: Calculated Rows and Missing Relationships

This sample report is a simple summary of information in a North/South grouping, which is not part of the database outline. When relationships that you need for reporting are missing in the database outline, often the best solution is to use calculated rows (or columns).

```
       Budget Payroll

                        Jan       Feb       Mar
                       ====      ====      ====

Northern Cities
================
New_York              1,940     1,930     1,930
Boston                1,610     1,610     1,610
Chicago               1,630     1,630     1,630
San_Francisco         1,815     1,815     1,815
Seattle               1,415     1,409     1,409

Southern Cities
================
Denver                1,499     1,499     1,499
Los_Angeles           1,757     1,787     1,787
Dallas                1,002     1,002     1,002
Phoenix                 900       900       900
Houston                 834       834       834

Total Northern        8,410     8,394     8,394
Total Southern        5,992     6,022     6,022
```

Use the following script to create Sample 15-B:

```
// Declare Calculated Rows to Sum Southern and Northern Cities
{ CALC ROW "Total Southern" OFF

// initially, set operation to OFF
  CALC ROW "Total Northern" OFF  }

<PAGE(Product,Scenario,Accounts)
{ RENAME "" } Product            // all products, so blank out
                                 // the Product Label
Budget
Payroll

     <COLUMN(Year)
     Jan  Feb  Mar

<ROW(Market)                     // Northern Cities

{ SETROWOP "Total Northern" +    // Accumulate for Northern

SKIP 3
IMMHEADING                       // Put out heading now so text
                                 // will go after it
Text 0 "Northern Cities" UCHAR
}

New_York Boston  Chicago San_Francisco Seattle

                                 // Southern Cities

{ SETROWOP "Total Southern" +   } // Accumulate for Southern
{ SETROWOP "Total Northern" OFF } // Stop Accumulation for Northern

{ SKIP Text 0 "Southern Cities" UCHAR }

Denver  Los_Angeles Dallas  Phoenix Houston

{ SKIP
PRINTROW "Total Northern"        // output calculated rows
PRINTROW "Total Southern"
}
     !
```

This report script, `ROWCALC2.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 15-C: Averaging Rows

This report sample restricts columns during calculation to average rows that contain partly numbers and percentages. The report must calculate the total regional average percentages using previously calculated rows that contain the total sales for the region. Also, the report must compute (for averaging) a count of regions. The number of regions is set as a constant in the database outline. If this number changes, the report definition must be modified. If a count of regions is not computed, a hard-to-notice error could result.

```
 Actual Total Sales for the 3 Video Products in Qtr1:  36,914   35,126   25,119
 Budget Total Sales for the 3 Video Products in Qtr1:  37,300   34,250   26,940
 ================================================   ======   ======   ======

                 Qtr1

Television          VCR   Camera
                     Profit Profit_%       Profit Profit_%  Profit   Profit_%
                     ======  =======      ====== ========  ======  =========

 New_York   Budget    1,020   20.40%        1,382   31.41%     540     16.68%
            Actual      847   17.66%        1,243   29.62%     352     11.79%
 Boston     Budget    1,020   24.88%        1,344   35.37%     277     11.79%
            Actual    1,405   33.48%        1,002   27.49%     207      9.28%
 Chicago    Budget    1,100   25.58%        1,062   31.24%     430     16.54%
            Actual      728   16.51%        1,190   30.68%     369     14.72%
 San_Fran~  Budget      930   21.63%          718   21.12%   1,270      31.75
            Actual      674   15.54%        1,197   31.12%   1,000      27.4%
 Seattle    Budget      390   15.60%          973   32.98%     376     16.00%
            Actual      340   12.20%          977   31.56%     312     13.79%
 Denver     Budget      690   22.26%          929   30.97%     462     18.86%
            Actual      334   11.94%          914   30.48%     361     15.92%
Los_Ange~   Budget      810   18.41%        1,101   29.76%     506     18.40%
            Actual      429    9.11%        1,127   28.81%     377     14.62%
 Dallas     Budget      780   21.08%        1,341   36.24%     333     13.88%
            Actual      163    4.69%        1,055   30.28%     243     10.71%
 Houston    Budget      690   24.64%        1,128   36.39%     432     18.00%
            Actual      256   10.44%        1,064   34.98%     241     10.98%
 Phoenix    Budget      630   20.32%          894   31.93%     498     20.75%
            Actual      251    8.49%          940   31.07%     261     11.99%

                                    Total Regions Averages

 Avg    Budget    806    21.61%    1,087    31.74%      512     19.02%
 Avg     Actual   543    14.70%    1,071    30.49%      372     14.82%
```

Use the following script to create Sample 15-C:

```
{ // Declare some of the Calculated Rows to be used
  CALC ROW "Avg~Budget" OFF
  CALC ROW "Avg~Actual" OFF
  CALC ROW "Tot Sales~Budget" OFF
  CALC ROW "Tot Sales~Actual" OFF
}

// We need the values of Market->Visual->Qtr1->Sales->Actual and
// Market ->Visual->Qtr1->Sales ->Budget to compute some
// percentages at the bottom, so get them now

Market
<CHILD Visual Qtr1 Sales
{ SAVEROW "Actual Sales" } Actual// stores into first 3
                           // data columns
{ SAVEROW "Budget Sales" } Budget// of these rows, which
                           // are cols 1-3
                           // change to columns 2-4 when we
                           // specify 2 row dimensions in
                           // next section

// since this is an example, not a formal report, we'll
// type out the values for Actual Sales and Budget Sales here so
// you can check the numbers:

{ SKIP 2
TEXT 0 "Actual Total Sales for the 3 Video Products in Qtr1:" 55
"*CALC" "Actual Sales"
TEXT 0 "Budget Total Sales for the 3 Video Products in Qtr1:" 55
"*CALC" "Budget Sales"
UCHARACTERS
SKIP 5 }
     !                     // Now we can do the main report

{ AFTER "%" 3,5,7  DECI 2 3,5,7 ZEROTEXT "--" MISSING "--"
  WIDTH 10 0 1  }
```

```
<PAGE(Year)
Qtr1

                <COLUMN(Product,Accounts)
                <CHILD Visual
                Profit     // split these 2 accounts onto
                           // 2 lines to prevent default
                Profit_%   // to asymmetric mode
                           // because both column
                           // dimensions have the same # of
                           // members selected. Could have
                           // used <SYM instead.

<ROW(Market,Scenario)
<ONSAMELEV New_York
            { SETROWOP "Avg~Actual" OFF
              SETROWOP "Avg~Budget" +

              CALC ROW "Count" = "Count" + 1.    }

            Budget

            { SETROWOP "Avg~Budget" OFF
              SETROWOP "Avg~Actual" +             }

           >{ SKIP }

            Actual

{ UCOLUMNS SKIP 2 }
{
  // at this point, Avg~Budget and Avg~Actual ARE NOT YET
  // AVERAGES--they are the SUM of the Profit rows of each type.
  // Before converting them to averages, the report computes
  // Profit as a % of total sales for each type.  Since we only
  // have 1 value for "Budget Sales" and "Actual Sales",
  // for each of the three visual products in those
  // rows, the report restricts the reference to those rows to
  // columns 2-4 while computing
  // the percentage columns 3, 5, and 7, based on profits in
  // columns 2, 4 and 6

  // calculate the percentages for Budget

CALC ROW "Avg~Budget" 3 = "Avg~Budget" 2 % "Budget Sales" 2
CALC ROW "Avg~Budget" 5 = "Avg~Budget" 4 % "Budget Sales" 3
CALC ROW "Avg~Budget" 7 = "Avg~Budget" 6 % "Budget Sales" 4

  // now calculate the averages
```

```
CALC ROW "Avg~Budget" 2  = "Avg~Budget" / "Count"
CALC ROW "Avg~Budget" 4  = "Avg~Budget" / "Count"
CALC ROW "Avg~Budget" 6  = "Avg~Budget" / "Count"

  // calculate the percentages for Actual

CALC ROW "Avg~Actual" 3 = "Avg~Actual" 2 % "Actual Sales" 2
CALC ROW "Avg~Actual" 5 = "Avg~Actual" 4 % "Actual Sales" 3
CALC ROW "Avg~Actual" 7 = "Avg~Actual" 6 % "Actual Sales" 4

  // now calculate the averages

CALC ROW "Avg~Actual" 2  = "Avg~Actual" / "Count"
CALC ROW "Avg~Actual" 4  = "Avg~Actual" / "Count"
CALC ROW "Avg~Actual" 6  = "Avg~Actual" / "Count"

TEXT C "Total Regions Averages"
PRINTROW "Avg~Budget"
PRINTROW "Avg~Actual" }
     !
```

This report script, `ROWAVG.REP`, is available in your `\ESSBASE\APP\DEMO\BASIC` directory.

Sample 16: Sorting by Top or Bottom Data Values

The following two reports demonstrate the use of TOP and BOTTOM conditional retrieval commands in a report script. For more information, see Chapter 35, Developing Report Scripts.

Sample 16-A: Bottom Data Values

This sample report demonstrates the basic use of the BOTTOM command. The report is based on the Sample Basic database.

		Measures			
		Actual		Budget	
		Jan	Dec	Jan	Dec
East	200	158	233	280	340
	300	184	277	240	210
	Diet	181	213	200	240
West	100	378	223	830	530
	300	755	971	830	950
	400	454	434	470	370
South	200	480	496	520	390
	Diet	355	404	490	430
	300	188	213	270	240
Central	300	790	824	930	810
	100	724	792	900	890
	400	691	785	660	650
Market	200	2,141	2,302	2,710	2,810
	300	1,917	2,285	2,270	2,210
	400	1,611	1,720	1,730	1,600

Use the following script to create Sample 16-A:

```
<Sym
<Column (Scenario, Year)
Actual Budget
Jan Dec
<Row (Market, Product)
<Ichild Market
<Ichild Product
<Bottom (3, @DataColumn(3))
      !
```

The BOTTOM command specifies that only the three lowest data values are returned for each row grouping, based on the target data values specified in column three (Budget, Jan). Notice that no row dimension is selected here, so the report output defaults to the innermost row.

This report script, `BOTTOM.REP`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory.

Sample 16-B: Top Data Values

This sample report demonstrates the basic use of the TOP command. The report is based on the Sample Basic database.

		Measures			
		Actual		Budget	
		Jan	Dec	Jan	Dec
East	Product	1,732	2,037	2,080	2,120
	100	924	1,026	960	990
	400	466	501	600	580
West	Product	2,339	2,448	2,980	2,710
	200	752	820	850	860
	Diet	663	629	850	730
South	Product	997	1,141	1,330	1,270
	100	329	432	540	640
	200	480	496	520	390
Central	Product	2,956	3,154	3,550	3,570
	Diet	1,080	1,064	1,340	1,300
	200	751	753	1,060	1,220
Market	Product	8,024	8,780	9,940	9,670
	100	2,355	2,473	3,230	3,050
	Diet	2,279	2,310	2,880	2,700

Use the following script to create Sample 16-B:

```
<Sym
<Column (Scenario, Year)
Actual Budget
Jan Dec
<Row (Market, Product)
<Ichild Market
<Ichild Product
<Top (10, @DataColumn(3))
      !
```

The TOP command specifies that only the three highest data values are returned for each row grouping, based on the target data values specified in column three (Budget, Jan). Notice that no row dimension is selected here, so the report output defaults to the innermost row.

This report script, `TOP.REP`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory.

Sample 17: Restricting Rows

The following report demonstrates the use of the RESTRICT conditional retrieval command in a report script. For more information, see Chapter 35, Developing Report Scripts.

```
                                             Measures

                                     Actual            Budget
                                    Jan       Dec       Jan       Dec
                               ======== ======== ======== ========

East              200                158       233       280       340
                  300                184       277       240       210
                  Diet               181       213       200       240
South             300                188       213       270       240
                  400           #Missing #Missing #Missing #Missing
```

Use the following script to create Sample 17:

```
<Sym
<Column (Scenario, Year)
Actual Budget
Jan Dec
<Row (Market, Product)
<Ichild Market
<Ichild Product
<RESTRICT (@DataCol(3) < $300.00 )
     !
```

The RESTRICT command specifies that only data values that are less than $300.00 are returned for each row grouping, based on the target data values specified in column three (Budget, Jan). Notice that no row dimension is selected here, so the report output defaults to the innermost row.

This report script, `RESTRICT.REP`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory.

Sample 18: Ordering Data Values

The following report demonstrates the use of the ORDERBY conditional retrieval command in a report script. For more information, see Chapter 35, Developing Report Scripts.

		Sales Scenario			
		Jan	Feb	Mar	Apr
New York	100-20	#Missing	#Missing	#Missing	#Missing
	100-30	#Missing	#Missing	#Missing	#Missing
	200-20	#Missing	#Missing	#Missing	#Missing
	200-30	#Missing	#Missing	#Missing	#Missing
	300-30	#Missing	#Missing	#Missing	#Missing
	Diet	#Missing	#Missing	#Missing	#Missing
	200-10	61	61	63	66
	400-30	134	189	198	198
	300-20	180	180	182	189
	400-20	219	243	213	223
	400-10	234	232	234	245
	300-10	483	495	513	638
	200-40	490	580	523	564
	200	551	641	586	630
	400	587	664	645	666
	300	663	675	695	827
	100-10	678	645	675	712
	100	678	645	675	712
	Product	2,479	2,625	2,601	2,835

Use the following script to create Sample 18:

```
<PAGE ("Measures")
<COLUMN ("Scenario", "Year")
<ROW ("Market", "Product")
"Sales"
"Scenario"
"Jan" "Feb" "Mar" "Apr"
"New York"

"Product" "100" "100-10" "100-20" "100-30" "200" "200-10" "200-20"
"200-30" "200-40" "300" "300-10" "300-20" "300-30" "400" "400-10"
"400-20" "400-30" "Diet" "100-20" "200-20" "300-30"

<ORDERBY ("Product", @DATACOL(1) ASC, @DATACOL(2) DESC, @DATACOL(3)
ASC @DataCol (4) DESC)
     !
```

The ORDERBY command is based only on data in the data columns. If the SUPPRESSMISSING command is not used in the report, `#MISSING` is considered to be the lowest data value. ORDERBY compares data values in the following order:

- Two values in the same column (for example, in COL1, the value associated with 200-10 is compared with the 400-30 data value, as shown in the example below).
- Data values between two data columns (for example, the data value in COL1 is compared with the data value in COL2, as shown in the example below).

If two data value are the same, the sort proceeds to the next column to determine the order.

In this example, the data value "61" in COL1 and COL2 are identical; the data in COL1 should be in ascending order, the data in COL2 should be in descending order. The two values are compared, and as they are the same, COL2 and COL3 are compared. Therefore, even though COL2 is supposed to be in descending order, the comparison for the row 400-30 was determined by the values in COL3, which is in ascending order.

	COL 1	COL 2	COL 3	COL 4
	=====	=====		
200-10	61	61	63	66
400-30	134	189	198	198
300-20	180	180	182	189

This report script, `ORDERBY.REP`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory.

Sample 19: Narrowing Member Selection Criteria

The following report demonstrates the use of the LINK command to narrow the members returned in a selection in a report script. For more information, see Chapter 35, Developing Report Scripts.

```
              Market Measures Scenario

                     Qtr1     Qtr2
                 ======== ========

    Product        24,703   27,107
100-10              5,096    5,892
100-20              1,359    1,534
100-30                593      446
200-10              1,697    1,734
200-20              2,963    3,079
200-30              1,153    1,231
200-40                908      986
300-10              2,544    3,231
300-20                690      815
300-30              2,695    2,723
400-10              2,838    2,998
400-20              2,283    2,522
400-30               (116)     (84)
100-20              1,359    1,534
200-20              2,963    3,079
300-30              2,695    2,723
```

Use the following script to create Sample 19:

```
<PAGE (Market)
<COLUMN (Year)
Qtr1 Qtr2
<ROW (Product)
<LINK (<UDA (product, naturally-flavored) OR <LEV (product, 0))
     !
```

The LINK command uses the AND, OR, and NOT Boolean operators to refine your search. In the preceding example, the product with the "naturally-flavored" user-defined attribute, as well as all Level 0 products, are returned in the search.

Be careful how you group operators in the LINK expression. Essbase evaluates operators from left to right. Use parentheses to group the expressions. For example, A OR B AND C is the same as ((A OR B) AND C). In the first expression, Essbase evaluates the expression from left to right, evaluating A OR B before evaluating AND C. In the second expression, Essbase evaluates the subexpression in parentheses (A OR B) before the whole expression, producing the same result. However, if you use (A OR (B AND C)), Essbase evaluates the subexpression in parentheses (B AND C) before the whole expression, producing a different result.

This report script, `LINK.REP`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory.

Chapter 37 Optimizing Your Reports

The time required to generate a report varies depending upon the size of the database you're reporting from, the number of report queries included in the script, the size of the report buffer, and other factors.

This chapter describes ways to optimize the time required to generate your reports. It discusses the following topics:

If you are migrating from a previous version of Hyperion Essbase, see the Essbase *Start Here* booklet for important information about system changes and enhancements.

Setting Configurable Variables

Configurable variables specify the size of the internal server buffers used for storing and sorting data extracted by a report. The following report variables are used in the conditional retrieval and data sorting commands:

- Retrieval Buffer Size setting
- Retrieval Sort Buffer Size setting
- NUMERICPRECISION setting

For more information about these settings, see the online *Technical Reference* in your `DOCS` directory. For information about the Report Extractor's process of retrieving data, see Chapter 34, Quick Start to Report Scripts.

Setting the Retrieval Buffer Size

The Retrieval Buffer Size setting specifies the size, in kilobytes, of the server buffer that holds extracted row data cells before they are evaluated by the RESTRICT, TOP, or BOTTOM commands. If this buffer is full, the rows are processed, and the buffer is reused. Both the Spreadsheet Retrieval Wizard and the Report Writer use this buffer to process retrievals.

You can adjust the buffer size on a per database basis. The default buffer size is set to 10 kilobytes. If you are increasing the size of the buffer, it is recommended that you do not exceed 100 kilobytes, although the size limit is set at 100,000 kilobytes. Larger buffer sizes can create problems when concurrent users generate reports at the same time.

To set the retrieval buffer size:

1. From the Application Manager menu, choose Database | Settings.
2. Select the **General** page.
3. Type the number, in kilobytes, of the required buffer size in the **Retrieval Buffer Size** box.

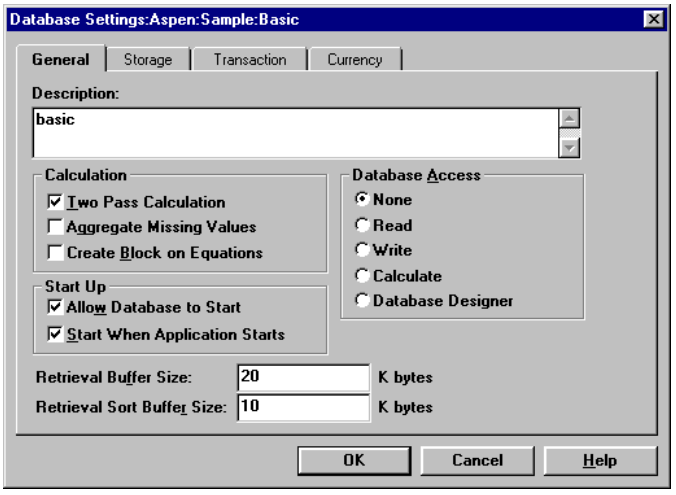

Figure 37-1, Setting the Retrieval Buffer Size

Notes:

- To determine the best buffer size for your report processing needs, test a report script with different settings.
- In Essbase Version 4 this variable was called REPTKBYTE and was set in the `ESSBASE.CFG` file. You can no longer set the variable in the `ESSBASE.CFG` file.

You can use the SETDBSTATEITEM command in ESSCMD to set the retrieval buffer size. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Setting the Retrieval Sort Buffer Size

The Retrieval Sort Buffer Size setting specifies the size, in kilobytes, of the server buffer that holds the data to be sorted during a Hyperion Essbase Spreadsheet Add-in or Application Manager Report Writer retrieval. If the sorting buffer is full, Essbase posts an error message.

You can adjust the buffer size on a per-database basis. The default buffer size is set to 10 kilobytes.

To set the retrieval sort buffer size:

1. From the Application Manager menu, choose Database | Settings.
2. Select the **General** page.
3. Type the number, in kilobytes, of the required sort buffer size in the **Retrieval Sort Buffer Size** box.

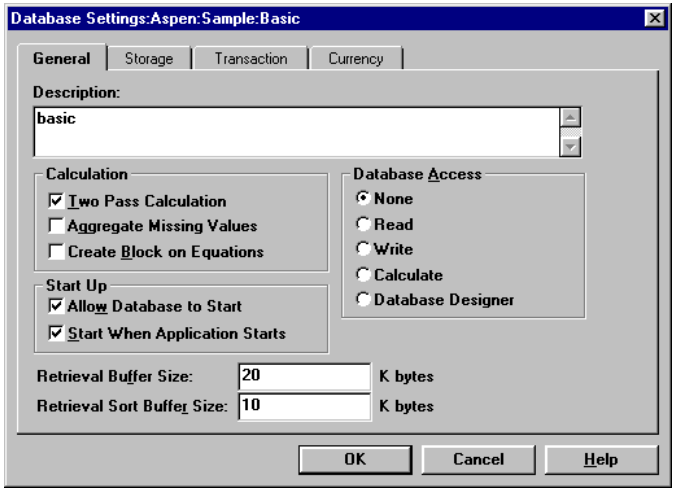

Figure 37-2, Setting the Retrieval Sort Buffer Size

Notes:

- To determine the best buffer size for your report processing needs, test a report script with different settings.
- In Essbase Version 4, this variable was called REPTKBYTESORTBUF and was set in the `ESSBASE.CFG` file.

You can use the SETDBSTATEITEM command in ESSCMD to set the retrieval sort buffer size. See the online *Technical Reference* in your `DOCS` directory for information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Setting the NumericPrecision Variable

The NUMERICPRECISION setting, used by the RESTRICT command, defines the number of precision digits the internal numerical comparison considers in the Report Extractor.

To set the NUMERICPRECISION variable:

1. Open the `ESSBASE.CFG` server file.

Note: The `ESSBASE.CFG` file is created manually. Consult your Database Administrator if the file is not available.

2. Find the following line in the file:

   `NUMERICPRECISION` *`X`*

   where *X* is a numeric value representing the number of precision digits that are used by the Report Writer for numerical comparison.

Note: If the line does not exist, type in the variable information using the example above. See the online *Technical Reference* in your `DOCS` directory for a syntax description and details about the NUMERICPRECISION setting, as well as more information about adding information to the `ESSBASE.CFG` file.

Generating Symmetric Reports

If report processing time is of primary importance, consider making all reports symmetric. Symmetric reports provide better processing performance than asymmetric reports, as the Report Extractor composes the member list based on all possible member combinations, which it can do in one pass. With asymmetric reports, the Extractor must retrieve and process each block of possible member combinations separately.

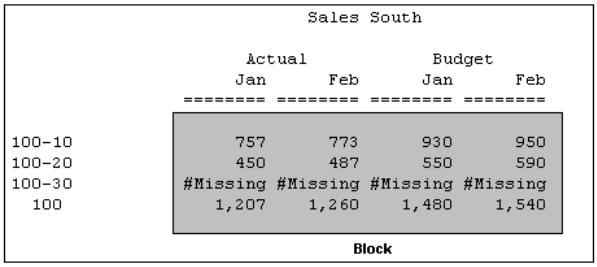

Figure 37-3, Symmetric Report Member Combinations

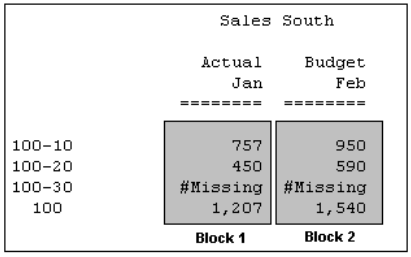

Figure 37-4, Asymmetric Report Member Combinations

For more information about how the Report Extractor retrieves data, see Chapter 34, Quick Start to Report Scripts.

Organizing Members to Optimize Data Extraction

The Report Extractor extracts data in a certain order. If you do not require a formatted report, you can reduce the time required to generate the report by:

- Structuring the report script in the same order as the Report Extractor extracts data
- Grouping dense dimensions in columns
- Grouping sparse dimensions in rows

This is particularly useful when creating large production reports.

The Report Extractor looks at data from bottom to top and right to left, starting from the bottom column member to the top column member and then proceeding from the innermost row member (right) to the outermost row member (left). In Figure 37-5, the report is read in numerical sequence.

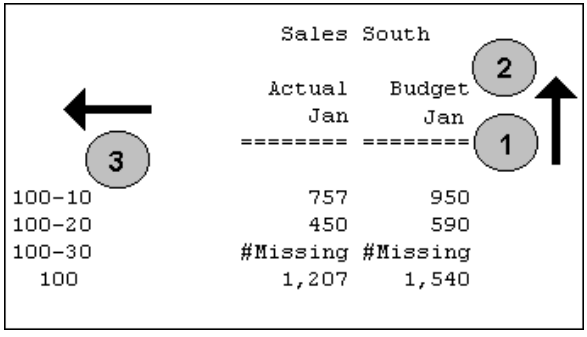

Figure 37-5, How the Report Extractor Examines Data

In Figure 37-5, the column members come from dense dimensions, and the row members come from sparse dimensions. To optimize the time needed to extract data, group dense dimensions first, then group sparse dimensions in the same sequence as they appear in your outline.

By nesting dense dimensions in the report columns, the Report Extractor only examines each block once, thus improving the performance time. Sparse dimensions follow the dense dimensions.

Working with Database Outlines

Report scripts extract data based on members identified in the database outline. This section discusses time implications when generating reports.

Accessing Outlines that Contain Dynamic or Transparent Members

If you generate a report that accesses a database outline containing Dynamic Calc And Store members, the first time that you generate the report takes longer than subsequent retrievals that access the same data block.

If you generate a report that accesses a database outline containing Dynamic Calc or Dynamic Time Series members, Essbase calculates the outline member every time a report is generated, which increases the reporting time.

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

If you run a report that contains transparent members, the report takes longer to generate, as it must access more than one server to retrieve the required data.

Ensuring that Members in the Report Script Synchronize with the Database Outline

When you run a report, it is important to ensure that the member names referenced in the report script match the members in the database outline. An error appears every time the Report Extractor cannot find a matching member name, and you must correct the name in the report script before the report continues processing.

Chapter 38 Copying Data Subsets and Exporting Data to Other Programs

You can move data between Hyperion Essbase OLAP Server databases or to another program by extracting an output file of the data you want to move. For example, you can copy a subset of an Essbase database from the Essbase server to Personal Essbase.

In order to meet the import format specifications of most other programs, use the Report Writer to create a text file.

This chapter contains information about the following topics:

Copying a Database Subset to Personal Essbase

This version of Essbase includes the ability to install both the Essbase server and client on a Windows NT workstation or a Windows 95 or Windows 98 machine. This is a one-port license and has its own license number. For information about installing and configuring Personal Essbase on a machine, see the *Installation Notes*.

Once you have installed Personal Essbase, you can copy the outline file (*dbname*.OTL) and a data subset from the Essbase server and load them into Personal Essbase. The Personal Essbase server does not communicate with the Essbase server.

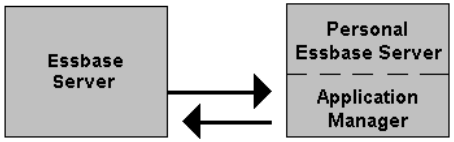

Figure 38-1, Essbase Server and Personal Essbase Interaction

Note: Do not create more than one application and two databases on the Personal Essbase server.

Summary of Steps

To copy a database subset to Personal Essbase, complete these steps. The following sections provide detailed information on completing each step.

1. On your Personal Essbase server, create a new application and database to contain the database subset.
2. Copy the outline file (for example, source_dbname.OTL) from your source database to the new database on Personal Essbase.

   You may need to rename the outline file to match the name of your Personal Essbase database (for example, target_dbname.OTL), overwriting the existing target database outline file.
3. Create an output file (for example, an ASCII text file) containing the required data subset.
4. Load the output file into the new database that you have created.

   See Chapter 19, Introducing Data Loading, for more information about loading data.

If required, you can repeat steps 3 and 4 to create an output file from the database on your Personal Essbase server and load the data back into the main Essbase database on a different machine.

The example in the following sections is based on the Sample Basic database. The data subset in the example is the Actual, Measures data for the West market. The example copies the data subset to a Personal Essbase server.

Creating a New Application and Database

Create a new application and database on your Personal Essbase server. You will copy the required subset of data into this new database. You can give this application and database any name. In this example, you copy a subset of data for the West market, and name the application `WEST.APP`, and the database `WESTMKTS.DB`.

To create the application and database:

1. In Application Manager, choose File | New | Application to open the **Create New Application** dialog box. Type the new application name in the **Application name** text box. For example, type `West`.
2. Click OK.
3. In Application Manager, choose File | New | Database to open the **Create New Database** dialog box. Type the new database name in the **Database name** text box. For example, type `Westmkts`.
4. Click OK.

   Essbase creates the new application and database.

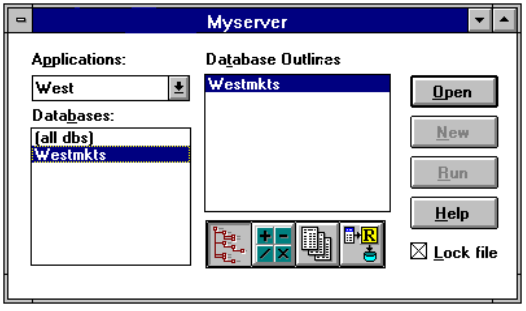

Figure 38-2, New West Application and Westmkts Database

Ensure that the new, empty database is not running. In Application Manager, select the new database and choose Application | Start/Stop from the main menu. A message box tells you if the database is running. If necessary, click **Yes** in the **Stop database** message box to stop the database.

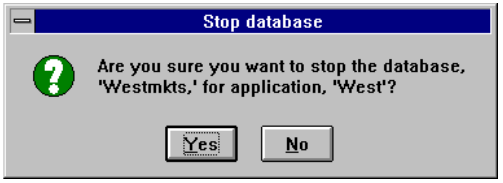

Figure 38-3, Stop Database Dialog Box

Copying the Outline File from Your Source Database

Copy the outline file (`.OTL`) of your source database to the new database that you have created. In this example, you copy the `BASIC.OTL` outline file from the Sample Basic database and rename it to `WESMKTS.OTL` on Personal Essbase.

How you copy the outline file depends on whether you can connect to the source Essbase database from your Personal Essbase machine.

- If you *can* connect, use File | Save As in Application Manager to copy the outline. See "Copying the Outline File Using Application Manager" on page 38-5.
- If you *cannot* connect (for example, if your Personal Essbase machine is a laptop machine that has no network connection), use your operating system to copy the outline file. For more information, see "Copying the Outline File Using the Operating System" on page 38-6.

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

Copying the Outline File Using Application Manager

Complete these steps if you *can* connect to the source Essbase database from your Personal Essbase machine.

1. Connect to the source server.
2. In Application Manager, open the source outline. For example, open Sample Basic.

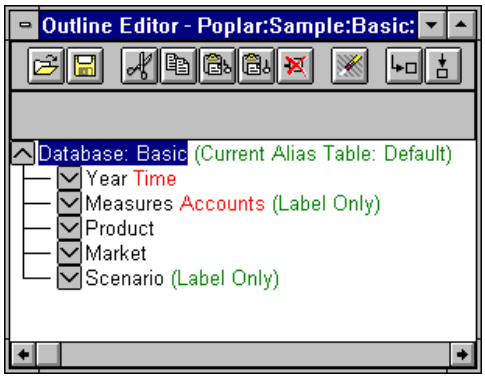

Figure 38-4, Sample Basic Outline

3. Choose File | Save As from the Outline Editor menu. The **Save Server Object** dialog box appears.

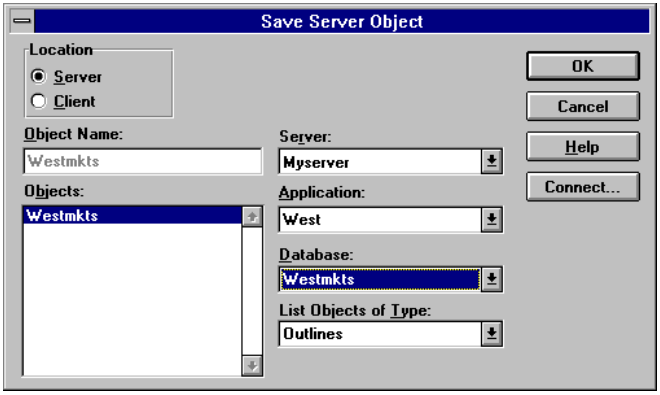

Figure 38-5, Save Server Object Dialog Box

4. In the **Server**, **Application**, and **Database** list boxes, select the new application and database that you have just created on the Personal Essbase server.
5. Click OK in the **Save Server Object** dialog box to copy the outline file. At the prompt, click **Yes** to replace the existing `WESTMKTS.OTL` file. The existing outline file is an empty file, which Essbase created automatically when you created the new application and database.
   - Essbase might prompt you to unlock and overwrite the file. If necessary, click **Yes** at the prompts.
   - Essbase might warn you that the outline file did not originate from the latest version on the server. If necessary, click **Yes** to continue.

Copying the Outline File Using the Operating System

Complete these steps if you *cannot* connect to the source Essbase database from your Personal Essbase machine.

1. Use the operating system to copy the source outline file. For example, copy `BASIC.OTL` to `WESTMKTS.OTL`. Give the copied outline exactly the same name as the new database.
2. Save the outline file in the `\`*`ARBORPATH`*`\APP\`*`appname`*`\`*`dbname`* directory on your Personal Essbase machine, where *ARBORPATH* is the directory in which your installed Essbase, and *appname* and *dbname* are the new application and database that you have created.

   For example, copy `BASIC.OTL` to a disk, renaming it to `WESTMKTS.OTL`. Then copy `WESTMKTS.OTL` from the disk to `C:\ESSBASE\APP\WEST\WESTMKTS\WESTMKTS.OTL` on your Personal Essbase machine. It is safe to overwrite the existing, empty `WESTMKTS.OTL` file.

Note: Ensure that the new outline file overwrites the existing, empty outline file, which Essbase created automatically when you created the new application and database.

3. Before using the new outline file, you need to stop and restart the new database. Select the new database in Application Manager. For example, select `Westmkts`.
4. Choose Application | Start/Stop from the main menu to stop the new database, and choose Application | Start/Stop again to restart the database.

You now have a copy of the database outline on your Personal Essbase server.

Creating an Output File Containing the Required Data Subset

Create an output file that contains the required data subset. The output file can be an ASCII text file, or a spreadsheet file. You can use the Report Writer to create an ASCII text file of the data subset. For example, use <IDESC to select a data subset.

Note: You can also use the Spreadsheet Client Retrieval Wizard to create a spreadsheet file of the data subset. For more information on using the Retrieval Wizard, see the *Spreadsheet Add-in User's Guides*.

To create an ASCII text file that contains the required data subset:

1. In Application Manager, select the source database. For example, select West Westmkts.

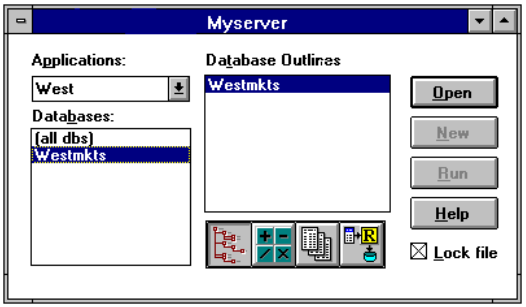

Figure 38-6, Westmkts Database Selected in Application Manager

   - If you *can* connect to your main Essbase database from your Personal Essbase machine, you can select the source database from your Personal Essbase machine.
   - If you *cannot* connect, use a different machine from your Personal Essbase machine to select the source database.

2. Click the Report icon,  , and then click **New** to open the Report Editor.

3. Write a report script that selects the required data subset. For information on writing report scripts, see Chapter 34, Quick Start to Report Scripts, and the online *Technical Reference* in your DOCS directory.

   For example, the following report script selects the Actual, Measures data for the West market from Sample Basic:

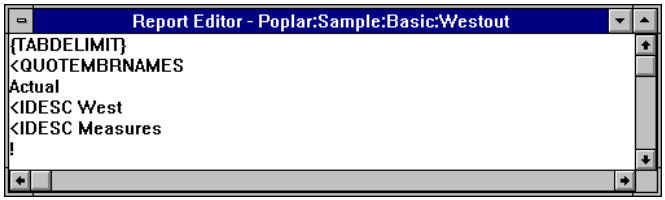

Figure 38-7, Report Editor With Sample Basic Report Script

   - Use TABDELIMIT to place tab stops between data, instead of spaces. This ensures that no member names or data values are truncated.
   - Use QUOTEMBRNAMES to place quotation marks ("") around member names that contain blank spaces. Essbase then recognizes the member names when it loads the data.

4. In the Report Editor, choose Report | Output Options to open the **Report Output Options** dialog box. Check the **File** and **Window** check boxes.

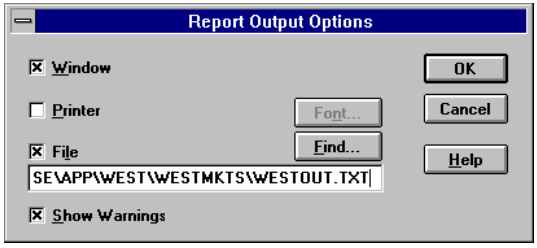

Figure 38-8, Report Output Options Dialog Box

5. Give the report output file any name with a `.TXT` extension. For example, `WESTOUT.TXT`.

   To load the data, the output file needs to be in the `\ARBORPATH\APP\appname\dbname` directory on your Personal Essbase server, where *ARBORPATH* is the directory in which you installed Essbase, and *appname* and *dbname* are the new application and database directories that you have created.

   If you are using your Personal Essbase machine, you can save the output file directly into the `\ARBORPATH\APP\appname\dbname` directory. For example, type `C:\ESSBASE\APP\WEST\WESTMKTS\WESTOUT.TXT` in the **File** text box.

   If you are *not* using your Personal Essbase machine, save the output file anywhere on the current machine. By default, Essbase saves the file on your Essbase client machine, and not on the Essbase server. When you run the report, use your operating system to copy the file to the `\ARBORPATH\APP\appname\dbname` directory on your Personal Essbase server. For example, use a disk to copy the file.

6. Click OK in the **Report Output Options** dialog box.

7. Save your report script and choose Report | Run from the Report Editor menu to run the report.

   Essbase displays the report in a window and sends it to the file you specified in the **Report Output Options** dialog box.

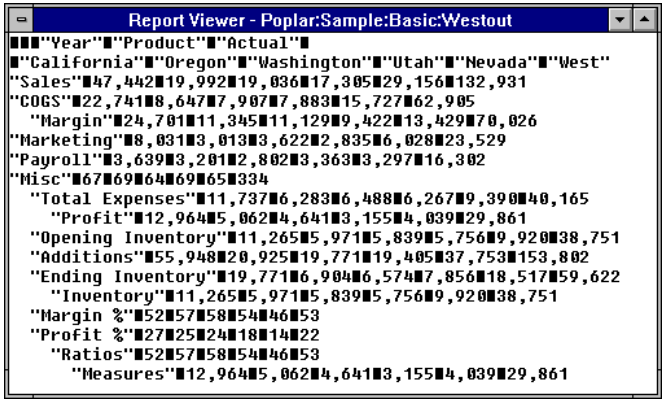

Figure 38-9, Sample Basic Report Script Output

If you are *not* using your Personal Essbase machine, remember to download and copy the file from the Essbase client directory to the `\ARBORPATH\APP\appname\dbname` on your Personal Essbase server. For example, copy the output file to `C:\ESSBASE\APP\WEST\WESTMKTS\WESTOUT.TXT`.

You are now ready to load the text file into your new database.

Loading the Output File Into the New Database

Load the output file into the new database on your Personal Essbase machine.

1. In Application Manager on your Personal Essbase server, select the new database. For example, select `Westmkts`.

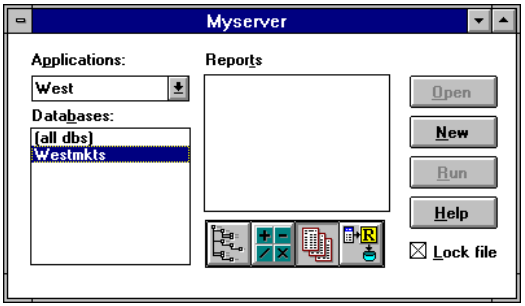

Figure 38-10, Westmkts Database Selected in Application Manager

2. Choose Database | Load Data from the Application Manager main menu.

   The **Data Load** dialog box appears.

3. Click **Find** to open the **Open Server Data File Objects** dialog box.

   - In the **Objects** list box, select the ASCII text file you have just created. For example, select `WESTOUT`.
   - Click OK.

Note: If WESTOUT does not appear, check that you gave it a `.TXT` extension, and placed it in the `\`*`ARBORPATH`*`\APP\WEST\WESTMKTS` directory. See "Creating an Output File Containing the Required Data Subset" on page 38-7.

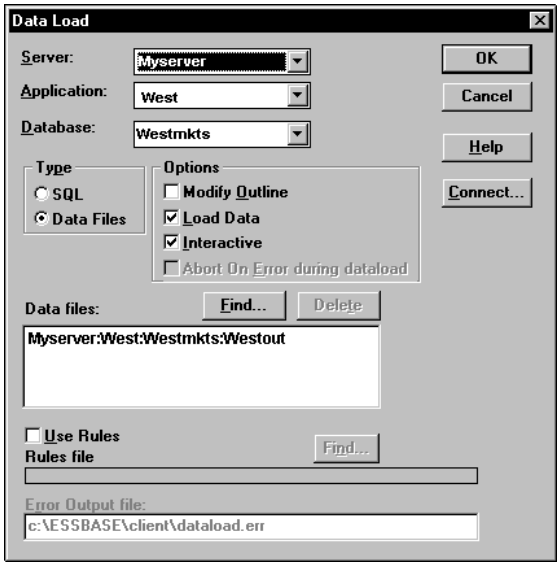

Figure 38-11, Data Load Dialog Box with WESTOUT Selected

4. Click OK in the **Data Load** dialog box to load the text file into the new database. For detailed information on loading data and any errors that may occur, see Chapter 19, Introducing Data Loading.

You can now view the data on your Personal Essbase machine. You might need to recalculate the database subset. Because you are viewing a subset of the database, a percentage of the data values will be `#MISSING`.

If required, you can copy report scripts and other object files to your Personal Essbase machine to use with the database subset you have created.

Using Report Scripts for Data Exporting

You can use report scripts to export Essbase data to other programs in text format. The Report Writer lets you create text files that meet the import format specifications of most other programs.

Before you can import data into some programs, you must separate, or delimit, the data with specific characters.

If you plan to import Essbase data into a program that requires special delimiters, use the MASK command. For the syntax and usage of Report Writer commands, see the online *Technical Reference* in your `DOCS` directory.

When you export data to a program that uses a two-dimensional, fixed-field format, you do not need to specify page or column dimensions. To create a two-dimensional report, you can specify every dimension as a row dimension. Use the ROWREPEAT command to add the name of each member specified to each row (rather than the default, nested style). The following script example and report illustrate this situation for a five-dimensional database:

```
<ROW (Year, Measures, Product, Market, Scenario)
{ROWREPEAT}
<ICHILD Year
Sales
<ICHILD "400"
East
Budget
     !
```

This example produces the following report:

```
Qtr1        Sales      400-10       East      Budget       900
Qtr1        Sales      400-20       East      Budget     1,100
Qtr1        Sales      400-30       East      Budget       800
Qtr1        Sales        400        East      Budget     2,800
Qtr2        Sales      400-10       East      Budget     1,100
Qtr2        Sales      400-20       East      Budget     1,200
Qtr2        Sales      400-30       East      Budget       900
Qtr2        Sales        400        East      Budget     3,200
Qtr3        Sales      400-10       East      Budget     1,200
Qtr3        Sales      400-20       East      Budget     1,100
Qtr3        Sales      400-30       East      Budget       900
Qtr3        Sales        400        East      Budget     3,200
Qtr4        Sales      400-10       East      Budget     1,000
Qtr4        Sales      400-20       East      Budget     1,200
Qtr4        Sales      400-30       East      Budget       600
Qtr4        Sales        400        East      Budget     2,800
  Year      Sales      400-10       East      Budget     4,200
  Year      Sales      400-20       East      Budget     4,600
  Year      Sales      400-30       East      Budget     3,200
  Year      Sales        400        East      Budget    12,000
```

Note: If you want to create a two-dimensional report that contains only bottom-level (level 0) data, use CHILDREN or DIMBOTTOM to select level 0 members.

- To list only level 0 data for specific members, use the CHILDREN command with the level 1 member as a parameter above the data you want to print.
- To list only level 0 data for a given member (including a dimension), use the DIMBOTTOM command as a parameter with the dimension name that contains the data you want to print.

For example, the following script uses the CHILDREN command to select the children of Qtr1, which is a level 1 member, and the DIMBOTTOM command to select all level 0 data in the Product dimension.

```
<ROW (Year, Measures, Product, Market, Scenario)
{ROWREPEAT}
{DECIMAL 2}
<CHILDREN Qtr1
Sales
<DIMBOTTOM Product
East
Budget
     !
```

This example produces the following report:

```
Jan      Sales    100-10     East       Budget        1,600.00
Jan      Sales    100-20     East       Budget          400.00
Jan      Sales    100-30     East       Budget          200.00
Jan      Sales    200-10     East       Budget          300.00
Jan      Sales    200-20     East       Budget          200.00
Jan      Sales    200-30     East       Budget        #Missing
Jan      Sales    200-40     East       Budget          700.00
Jan      Sales    300-10     East       Budget        #Missing
Jan      Sales    300-20     East       Budget          400.00
Jan      Sales    300-30     East       Budget          300.00
Jan      Sales    400-10     East       Budget          300.00
Jan      Sales    400-20     East       Budget          400.00
Jan      Sales    400-30     East       Budget          200.00
Feb      Sales    100-10     East       Budget        1,400.00
Feb      Sales    100-20     East       Budget          300.00
Feb      Sales    100-30     East       Budget          300.00
Feb      Sales    200-10     East       Budget          400.00
Feb      Sales    200-20     East       Budget          200.00
Feb      Sales    200-30     East       Budget        #Missing
Feb      Sales    200-40     East       Budget          700.00
Feb      Sales    300-10     East       Budget        #Missing
Feb      Sales    300-20     East       Budget          400.00
Feb      Sales    300-30     East       Budget          300.00
Feb      Sales    400-10     East       Budget          300.00
Feb      Sales    400-20     East       Budget          300.00
Feb      Sales    400-30     East       Budget          300.00
Mar      Sales    100-10     East       Budget        1,600.00
Mar      Sales    100-20     East       Budget          300.00
Mar      Sales    100-30     East       Budget          400.00
Mar      Sales    200-10     East       Budget          400.00
Mar      Sales    200-20     East       Budget          200.00
Mar      Sales    200-30     East       Budget        #Missing
Mar      Sales    200-40     East       Budget          600.00
Mar      Sales    300-10     East       Budget        #Missing
Mar      Sales    300-20     East       Budget          400.00
Mar      Sales    300-30     East       Budget          300.00
Mar      Sales    400-10     East       Budget          300.00
Mar      Sales    400-20     East       Budget          400.00
Mar      Sales    400-30     East       Budget          300.00
```

See Chapter 36, Examples of Report Scripts, for another example of formatting for data export.

Importing Data Into Other Databases

Before you import data into some programs, you must delimit the data with specific characters. If you plan to import Essbase data into a program that requires special delimiters, use the MASK command. For the syntax and usage of Report Writer commands, see the online *Technical Reference* in your `DOCS` directory.

Part VII Managing Multidimensional Essbase Data Storage

Part VII describes how to manage data storage in Hyperion Essbase OLAP Server databases, including how to use the Hyperion Essbase Storage Manager and ensure data integrity. Part VII contains the following chapters:

- Chapter 39, Introducing the Essbase Storage Manager, introduces you to the Essbase Storage Manager and describes how to use it to retrieve data, compress data, allocate storage, handle fragmentation, and restructure databases.
- Chapter 40, Specifying Storage Manager Settings, describes how to set Storage Manager settings to specify the index cache size, the data cache size, the index page size, the isolation level, the disk volumes, and data compression using the Hyperion Essbase Application Manager and ESSCMD.
- Chapter 41, Ensuring Data Integrity, introduces you to transactions, isolation levels, data redundancy, and how Essbase locks data, commits data, rolls back transactions in the event of a server interruption, and checks structural integrity.

Chapter 39 Introducing the Essbase Storage Manager

This chapter describes how Hyperion Essbase accesses and stores data.

This chapter contains the following sections:

About the Storage Manager

The Hyperion Essbase Storage Manager is a layer of Essbase server software that communicates with the Hyperion Essbase OLAP Server. The server sends data requests to the Storage Manager.

The Storage Manager has the following functions:

- Handles disk and memory storage of Essbase files
- Handles data retrieval
- Controls Essbase-related I/O
- Consolidates free space for re-use
- Manages concurrent operations
- Recovers databases after a server crash
- Issues locks
- Manages transactions
- Handles data updates from the server

The Storage Manager contains several components that control all aspects of retrieving and storing data. The following table summarizes these components.

Table 39-1, Storage Manager Components

Component	Main Function
Index Manager	Finds and tracks the location of requested data
Data Manager	Fetches the data pointed to by the index, and stores the data
LRO Manager	Handles retrieval and storage of linked reporting object (LROs)
Lock Manager	Handles locking and release of data blocks to regulate concurrent data access
Transaction Manager	Tracks transactions and handles internal commit and abort operations

For information on how the client communicates with the server, see Chapter 44, Running Essbase, Applications, and Databases.

Index Manager

The Index Manager handles requests for data and determines when Essbase needs to access data on disk and bring it into the index cache.

The Index Manager controls the following components.

Component or Term	Main function
Index	Contains all index files. Collective term for all index files in a database.
Index File	Stores retrieval information in index pages. Essbase names each index file incrementally: ESS*xxxxx*.IND, where *xxxxx* is a number starting with 00001.
Index Page	Holds index entries that point to data blocks.
Index Entry	Points to a data block. An index entry exists for every intersection of sparse dimensions.
Index Cache	Holds multiple index pages in memory.

The *index* refers to all index files for a single database. The index can span multiple volumes, and more than one index file can reside on a single volume. Use the Disk Volumes setting to specify disk spanning parameters. See Chapter 40, Specifying Storage Manager Settings, for information on setting index cache size and index page size, and allocating space with the Disk Volumes setting.

Data Manager

The Data Manager brings data blocks into memory, writes them out to disk, and handles data compression. The Data Manager controls the following components.

Component	Main Function
Data File	Stores retrieval information in data blocks. Essbase names each data file incrementally: ESS*xxxxx*.PAG, where *xxxxx* is a number starting with 00001.
Data Block	A multidimensional array representing cells of the dense dimensions for a given intersection of sparse dimensions.
Data Cache	Holds data blocks in memory.

The data cache size and the data block size determine how many data blocks can fit into memory at one time. Data files for a single database can span multiple volumes; more than one database can reside on the same volume. See Chapter 40, Specifying Storage Manager Settings, for information on specifying data cache size and allocating disk space.

Note: This documentation refers to a *data file* as an individual file in the database, but Essbase users often refer to the data file or *page file* as a collective unit that contains all .PAG files for a single database and can span multiple volumes.

Allocation Manager

The Allocation Manager manages index and data storage on disk, including:

- File open and close operations
- Which volume to use for a new file
- Sequence of volume use

The Allocation Manager allocates space based on the Disk Volumes setting. You can specify Disk Volumes in the Application Manager or with ESSCMD. See Chapter 40, Specifying Storage Manager Settings, for information on the Disk Volumes setting.

The Allocation Manager makes decisions about when and where to create a new file. First, it attempts to place updated data in a free fragment from an existing file. Next, it attempts to expand an existing file. If it cannot expand the existing file or write to a new file on the same volume, it posts a warning and attempts to use the next specified volume. If all specified volumes are full, an error message appears.

LRO Manager

Linked Reporting Objects (LRO) lets you associate objects, such as flat files, with data cells.

The Storage Manager stores information about linked reporting objects in an LRO catalog. Each catalog resides in its own Essbase index page, and coexists in an index file (`ESSxxxxx.IND`) with other, non-LRO Essbase index pages. Users can create and store LRO files, with an `.LRO` extension, in the `BIN` directory (for example, `\ESSBASE\BIN`).

For more information about linked reporting objects, see Chapter 11, Linking Objects to Your Essbase Data, and the *Spreadsheet Add-in User's Guide*.

Lock Manager

The Lock Manager issues locks on data blocks and controls concurrent access to data. The *committed access* and *uncommitted access* isolation levels use different locking schemes. For more information on isolation levels and locking, see Chapter 41, Ensuring Data Integrity.

Transaction Manager

The Transaction Manager controls transactions and commit operations, and assists with database recovery.

Essbase commits data automatically, triggered by transactions that modify data: data loading, calculating, restructuring, and spreadsheet Lock and Send operations.

Essbase commits data differently depending on whether your transaction isolation level is set to Committed or Uncommitted access (the default). See Chapter 40, Specifying Storage Manager Settings, for information on specifying isolation level.

The Transaction Manager maintains a transaction control table, *`dbname`*`.TCT`, to track transactions.

See Chapter 41, Ensuring Data Integrity, for more information on commit operations and recovery.

Storage Manager Startup

This section provides a summarized, high-level list of the steps that start up the Storage Manager.

1. A user starts a database.
2. Essbase loads the database.
3. The Essbase Agent passes database settings to the server.
4. The Storage Manager begins its initialization process.
5. The Storage Manager starts its components: the Index Manager, Lock Manager, Linked Reporting Objects (LRO) Manager, Data Manager, and the Transaction Manager.

If it encounters a fatal error, the Storage Manager shuts itself down and re-starts, repeating steps 2 through 5. "Fatal Error Handling" on page 39-5 describes fatal errors.

Fatal Error Handling

When the Storage Manager encounters a fatal error, it shuts down and re-starts, attempting to re-initialize itself and proceed with database recovery. When this happens, Essbase displays error messages similar to the following:

```
Reinitializing Storage Manager due to a fatal error.
```

(This message is followed by other informational messages related to database recovery.)

```
Invalid transaction handle passed to Storage Manager.
```

When you see either of these messages, you know that the Storage Manager shut itself down and is attempting to start up. Check the server log and determine whether Essbase issued a fatal error message just before generating the re-initialization messages. See Chapter 45, Monitoring Performance Using Diagnostics, for information about the server log. See "Transaction Rollback" on page 41-9 for more information about recovering transactions.

- If the Storage Manager did encounter a fatal error, in most cases you need to start over any operation that was active at the time of the fatal error. If the operation was a calc script or a data load, you might be able to continue where it left off; check the server log to see how far Essbase processed the operation. When in doubt, start the operation over.
- If the Storage Manager did not encounter a fatal error, contact your software provider's Technical Support to determine what caused the Storage Manager to shut down and re-start.

The Storage Manager considers the following errors "fatal:"

- One or more control fields have unexpected or inconsistent values.
- The Storage Manager detects data corruption.
- The Storage Manager cannot perform an operation that is necessary to ensure data integrity.
- The Storage Manager encounters a condition that could lead to data corruption.

Retrieving Data Through the Storage Manager

When you perform any operation resulting in a data request, the index searches for the requested data block(s). Figure 39-1 shows how the Storage Manager retrieves data if the data is not already in memory.

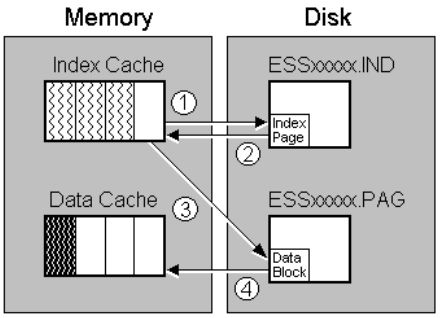

Figure 39-1, How the Storage Manager Retrieves Data

If a requested data block is not already in the data cache, the Storage Manager retrieves it as follows:

1. The Index Manager, unable to find the appropriate index page in the index cache, searches the index on disk and retrieves the index page that corresponds to the requested data.
2. Essbase pages the appropriate index page to the index cache.
3. The index page contains the index entry that points to the location of the requested data block. Essbase follows the pointer to the data block.
4. Essbase retrieves the data block and pages it to the data cache.

When the data is in the data cache, it is available for modification.

Data Compression

A data compression setting determines whether Essbase compresses data blocks when it writes them out to disk. Essbase fully expands the compressed data blocks, including empty cells, when the blocks are swapped into memory. Data compression is enabled by default. Generally, data compression optimizes storage use.

There are two types of data compression: bitmap and Run-Length Encoding (RLE).

Bitmap Data Compression

With this type of compression, Essbase uses a bitmap to represent data cells, and stores only the bitmap, block header, and other control information. A bitmap uses only one bit for every cell in the data block, including the missing values. When a data block is not compressed, Essbase uses eight bytes to store every non-missing cell.

When using bitmap compression, Essbase stores only non-missing values and does not compress repetitive values or zeros (contrast with RLE compression, described in "RLE Data Compression" on page 39-8). When Essbase pages a data block into memory, it fully expands the data block, using the bitmap to recreate the missing values.

Because the bitmap uses one bit for every cell (whether or not the value is missing) in the data block, the bitmap scheme provides a fixed overhead for data compression. Figure 39-2 represents a portion of a data block, as an example. In this example, Essbase would use 64 bytes to store this data in the fully expanded block, but uses one byte (eight bits) to store the bitmap of the compressed data on disk. (Essbase also uses a 72-byte block header for each block, whether the block is compressed or not.)

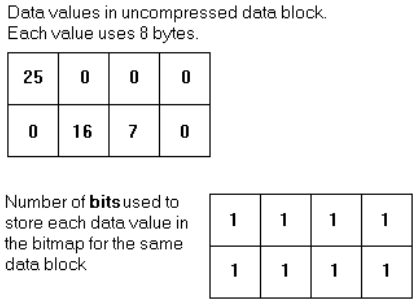

Figure 39-2, Bitmap Data Compression

RLE Data Compression

When using the Run-Length Encoding (RLE) compression scheme, Essbase compresses any consecutive, repetitive values—any value that repeats three or more times consecutively—including zeros. Essbase keeps track of each repeating value and the number of times it is repeated consecutively.

In the example in Figure 39-3, Essbase would use 64 bytes to store the data in the fully expanded block, but uses 40 bytes to store the compressed data on disk. (Essbase also uses a 72-byte block header for each block, whether the block is compressed or not.)

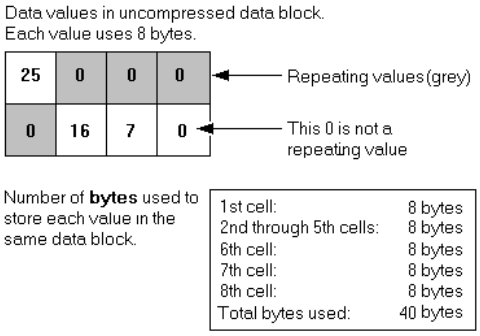

Figure 39-3, RLE Data Compression

Which Type of Compression Is Preferable?

In most cases, the bitmap compression type conserves disk space more efficiently. However, much depends on the configuration of your data.

The RLE compression type might be preferable under the following circumstances:

- The average block density in your database is very low (three percent or lower).
- Your database has many consecutive zero values, or other consecutive repeating values.

When Might You Want to Disable Data Compression?

You might want to disable data compression if the blocks already have very high density (90% or greater), and if there are not many consecutive repeating data values. Under these conditions, enabling compression would consume resources unnecessarily.

Changing the Data Compression Settings

Data compression is enabled by default. To change this option from the Application Manager, choose Database | Settings | Storage, and select an option from the **Data Compression** list box. Your options are `Bitmap` (the default), `RLE`, and `No Compression`.

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

When you change the data compression type, Essbase uses the option you choose when it writes out data blocks to the database. Changing the compression type has no effect on blocks already on disk (until they are accessed, updated, and written out again).

See"Specifying Data Compression" on page 40-21 for information on changing data compression settings.

Checking the Compression Ratio

The compression ratio represents the ratio of the compressed block size (including overhead) to the expanded block size, regardless of the compression type in effect. To check the compression ratio, choose Database | Information in the Application Manager and click the **Statistics** tab, or issue GETDBSTATS in ESSCMD.

Note: The compression ratio can vary widely from block to block.

Notes on Data Compression

Because Essbase compresses data blocks as they are written to disk, it is possible for bitmap, RLE, and non-compressed data blocks to coexist in the same data file.

- When a compressed data block is brought into memory, Essbase expands the block to its full size, regardless of the scheme that had been used to compress it.
- When Essbase stores a block on disk, Essbase treats the block the same whether the block was compressed or non-compressed when it was brought into memory. In either case, Essbase compresses the block according to the compression type you have specified, or does not compress the block if you chose no compression.
- If compression is not enabled, Essbase writes out the fully expanded block to disk.

Allocating Storage

Essbase automatically allocates storage for index (ESSxxxxx.IND) and data (ESSxxxxx.PAG) files. By default, Essbase uses the volume where your ARBORPATH directory resides, and uses all the available space on that volume. If you want to use multiple volumes to store your index and data files, or if you want to install Essbase on one volume but store the files on another, use the Disk Volumes setting.

The Disk Volumes setting lets you control how much space Essbase uses to store index files and data files and on which volumes (drives).

The Disk Volumes option lets you specify the name of a disk (or a letter corresponding to a disk drive, such as C). Essbase automatically stores the data and index files in the APP directory you are using for Essbase (for example, \ESSBASE\APP).

Essbase creates new data and index files when:

- The total sizes of all the files reach the maximum size specified in the Disk Volumes setting. By default, this total includes the index and data file sizes, added together. If you specify Index under **File Type**, this total refers to all the index files on that volume. Likewise, if you specify Data under **File Types**, this total refers to all the data files on that volume.

  For example, suppose you want to use up to 12 GB for Essbase files on volume E, 16 GB on volume F, and 16 GB on volume G. Essbase creates a new file on volume F when the sizes of your index and data files reach 12 GB on volume E, and more data needs to be written out to disk.

- The size of an individual index or data file on any volume exceeds two gigabytes. In the above example, suppose volumes E and F have reached their capacities and Essbase is using volume G. Figure 39-4 illustrates this example.

  Essbase creates file ESS00001.IND and fills it to the default limit of 2 GB.

  Essbase creates file ESS00001.PAG, which is now 1 GB in size.

  You have specified a limit of 16 GB on volume G, and you have used 3 GB of it. You have 13 GB left to use on volume G, but ESS00001.IND has reached the maximum file size of 2 GB. The next time Essbase needs more storage when writing index files to disk, Essbase creates a new file on the same volume and names it ESS00002.IND. Essbase will fill ESS00002.IND until its 2 GB limit, and will then create ESS00003.IND. Essbase does the same with data files.

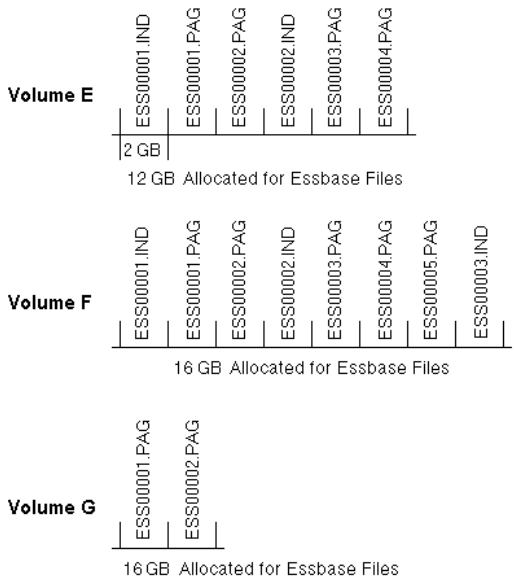

Figure 39-4, Example of How Essbase Stores Files Across Volumes

Essbase names files consecutively, starting with `ESS00001.xxx` where *xxx* is `IND` for an index file and `PAG` for a data file, and continuing up to `ESS99999.xxx`. This naming convention applies to each volume, so in the above example, volumes E, F, and G each have files named `ESS00001.PAG` and `ESS00001.IND`.

You can specify the Disk Volumes setting from the Application Manager menu (choose Database | Settings | Storage).

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

For more information on how to specify Disk Volumes and related settings, see "Specifying Disk Volumes" on page 40-16.

Fragmentation

Even when data compression is enabled, a small amount of fragmentation can occur. Fragmentation is unused disk space that results when Essbase writes a data block to a new location on disk, leaving unused space in the data block's former location.

The Storage Manager merges adjacent fragments into increasingly larger fragments so that the space is more likely to be re-used.

Restructuring Databases

As your business changes, your Essbase database outline usually changes to match. Some database outlines change frequently. Essbase has two common ways to change a database outline:

- Using the Outline Editor
- Using dimension building

If your changes to the outline are relatively minor, such as renaming a member name, there is little impact to your Essbase server. More significant changes, such as moving a member to a different parent, mean the data needs to be recalculated. The most significant changes are those which affect the way the data is stored on the disk. When the data storage needs to change, this is called *restructuring* the database. Changes which require restructuring the database are the most time-consuming, unless you discard the data before restructuring.

Essbase uses three types of restructure operations.

- Full restructure: Essbase restructures the blocks in the data files if a member of a dense dimension is moved, deleted, or added, and creates new data files. Essbase automatically regenerates the index whenever it restructures the data blocks so that the index entries point to the new locations of the data blocks. Essbase marks all restructured blocks as dirty, so you need to recalculate the database after a full restructure. Full restructuring is the longest of the restructures, and for large databases can take a relatively long time to complete.
- Index-only restructure: Essbase restructures the index if a member of a sparse dimension is moved, deleted, or added, and creates new index files. Restructuring the index is relatively fast, depending on the size of the index.
- Outline-only restructure: Essbase does not restructure the index or data files. Member name changes, creation of aliases, and dynamic calculation formula changes are examples of changes that affect only the database outline.

If you are using Intelligent Calculation in your database, then any time data blocks are restructured, all the restructured blocks are marked as dirty, and the next default Intelligent Calculation will be a full calculation. If you change a name or a formula, Essbase does not mark the block as dirty. Therefore, you would have to use another method to recalculate the member or the database.

If you change your outline frequently, you might want to consider enabling Incremental Restructuring. See "Incremental Restructuring" on page 39-16.

For more information about...	See...
Intelligent Calculation	Chapter 33, Optimizing Your Calculation Using Intelligent Calculation
Sparse and dense dimensions	Chapter 4, Basic Architectural Elements
Dimension building	Chapter 12, Introducing Dynamic Dimension Building
The Outline Editor	Chapter 8, Creating and Changing Database Outlines

How Essbase Restructures Data

When Essbase restructures both the data blocks and index, it uses the following files:

- *dbname*.IND
- ESSxxxxx.IND
- ESSxxxxx.PAG
- *dbname*.OTL
- *dbname*.ESM
- *dbname*.TCT

To perform a **full restructure**, Essbase does the following:

1. Creates temporary files that are copies of the .IND, .PAG, .OTL, .ESM, and .TCT files. The temporary files substitute the last character of the file extension to end in another character, so the file names are *dbname*.INN, ESSxxxxx.INM, ESSxxxxx.PAN, *dbname*.OTN, *dbname*.ESN, and *dbname*.TCU.
2. Reads the blocks from the old database files, restructures them in memory, and then stores them in the new temporary files. This step takes the most time.

3. Removes the old database files, including .IND, .PAG, .OTL, .ESM, and .TCT files.
4. Renames the temporary files to the correct file names: .IND, .PAG, .OTL, .ESM, and .TCT.

If an error or system failure occurs while Essbase is restructuring, it would most likely occur during step 2. To recover, you could simply restart the database. However, you need to delete the temporary files, both to free up disk space and to avoid conflicts the next time you restructure the database.

If a system failure occurred during steps 1, 3, or 4, then you would need to review the directory of your disk and evaluate how far the restructuring had progressed. Recovering involves removing the temporary files, or, if all but step 4 had been done, renaming the temporary files to the correct file names.

When Essbase restructures just the index, it uses the following files:

- *dbname*.IND
- ESSxxxxx.IND
- *dbname*.OTL
- *dbname*.ESM
- *dbname*.TCT

To perform an **index-only restructure**, Essbase does the following:

1. Renames the old ESSxxxxx.IND files to ESSxxxxx.INM .
2. Creates new index files (ESSxxxxx.IND) to store changed index information from the restructuring operation.
3. Removes the old index (ESSxxxxx.INM).

If a system failure occurred during any of these steps, then you would need to review the directory of your disk and evaluate how far the restructuring had progressed.

To recover your database, look for ESS00001.INM in your database directory (for example, \ESSBASE\APP\SAMPLE\BASIC).

- If ESS00001.INM exists, delete all ESS*.IND files and rename all ESS*.INM files to ESS*.IND.
- If ESS00001.INM does not exist, delete all ESS*.INM files and try the operation again.

Optimizing Restructure Operations

If your outline changes frequently, analyze your outline and the types of changes you're making. Remember, changes to sparse dimensions are relatively fast because only the index needs to change. Changes to dense dimensions are relatively slow because the data blocks need to be rebuilt.

The following are the types of restructure operations, in order of decreasing performance:

- Outline only (affects no `.IND` or `.PAG` files)
- Index (`.IND`) files only.
- Index and data (`.PAG`) files due to added, deleted, or moved members, and other operations as listed in Table 39-2.
- Index and data files due to a dense dimension becoming sparse or a sparse dimension becoming dense.

The following operations are slow:

- Changing a dense dimension to a sparse dimension.
- Changing a sparse dimension to a dense dimension.

To improve performance during restructure, consider doing the following:

- Restructure only input-level or level 0 data. These options require less disk space than restructuring all data. When you change the outline such that Essbase needs to restructure data, Essbase displays a dialog box that lets you choose to restructure only input-level or level 0 data. See "Saving a Modified Outline" on page 39-23.
- If you change some dimensions frequently, make them sparse.

Incremental Restructuring

If you make frequent changes to your outline, you might want to consider enabling incremental restructuring. When incremental restructuring is enabled, Essbase defers restructuring when you change the database outline or a dimension in a way that does not cause structural changes. Essbase restructures the index and, if necessary, the affected block the next time it is accessed.

When incremental restructuring is enabled, Essbase defers restructuring for many database changes. See Table 39-2.

The following changes override incremental restructuring; that is, they result in immediate restructuring, regardless of whether incremental restructuring is enabled.

- Adding or deleting a dimension
- Deleting a stored member of a sparse dimension
- Changing a dimension definition from sparse to dense, or from dense to sparse
- Changing the order of sparse dimensions
- Changing the order of dense dimensions
- Moving a member in a sparse dimension

Certain member additions or changes to sparse dimensions can also trigger immediate restructuring. For more information, see Table 39-2.

You can enable incremental restructuring for:

- An individual database in an application
- All databases in an application
- All databases in all applications

To enable incremental restructuring, use the INCRESTRUC parameter in your `ESSBASE.CFG` file. For more information on the INCRESTRUC parameter and syntax, see the online *Technical Reference* in your `DOCS` directory.

Essbase logs outline changes in an internal file, *`database_name`*`.OCL`. Essbase clears the file whenever it does a full database restructure, or when you clear or reset a database. *`database_name`*`.OCL` can grow quite large in the meantime. To clear this file, issue VALIDATE in ESSCMD. VALIDATE causes Essbase to restructure any blocks whose restructure was deferred, and clears the file. When you issue VALIDATE, make sure the database is *not* in Read-only mode (Read-only mode is used for archiving). For more information on the VALIDATE command, see "Checking Structural Integrity" on page 41-11.

Note: If you are using Linked Reporting Objects (LROs) in a database, incremental restructuring is automatically disabled on that database. When you have enabled incremental restructuring for all databases, the presence of an LRO in a database disables INCRESTRUC for that database, but does not affect the other databases on the server.

Restructuring Considerations

This section discusses features and options that cause Essbase to track additional information, and therefore affect performance.

Outline Change Log

If you activate the Outline Change Log, Essbase records all activity that affects the outline (member name changes, member moves, and so on). The more changes you make to the outline, the more updates Essbase must make to this log. By default, Essbase does *not* log outline changes. To activate the log, set OUTLINECHANGELOG to `TRUE` in the `ESSBASE.CFG` file. For more information about the log, see "The Outline Change Log File" on page 45-34.

IBM Relational Storage Manager

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

Essbase Partitioning

When you use Hyperion Essbase Partitioning, Essbase tracks outline changes so that you can synchronize the database outlines across partitions. Tracking outline changes slows restructuring performance, particularly when there are many structural changes.

If Essbase restructures data when you are using Partitioning, perform the following steps to make sure the data is synchronized across partitions:

1. Validate the partitions. Use the Partition Wizard's **Validate** page. See "Validating the Partition" on page 15-29 for more information.

Note: To validate a partition, you must have DB Designer or higher privileges.

2. Synchronize the partitions' outlines. Choose Database | Synchronize Outline to open the **Synchronize Outline** dialog box. See "Synchronizing Outlines" on page 15-40 for more information.

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

Impact of Common Outline Changes

The following table shows all operations that affect data values and restructuring. As a guideline, it is always a good idea to recalculate the database after any type of restructure operation.

The third column in this table refers to incremental restructuring. See "Incremental Restructuring" on page 39-16 if the term is unfamiliar to you.

For information about dynamic calculation members, see Chapter 28, Dynamically Calculating Data Values.

Note: If you are using Hyperion Essbase Partitioning, restructuring affects only the local, target database, not the database to which you are connected.

Table 39-2, How Actions Affect Databases and Restructuring

Action	Data and restructure implications	Incremental restructuring implications (if enabled)
Rename member	None	None
Change member formula	Data needs to be recalculated to reflect changes in formulas. No restructure.	None
Delete member of sparse dimension	Data needs to be recalculated to reflect changes to relationships. Essbase deletes pointers in the index file to blocks represented by the deleted member. Because the blocks are no longer pointed to, they become free fragments.	Essbase restructures the index; overrides incremental restructuring. For shared and label-only members: restructure deferred.
Delete member of dense dimension	Data needs to be recalculated to reflect changes to relationships. Essbase restructures the data file to reflect a changed block size. Essbase restructures the index file.	Restructure deferred
Delete member with shared attribute in sparse or dense dimension	Data needs to be recalculated. The data remains associated with the original member name, but because the parent of the shared member might have depended on the child's data, recalculation is needed. No restructure.	None
Add member to sparse dimension	Data needs to be loaded for new member or calculated to derive new values. Essbase restructures the index file.	Restructure deferred
Add member to dense dimension	Data needs to be loaded for new members or calculated to derive new values. Data needs to be recalculated. Essbase restructures the data file to reflect a changed block size. Essbase restructures the index file.	Restructure deferred

Table 39-2, How Actions Affect Databases and Restructuring

Action	Data and restructure implications	Incremental restructuring implications (if enabled)
Add member with shared attribute to sparse or dense dimension	Data needs to be recalculated. The new shared member will affect the consolidation to its parent. No restructure.	None
Re-define Dynamic Calc member as Dynamic Calc And Store	For dense dimension members: Essbase restructures both index and data files. For sparse dimension members: No restructure.	Restructure deferred
Re-define Dynamic Calc And Store member as Dynamic Calc	Essbase restructures both index and data files.	Restructure deferred
Re-define regular *dense dimension* member as Dynamic Calc And Store	Essbase restructures the index files.	Restructure deferred
Re-define regular *dense dimension* member as Dynamic Calc	Essbase restructures both index and data files.	Restructure deferred
Re-define *sparse dimension* Dynamic Calc And Store or Dynamic Calc member as regular member	No restructure	No restructure
Re-define *sparse dimension* regular member as Dynamic Calc or Dynamic Calc And Store	Essbase restructures both index and data files.	Restructure deferred
Re-define *dense dimension* Dynamic Calc And Store member as regular member	No restructure	No restructure
Re-define *dense dimension* Dynamic Calc member as regular member, and vice versa	Essbase restructures both index and data files.	Restructure deferred
Add, delete, or move *sparse dimension* Dynamic Calc member	Essbase restructures index files only.	For member add or delete: restructure deferred. For member move: Essbase restructures index files only (overrides incremental restructure).

Table 39-2, How Actions Affect Databases and Restructuring

Action	Data and restructure implications	Incremental restructuring implications (if enabled)
Add, delete, or move *sparse dimension* Dynamic Calc And Store member	Essbase restructures index files only.	For member add: restructure deferred. For member move or delete: Essbase restructures index files only; overrides incremental restructure.
Add, delete, or move a *dense dimension* Dynamic Calc And Store member	Essbase restructures both index and data files.	Restructure deferred
Add, delete, or move a *dense dimension* Dynamic Calc member	No restructure	No restructure
Move regular member within a sparse dimension	Data needs to be recalculated to reflect changes in consolidation. Essbase restructures the index file.	Essbase restructures the index file;overrides incremental restructure.
Move regular member within a dense dimension	Data needs to be recalculated to reflect changes in consolidation. Essbase restructures both index and data files.	Restructure deferred
Change dense/sparse setting	Data needs to be recalculated. Essbase restructures both index and data files.	Essbase restructures both index and data files; overrides incremental restructure.
Change Label Only setting	Data needs to be recalculated. Essbase restructures both index and data files.	Restructure deferred
Change shared member setting	Data needs to be recalculated to reflect changed data value of child. Essbase restructures both index and data files.	Restructure deferred
Change attribute (other than dense/sparse, label, or shared)	Data might need to be recalculated to reflect changed consolidation attributes, such as changing Time Balance from First to Last.	None

Table 39-2, How Actions Affect Databases and Restructuring

Action	Data and restructure implications	Incremental restructuring implications (if enabled)
Change the order of two sparse dimensions	No calculation or data load impact. Essbase restructures the index file.	Essbase restructures the index file; overrides incremental restructure.
Change the order of two dense dimensions or change the order of a dense and sparse dimension	Data needs to be recalculated. Essbase restructures both index and data files.	Essbase restructures both index and data files; overrides incremental restructure.
Change the order of dimensions	Data needs to be recalculated. Essbase restructures both index and data files.	Essbase restructures both index and data files; overrides incremental restructure.
Create, delete, clear, rename, or copy an alias table	None	None
Import an alias table or set member aliases	None	None
Change the case-sensitive setting	None	None
Name levels and generations	None	None
Create, change, or delete a user-defined attribute	None	None
Change a member alias or add an alias to a member	None	None

How Essbase Restructures after Outline Changes

This section tells you what to expect when you change, add, or delete dimensions.

Saving a Modified Outline

In the Outline Editor, if you change a dimension in the outline, such that Essbase needs to restructure the data file, Essbase displays the following dialog box when you save the outline changes:

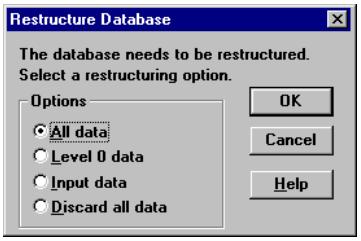

Figure 39-5, Restructure Database Dialog Box

The **Restructure Database** dialog box specifies a restructuring option for blocks. If the database contains data, you need enough free disk space on the server to create a backup copy of the database. This ensures that any abnormal termination during the restructure process will not corrupt a database.

- **All data**. This option preserves all data, but is generally slower than the other options.
- **Level 0 data**. This option preserves data only for level zero (leaf node) members. If you change the database and need to recalculate, and if all data required for the calculation is in level zero members, then this is the optimal restructure option.

  Selecting this option deletes all upper level blocks before restructuring. This reduces the disk space required to restructure and improves calculation times when the database is recalculated. Essbase recreates the upper-level blocks when you calculate the database.

- **Input data**. This option preserves only those blocks containing data that is loaded. Many applications contain data that is entered at parent levels. Selecting this option prevents any blocks created by data loading from being deleted, whether they are upper- or lower-level blocks. If you change the database and need to recalculate, and if you load data into various levels of your outline, then this is the optimal restructure option.

  Selecting this option deletes all blocks that contain strictly calculated values before restructuring. This reduces the disk space required to restructure and improves calculation times when the database is recalculated.

- **Discard all data.** This option preserves no data. Use this option when you expect to reload the data or when your outlines is so radically changed that no existing data applies.

Saving an Outline with One or More New Dimensions

If you add one or more new dimensions, then any data previously existing in the database must be associated with one member of each new dimension. For example, adding a dimension called Channel to the Sample Basic outline implies that all previous data in Sample Basic was associated with a particular channel or the sum of all channels. See "Adding and Deleting Dimensions and Members" on page 3-15 for more information.

If you add one or more new dimensions, the following dialog box appears:

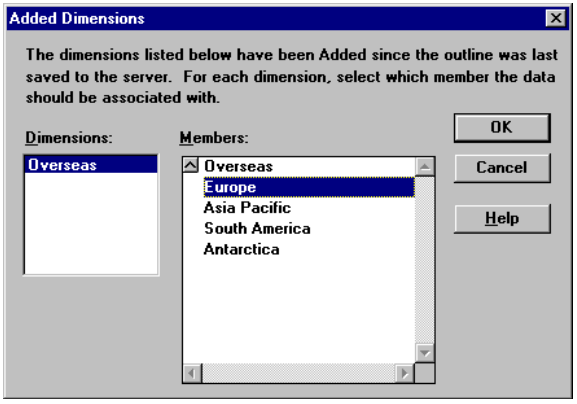

Figure 39-6, Added Dimensions Dialog Box

Saving an Outline with One or More Deleted Dimensions

If you delete one or more dimensions, then only the data associated with one member of each deleted dimension can be retained. For example, removing a dimension called Market from the outline implies that the data that will remain in the database after the restructure operation is associated with a specified member of the Market dimension. See "Adding and Deleting Dimensions and Members" on page 3-15 for more information.

One application of this technique is to delete a dimension from the database, such as Market, save the database as another name, such as New York, and specify the member to keep, such as New York. The new database is a working database for the New York market, and analysts in that region could use it for their analysis.

If you delete one or more dimensions, the following dialog box appears:

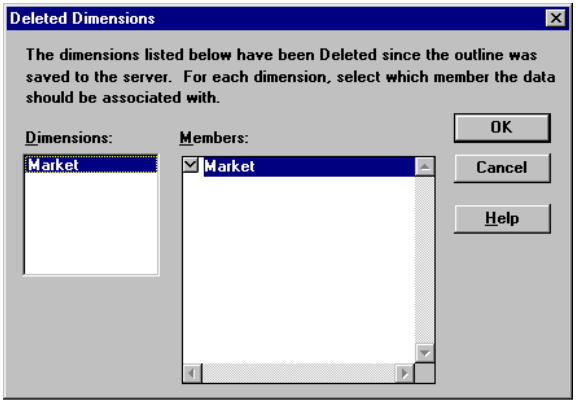

Figure 39-7, Deleted Dimensions Dialog Box

Notes:

- Essbase might display a "Restructuring not required" message, yet still perform an index-only restructure. This would most likely occur if you made changes to a sparse dimension.
- If you try to cancel a restructure operation, Essbase might issue a "Can't cancel" message. If that happens, Essbase is performing final cleanup and is at a stage of the restructure that is too late to cancel.

Chapter 40 Specifying Storage Manager Settings

This chapter provides a table listing the major Storage Manager settings, and tells you how to change them. It contains the following sections:

This chapter assumes that one of the following is true:

- You have read Chapter 14, Sizing Your Database, and Chapter 39, Introducing the Essbase Storage Manager.
- You are familiar enough with Essbase to know the settings values you want to use.

Note: **If you migrated from Version 4, read the migration information in the *Start Here* booklet before changing settings in this chapter.**

About Storage Manager Settings

You can customize Essbase storage settings at the database level—you can have different settings for every database on your server. The information in this section helps you understand what each setting controls and how to adjust it. For a table listing these and other performance-related settings, see Appendix A, Performance-Related Settings.

The major Storage Manager settings you can customize are:

- Index Cache Size
- Data Cache Size
- Isolation Level
- Disk Volumes
- Data Compression

This chapter focuses on how to specify these settings. For background information, see Chapter 39, Introducing the Essbase Storage Manager.

Where to Specify and Change Essbase Storage Settings

You can specify most storage settings using the **Database Settings** dialog box in Application Manager, or in ESSCMD.

Warning: Previous versions of Essbase let you specify some Storage Manager settings in the `ESSBASE.CFG` server file. In Version 5, Essbase ignores `.CFG` settings except when migrating from Version 4.x.

Default values depend on whether you're migrating from Version 4. If you are, Essbase takes the Version 4 value or its equivalent. If you are not migrating from Version 4, Essbase uses the default values discussed in this chapter.

Two ESSCMD commands control database settings: SETDBSTATE and SETDBSTATEITEM.

- SETDBSTATE exists for compatibility with Version 4 and uses the syntax `SETDBSTATE parameter value`, in interactive mode; for example, `SETDBSTATE accessLevel 1`.
- SETDBSTATEITEM, new with Version 5, controls more settings than SETDBSTATE and uses the syntax `SETDBSTATEITEM n`, in interactive mode. *n* is an integer corresponding to a parameter. For example, `SETDBSTATEITEM 4` corresponds to `accessLevel`; Essbase prompts you for database and application name.
- As with other ESSCMD commands, you can supply application name, database name, and parameter values on the command line.
- For syntax information about SETDBSTATE and SETDBSTATEITEM, see the online *Technical Reference* in your `DOCS` directory. For more information about ESSCMD, see Chapter 43, Performing Interactive and Batch Operations Using ESSCMD.

Essbase uses the most recent setting, whether you use SETDBSTATEITEM, SETDBSTATE, or the **Database Settings** dialog box.

Table 40-1, Scope of Essbase Settings

Where set	Scope	When effective	Overrides
Server Settings dialog box	Server-wide (all databases in all applications)	Immediately	• `ESSBASE.CFG` settings[1] • Essbase installation or migration defaults
Application Settings dialog box	Application-wide (all databases within the application)	Immediately	• `ESSBASE.CFG` settings[1] • SetAppState command • Essbase installation or migration defaults
SetAppState command in ESSCMD	Application-wide (all databases within the application)	Immediately	• `ESSBASE.CFG` settings[1] • **Application Settings** dialog box • Essbase installation defaults

Table 40-1, Scope of Essbase Settings

Where set	Scope	When effective	Overrides
Database Settings dialog box	Database-specific	For index cache, data cache, and disk volumes settings: upon stop and restart of database. For index page size: when database is started but not loaded. For Isolation Level and Concurrency Parameters: the next time there are no active transactions. For all other settings: immediately.	• ESSBASE.CFG settings[1] • SetDbState settings • Essbase installation defaults
SetDbStateItem < command in ESSCMD	Database-specific	For index cache, data cache, and disk volumes settings: Upon stop and restart of database. For index page size: when database is started but not loaded. For Isolation Level, Time-Out, and Pre-Image Access: The next time there are no active transactions. For all other settings: immediately.	• ESSBASE.CFG settings[1] • **Database Settings** dialog box • Essbase installation defaults

1. In previous releases of Essbase, the ESSBASE.CFG server file controlled some Storage Manager settings. ESSBASE.CFG affects all databases and applications. Starting with Version 5, specify all Storage Manager parameters in the Application Manager or by using ESSCMD. **The Storage Manager uses ESSBASE.CFG for initial settings when migrating from Version 4 only**, and ignores ESSBASE.CFG after migration.

Precedence of Settings

SETAPPSTATE, SETDBSTATE, and SETDBSTATEITEM can override dialog box settings, and vice versa. Using ESSCMD and dialog boxes are different ways to change the same settings. The setting's most recent value overrides any previously set value, whether it was set through a command or a dialog box. Both methods permanently override `ESSBASE.CFG` settings, as the following paragraph explains.

Essbase reads the `ESSBASE.CFG` file when you start the server, and then applies Application Manager and ESSCMD settings. Essbase does not use `ESSBASE.CFG` for Storage Manager settings, except during migration from Version 4.

For information on stopping and starting servers, applications, and databases, see Chapter 44, Running Essbase, Applications, and Databases.

Using the Settings Dialogs in the Application Manager

You can set database, application, and server options using the settings dialog boxes in the Application Manager. This section covers database settings. See Chapter 16, Managing Security at Global and User Levels, for application and server settings information.

To check or customize your database settings, choose Database | Settings in the Application Manager.

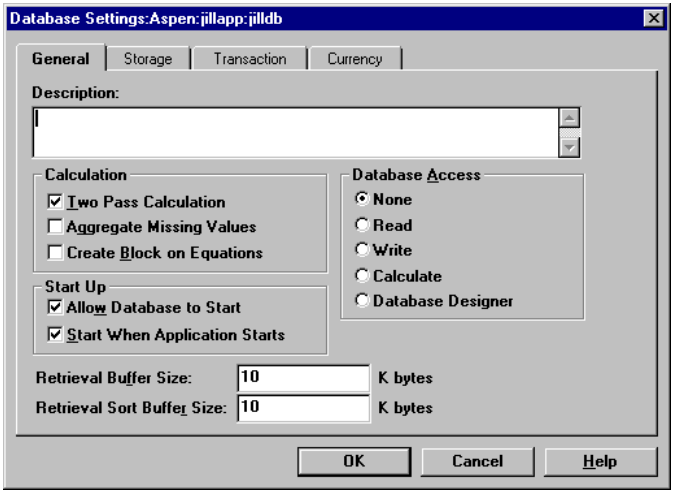

Figure 40-1, The Database Settings Dialog Box

The dialog box has four tabs, which correspond to the following pages.

The **General** page includes settings for:

- Calculations—see Part V, Calculating Your Data, for more information.
- Database access—see Chapter 16, Managing Security at Global and User Levels for more information.
- Database startup—see the Application Manager online help for more information.
- Retrieval buffers—see Chapter 37, Optimizing Your Reports, for more information.

The **Storage** page includes settings for:

- Data compression—see "Specifying Data Compression" on page 40-21 for more information.
- Data cache size—see "Specifying Data Cache Size" on page 40-9 for more information.
- Index cache size—see "Using the Settings Dialogs in the Application Manager" on page 40-5 for more information.
- Index page size—see "Specifying Index Page Size" on page 40-11 for more information.
- Location/Disk Volumes—see "Specifying Disk Volumes" on page 40-16 for more information.

The **Transaction** page includes settings for:

- Isolation level—see "Specifying Isolation Level" on page 40-12 for more information.
- Synchronization point—see "Specifying Isolation Level" on page 40-12 for more information.
- Concurrency parameters—see "Specifying Isolation Level" on page 40-12 for more information.

The **Currency** page includes settings for currency databases. See Chapter 42, Designing and Building Currency Conversion Applications, for more information.

Notes:

- Changes you make in any pages of this dialog box affect only the database you specify.
- For information about when settings take effect, see Table 40-1.

Using ESSCMD for Database Settings

You can use the SETDBSTATEITEM or SETDBSTATE command in ESSCMD to set or change database settings. See the online *Technical Reference* in your `DOCS` directory for information about these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Notes:

- Database settings you set or change with these commands affect only the specified database.
- Some settings take effect immediately, and others take effect the next time the database is started. See Table 40-1 for details.
- Use GETDBSTATE to view current settings.
- The ESSCMD commands, such as GETDBSTATE and SETDBSTATEITEM, as well as the **Database Settings** and **Database Information** dialog boxes, reflect changes to Data Cache size, Index Cache size, Index Page size, and Disk Volumes settings, *before* the changes take effect (that is, before the database has been re-started).
- For information on stopping and starting databases, see Chapter 44, Running Essbase, Applications, and Databases.

Using SETDBSTATEITEM in Batch Mode

You can include SETDBSTATEITEM (or SETDBSTATE) in batch scripts. For information about writing batch scripts, see Chapter 43, Performing Interactive and Batch Operations Using ESSCMD.

For simple items, specify the command, item number, application, database, and value.

```
SETDBSTATEITEM 2 "JILLAPP" "JILLDB" "Y";
```

For items that require multiple parameters, such as Isolation Level (item 18), specify multiple parameters.

```
SETDBSTATEITEM 18 "SAMPLE" "BASIC" "1" "Y" "-1";
```

To accept a default value or leave a value the same as it was before, use blanks enclosed in quotation marks.

```
SETDBSTATEITEM 18 "SAMPLE" "BASIC" "1" " " " " ";
```

Use a separate SETDBSTATEITEM command for each parameter; you cannot string item numbers together on the same line.

Specifying Index Cache Size

Index cache size is an important setting; it determines the maximum memory Essbase allocates to hold index information. This section tells you how to specify the setting, and assumes you have considered the value you want. See "Sizing Your Index and Data Caches" on page 14-12 for information about determining an appropriate index cache size.

When you specify a setting, such as `10M`, Essbase displays its corresponding value in bytes (B) in ESSCMD, and in kilobytes (K) in the **Database Settings** dialog box.

Using the Application Manager

Choose Database | Settings and click the **Storage** tab. Enter the desired value *in kilobytes* in the **Index Cache** text box.

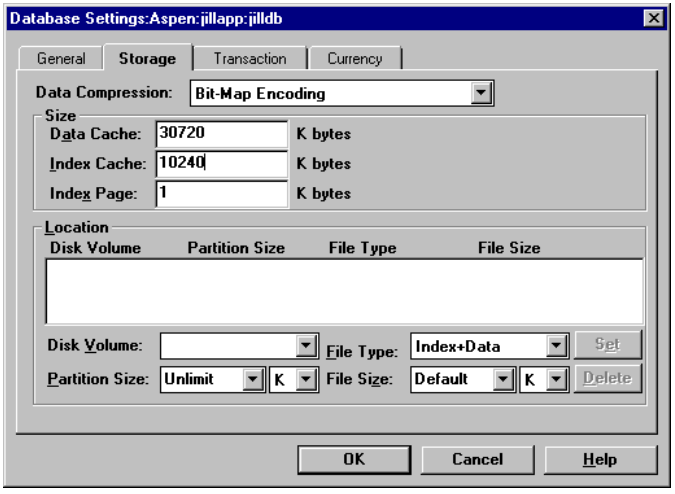

Figure 40-2, Setting Index Cache Size

Using ESSCMD

Enter `SETDBSTATEITEM 12` or `SETDBSTATE` in ESSCMD and follow the prompts, or supply the required values on the command line. For example,

```
SETDBSTATEITEM 12 "JILLAPP" "JILLDB" "10M"
```

ESSCMD lets you specify this value in bytes (B), kilobytes (K), megabytes (M), or gigabytes (G). ESSCMD displays minimum, maximum, and current values.

When Is This Change Effective?

Changes to this setting take effect the next time you start the database.

Specifying Data Cache Size

Data cache size is an important setting; it determines the maximum memory Essbase allocates to hold data. This section tells you how to specify the setting, and assumes you have considered the value you want. See "Sizing Your Index and Data Caches" on page 14-12 for information about determining an appropriate data cache size.

When you specify a setting, such as `30M`, Essbase displays its corresponding value in bytes (B) in ESSCMD, and in kilobytes (K) in the **Database Settings** dialog box.

The minimum value for this setting is 3MB (3145728 bytes). For compatibility with previous versions of Essbase, the Storage Manager accepts a value of 3000000, and automatically converts the value to the actual minimum of 3MB.

Using the Application Manager

Choose Database | Settings and click the **Storage** tab. Enter the desired value *in kilobytes* in the **Data Cache** text box.

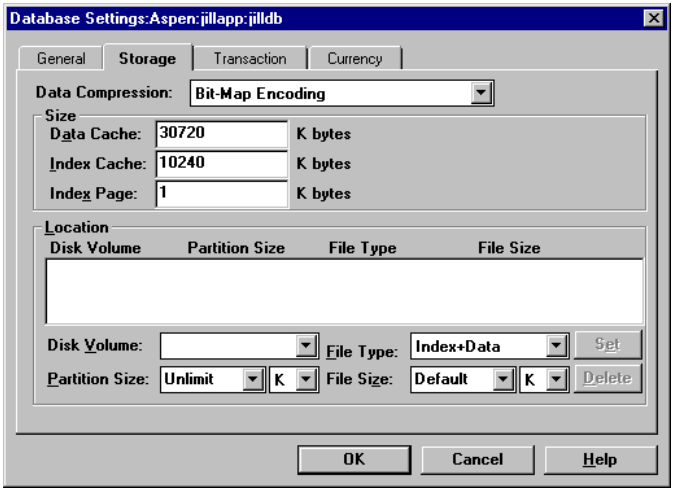

Figure 40-3, Setting Data Cache Size

Using ESSCMD

Enter `SETDBSTATEITEM 5` or `SETDBSTATE` in ESSCMD and follow the prompts, or supply the required values on the command line. For example,

```
SETDBSTATEITEM 5 "JILLAPP" "JILLDB" "30M"
```

ESSCMD lets you specify this value in bytes (B), kilobytes (K), megabytes (M), or gigabytes (G). ESSCMD displays minimum, maximum, and current values.

When Is This Change Effective?

Changes to this setting take effect the next time you start the database.

Specifying Index Page Size

The index page holds index entries; an index entry exists for every combination of sparse dimensions.

Note: The default setting, one kilobyte, is recommended.

Using the Application Manager

Choose Database | Settings and click the **Storage** tab. Enter the desired value *in kilobytes* in the **Index Cache** text box.

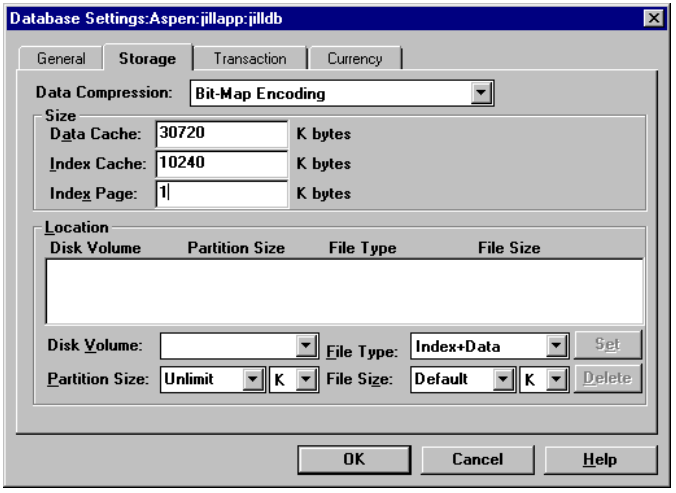

Figure 40-4, Setting Index Page Size

Using ESSCMD

Enter `SETDBSTATEITEM 13` or `SETDBSTATE` in ESSCMD and follow the prompts, or supply the required values on the command line. For example,

```
SETDBSTATEITEM 13 "JILLAPP" "JILLDB" "1024"
```

ESSCMD lets you specify this value in bytes (B), or kilobytes (K). The default unit is bytes. ESSCMD displays minimum, maximum, and current values.

When Is This Change Effective?

Changes to this setting take effect the next time you start the database; the database must be empty for this change to take effect.

Specifying Isolation Level

The Storage Manager offers two levels of isolation for transactions: *Committed* access and *uncommitted* access. Uncommitted access is the default.

Committed access restricts Read/Write access on data blocks to one transaction at a time; Essbase retains Write locks on all affected blocks until the transaction commits. *Uncommitted* access can produce unpredictable results, because Essbase acquires and releases Write locks as needed during the transaction. Uncommitted access is compatible with Version 4 transaction management. Committed access is recommended. For more information on these isolation levels, see Chapter 41, Ensuring Data Integrity.

Depending on which type of access you specify, Essbase uses additional parameters, which you can set in the Application Manager or in ESSCMD.

Notes on Isolation Level Options

If you choose Committed access, Essbase uses these parameters:

- Pre-Image access (enabled by default). Pre-Image access provides users Read-Only access to data blocks that are locked for the duration of another concurrent transaction. Users see the last committed data values for those data blocks.
- Wait (dialog box) or Time Out (ESSCMD).
    - Indefinite wait (the default): your transaction waits to acquire a lock on the required locked block(s).
    - Immediate access, or no wait: if a required block is locked for another transaction, your transaction does not start and Essbase posts a Time Out message.
    - A number of seconds that you specify: your transaction waits that number of seconds to acquire a lock on required locked blocks. If the specified time runs out before your transaction acquires a lock, Essbase posts a Time Out message.

Warning: Under committed access, Essbase retains redundant data until a transaction commits. Allow disk space for double the size of your database to accommodate redundant data.

If you choose Uncommitted access, Essbase uses these parameters:

- Commit Block (number of blocks modified before internal commit). The default is 3000.

  If you set Commit Block to 0, which means no implicit commit, Essbase commits blocks at the end of a transaction.

  If you are migrating from Version 4.x, Essbase uses the value you specified for COMMITBLOCKS in the `ESSBASE.CFG` file. If you did not specify COMMITBLOCKS in Version 4, Essbase computes a default value based on the commit threshold instead of using the default of 0. See the Migration section of the *Start Here* booklet.

- Commit Row (number of rows to data load before internal commit). The default is 0, which means no implicit commit; Essbase commits rows at the end of the data load.
- If either Commit Block or Commit Row has a non-zero value, Essbase commits data when it reaches the first threshold. For example, if Commit Block is 10 but Commit Row is 0, and you load data, Essbase commits after 10 blocks are updated. If Commit Block is 5 and Commit Row is 5, and you load data, Essbase commits after 5 rows are loaded, or 5 blocks are updated, whichever happens first.

Warning: Essbase retains redundant data until it commits. Allow disk space for double the size of your database to accommodate redundant data, particularly if Commit Block and Commit Row settings are set to 0.

Using the Application Manager

Choose Database | Settings and click the **Transaction** tab. Click **Committed** or **Uncommitted** in the **Isolation Level** group.

Depending on which type of access you specify, the **Concurrency Parameters** group or the **Synchronization Point** group is available.

Suppose you want committed access, pre-image access, and a 60-second wait for locked data blocks. Specify settings as in the following example.

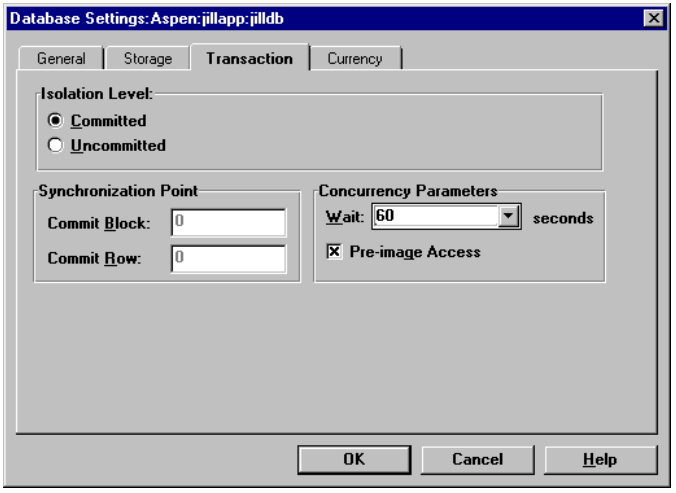

Figure 40-5, Example of Setting Committed Isolation Level

Suppose you want uncommitted access, and you want Essbase to perform a commit operation after 10 blocks have been updated, or after 10 rows have been loaded. Specify settings as in the following example.

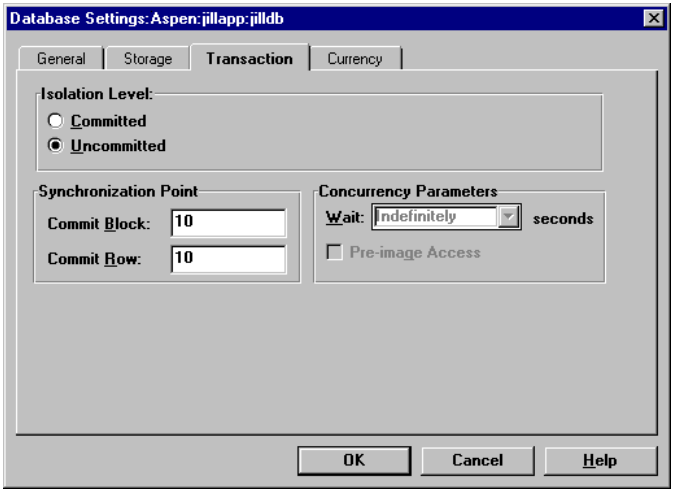

Figure 40-6, Example of Setting Uncommitted Isolation Level

Using ESSCMD

Enter `SETDBSTATEITEM 18` in ESSCMD and follow the prompts, or supply the required values on the command line.

Choose `1` (Committed access) or `2` (Uncommitted access, the default). Depending on which type of access you specify, ESSCMD prompts you for other parameters (or you can supply the values on the command line).

If you choose `1` (Committed access), ESSCMD prompts for:

- Pre-image access; `Y` (Yes) or `N` (No, the default). Pre-image access provides users Read-Only access to data blocks that are locked for the duration of a transaction. Users see the last committed data values for those data blocks.
- Wait (dialog box) or Time Out (ESSCMD): `-1`, `0`, or *n*.
  - `-1` is Indefinite wait
  - `0` is Immediate access, or no wait
  - *n* is a number of seconds that you specify

If you choose `2` (Uncommitted access), ESSCMD prompts for the following values. See "Notes on Isolation Level Options" on page 40-12 for important details about these options.

- Number of blocks modified before internal commit
- Number of rows to data load before internal commit

You can also specify Isolation Level parameters (Pre-image access and so on) by specifying parameters 19-22 on SETDBSTATEITEM. Enter `SETDBSTATEITEM` with no parameters; ESSCMD displays a list of each parameter by number, with a description.

Here is an example of using SETDBSTATEITEM to set isolation level. This example enables committed access and pre-image access, and specifies indefinite wait time.

```
SETDBSTATEITEM 18 "JILLAPP" "JILLDB" "1" "Y" "-1"
```

For more syntax information, see the online *Technical Reference* in your `DOCS` directory.

When Is This Change Effective?

Changes to this setting take effect the next time there are no active transactions.

Specifying Disk Volumes

Use Disk Volumes to specify where you want to store Essbase index files (ESSxxxxx.IND) and data files (ESSxxxxx.PAG). See "Allocating Storage" on page 39-10 for information about how Essbase allocates storage across volumes.

Essbase uses the following options in conjunction with Disk Volumes.

- Name of volume
- For each volume:
    - Maximum space to use on that volume (called Partition Size in Application Manager; Volume Size in ESSCMD)
    - File type. You can specify index files, data files, or both. The default is index and data files on the same volume.
    - Maximum file size. The default is 2097152 kilobytes (two gigabytes); this is the recommended value. When Essbase reaches the maximum file size, it creates a new file and names it incrementally. For example, when ESS00001.IND is filled to maximum size, Essbase creates ESS00002.IND.

You can specify disk volume names in any order (you do not have to use alphabetical order), but do specify the order in which you want the volumes used. You do not have to specify the volume where Essbase is installed as one of the volumes; you can install on one volume and store all data on other volumes.

You can tell Essbase to stop storing files on a particular volume. Essbase can still access that volume as needed; it just stops storing additional index and data information there. To do this, select the volume definition you want to remove as shown in Figure 40-7 and click **Delete**. Essbase stops using this disk for new information.

Notes:

- If you do not specify the Disk Volumes setting at all, Essbase stores data only on the volume where your ARBORPATH directory resides. If the ARBORPATH variable is not set, Essbase stores data only on the volume where the server was started.
- If the volume reaches capacity, Essbase issues a warning message before moving on to the next volume.
- If all specified volumes reach capacity, Essbase stops any ongoing database operations and issues an error message. If this happens, shut down the database, allocate more disk space, and restart the database.

You set Disk Volumes on a per database basis. It is possible for more than one database to use space on the same volume, so allocate space carefully. For example, if you specify 7 GB on Volume A for Database 1, and 7 GB on Volume A for Database 2, you have allocated 14 GB for Essbase files on Volume A.

Warning: If you specify a volume name, but not volume size, Essbase uses all available space on that volume.

For more information, see "Allocating Storage" on page 39-10.

Using the Application Manager

Choose Database | Settings | Storage.

To add a new volume, or to change values for a previously allocated volume:

1. Enter the volume name in the **Disk Volume** text box.
2. Specify **Partition Size**. This is the maximum space Essbase uses on that volume. The default value is `Unlimited`; the minimum is 8192 kilobytes.
3. Specify **File Type**. You can specify index files, data files, or both. The default is `Index + Data`.
4. Specify **File Size.** This is the maximum size that each file specified in **File Type** can attain before Essbase creates a new file. The default value is 2 GB (2097152 kilobytes); the minimum value is 8192 kilobytes.
5. Click **Set**.

To stop Essbase from storing additional files on an allocated volume (that is, to "deallocate" a volume), enter the volume name in the **Disk Volume** text box and click **Delete**. Essbase writes no new files to that volume, but continues to access and use files already created on that volume.

For example, suppose you do not want any more information to be stored on Volume F. Select the volume definition for volume F and press **Delete**, as shown in Figure 40-7.

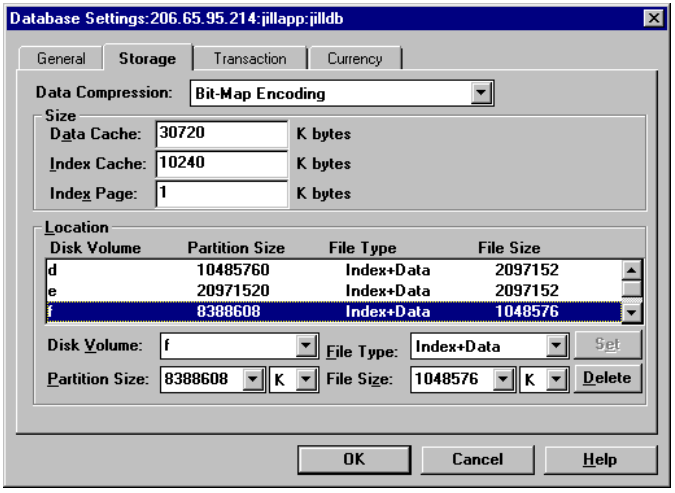

Figure 40-7, Deallocating a Volume

Now, Essbase stores no additional data on Volume F.

Allocating Storage: An Example

Suppose you want to use up to 20 GB for Essbase files on Volume E, 25 GB on Volume F, and 25 GB on Volume G. You are using the default file size limit of 2 GB.

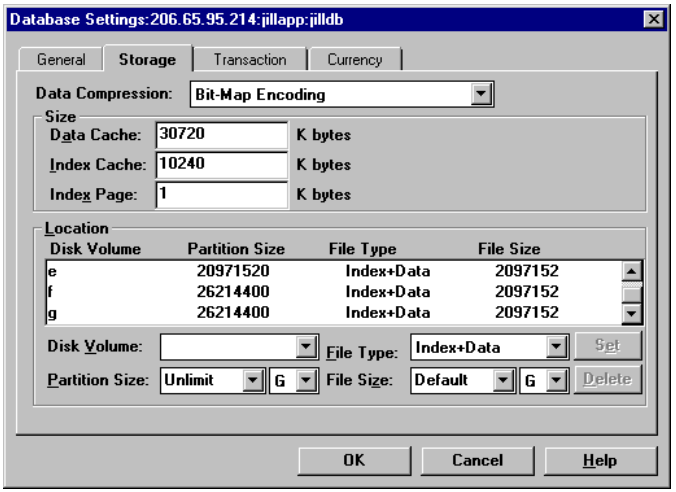

Figure 40-8, Example of Using Disk Volumes

When you load data, Essbase stores up to 20 GB on Volume E; if the database is larger than 20 GB, Essbase stores the next 25 GB on Volume F, and so on.

Using ESSCMD

To allocate a new volume, enter `SETDBSTATEITEM 23` in ESSCMD and follow the prompts, or supply the required values on the command line.

ESSCMD prompts you for the following values, unless you supply them on the command line:

- Number of new disk volume definitions
- Volume Name
- Volume Size (maximum space to use on that volume). The default value is Unlimited; the minimum setting is 8 megabytes.

  ESSCMD lets you specify this value in bytes (B), kilobytes (K), megabytes (M), or gigabytes (G). ESSCMD displays minimum, maximum, and current values, and 0 for Unlimited.

- File Type. You can specify index files, data files, or both. The default is `3-Index|Data` (index and data files on the same volume).
- File Size. This is the maximum size that each file specified in File Type can attain before Essbase creates a new file. The default value is 2 gigabytes; the minimum setting is 8 megabytes.

  ESSCMD lets you specify this value in bytes (B), kilobytes (K), megabytes (M), or gigabytes (G). ESSCMD displays minimum, maximum, and current values.

The following example allocates up to 10 gigabytes on Volume E, sets a maximum file size of 100 megabytes, and specifies that data files should only be stored on E:

```
SETDBSTATEITEM 23 "JILLAPP" "JILLDB" "1" "E" "10G" "2" "100M"
```

To change the settings on an allocated volume, enter `SETDBSTATEITEM 24` in ESSCMD and follow the prompts, or supply the required values on the command line.

ESSCMD prompts you for the following values, unless you supply them on the command line:

- Modify which volume definition
- Volume Name
- Volume Size
- File Type
- File Size

The following example allocates up to 20 gigabytes on Volume C, and sets a maximum file size of 1 gigabyte:

```
SETDBSTATEITEM 24 "JILLAPP" "JILLDB" "1" "C" "20G" "3" "1G"
```

To stop Essbase from storing additional files on an allocated volume, enter `SETDBSTATEITEM 25` in ESSCMD and follow the prompts, or supply the required values on the command line. Essbase continues accessing files on the deallocated volume, but does not write new files to it.

ESSCMD prompts you for the following value, unless you supply it on the command line:

- Delete which volume definition

The following example deallocates the fourth previously specified volume:

```
SETDBSTATEITEM 25 "JILLAPP" "JILLDB" "4"
```

- On UNIX, *volume_name* is a mounted UNIX file system. You must enter a fully qualified path name up to the name of the directory you are using for Essbase. Essbase automatically appends the `/app` directory to the path; do not specify the `/app` directory. For example:

  ```
  /vol2/essbase 10M
  ```

Partition Size is the maximum space, in kilobytes, allocated to the volume. The default value is `Unlimited`—Essbase uses all available space on that volume.

When Is This Change Effective?

Changes to this setting take effect when you stop and re-start the database, for new files. Existing files and volumes are not affected.

Specifying Data Compression

Essbase uses the Data Compression parameter to specify whether you want Essbase to compress data on disk, and which compression scheme to use. See Chapter 39, Introducing the Essbase Storage Manager, for information about data compression.

Essbase uses the following data compression schemes.

- Bitmap compression, the default. Essbase stores only non-missing values, using a mapping scheme.
- Run-Length Encoding (RLE). Essbase compresses repetitive, consecutive values, including zeros and `#MISSING` values.

You can also disable compression. Compression is enabled by default.

Using the Application Manager

To specify data compression:

1. Choose Database | Settings and click the **Storage** tab.
2. Choose a value from the **Data Compression** list box.

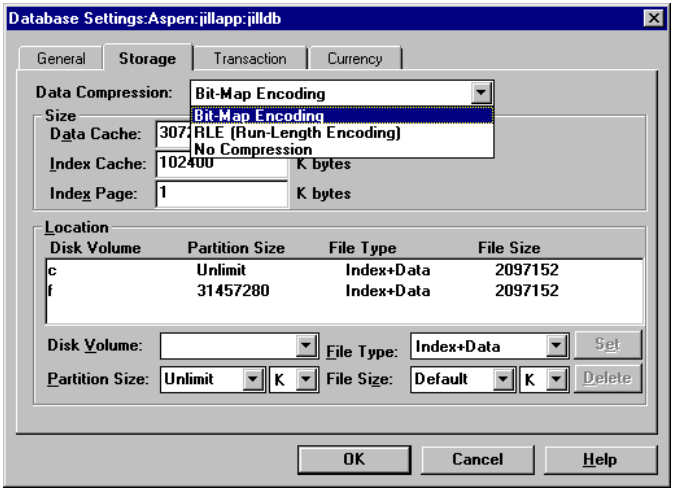

Figure 40-9, Setting Data Compression

Using ESSCMD

You can set data compression using SETDBSTATEITEM or SETDBSTATE.

Using SETDBSTATEITEM

To enable or disable data compression, enter `SETDBSTATEITEM 14` in ESSCMD and follow the prompts, or supply the required values on the command line.

ESSCMD prompts you for the following values, unless you supply them on the command line:

- Data Compression on Disk? Enter `Y` (yes, the default) or `N` (No).
- Data Compression Type. Enter `1` (Run-Length Encoding) or `2` (Bitmap, the default).

To specify data compression type, enter `SETDBSTATEITEM 15` in ESSCMD and follow the prompts, or supply the required values on the command line. ESSCMD prompts you for a value of `1` (Run Length Encode) or `2` (Bitmap, the default).

The following example enables Bitmap compression:

```
SETDBSTATEITEM 14 "JILLAPP" "JILLDB" "Y" "2"
```

For more syntax information, see the online *Technical Reference* in your `DOCS` directory.

Using SETDBSTATE

Enter `SETDBSTATE` in ESSCMD. Follow the prompts until the last prompt, `Data Compression on Disk?` Enter `Y` (yes, the default) or `N` (No).

To change compression type, use SETDBSTATEITEM or the **Database Settings** dialog box.

For more syntax information, see the online *Technical Reference* in your `DOCS` directory.

When Is This Change Effective?

Changes to this setting take effect as Essbase writes data blocks to disk. Essbase does not change compression schemes, or enable or disable compression, for blocks already on disk. For blocks already on disk when you change the data compression settings, Essbase uses the specified compression scheme the next time Essbase accesses, updates, and stores those blocks.

Controlling Data Block Size

Data block size is a configuration parameter that is not modifiable through settings dialogs or ESSCMD. This is determined by the amount of data in a particular combination of dense dimensions. Data block size is 8*n* bytes, where *n* is the number of cells that exist for that combination of dense dimensions. *Eight to 64 kilobytes is the optimum size range*.

Chapter 41 Ensuring Data Integrity

This chapter tells you how Essbase handles transactions and locking, and other ways Essbase protects your data.

This chapter includes the following sections:

If you are migrating your data from a previous release of Essbase, see the *Start Here* booklet.

What Is a Transaction?

When a database is in Read/Write mode, Essbase considers every update request (such as data load, calculation, or a statement in a calc script) a transaction. Essbase tracks information about transactions in a Transaction Control Table.

The Transaction Control Table (TCT) contains an entry for each transaction, and tracks its status as active, committed, or aborted.

About Isolation Levels

Essbase offers two isolation levels for accessing data: *committed access* and *uncommitted access* (the default). You can optimize data integrity by using committed access to data blocks.

Note: Some of the behavior of the Storage Manager depends on which type of access you have.

About Committed Access

Committed access provides a high level of data consistency. Only one transaction at a time can update data blocks. Under committed access, the Storage Manager issues short-term Read locks and long-term Write locks. Essbase provides options that determine when Essbase issues locks, as the following table shows.

Option	Description	Where Set
Wait or Time-out	Lets you decide whether (and how long) to wait for locked blocks to become available when you start an operation, or start the operation only if the required blocks are immediately available.	Application Manager: Database|Settings, **Transaction** page ESSCMD: SetDbStateItem 20
Pre-Image Access	Allows you to view data (in Read-only mode) when the blocks are locked by another transaction; you view data as it existed when it was last committed (before the other transaction started).	Application Manager: Database|Settings, **Transaction** page ESSCMD: SetDbStateItem 19

Notes:

- When you have Pre-Image access enabled, you are not limited to Read-only access to data blocks; if you need Write access to locked blocks, your transaction waits for Write access or times out, depending on the Wait or Time-out setting.
- If you do not have Pre-Image access enabled, and if you need Read or Write access to locked blocks, your transaction waits for Write access or times out, depending on the Wait or Time-out setting.
- Under committed access, Essbase retains some redundant data to ensure data consistency. To accommodate the data redundancy, allow disk space for double the size of your entire database.

- To enable committed access, change your access type, or change related options, choose Database | Settings and click the **Transaction** page in the Application Manager. For more information, see Chapter 40, Specifying Storage Manager Settings.

About Uncommitted Access

Uncommitted access uses only short-term Write locks, so the outcome of concurrent transactions is less predictable than with committed access. Uncommitted access is consistent with the concurrent activity under Essbase Version 4. See "How Essbase Handles a Transaction" on page 41-3 for more information.

For information about storage settings used with uncommitted access (Commit Blocks and Commit Rows), see "Specifying Isolation Level" on page 40-12.

How Essbase Handles a Transaction

Essbase tracks transactions from start to finish, swapping data blocks in and out of memory as needed, and committing them when the transaction completes. The following list describes how Essbase handles a transaction, and applies to both committed and uncommitted access.

1. A user or batch program begins an operation.
2. The OLAP engine issues an internal API call (`EssSMTransBegin`) to start the transaction.
3. The Storage Manager begins the transaction.
4. The OLAP engine requests data from the Storage Manager.
5. The Storage Manager locates the requested data. It passes the data, and some associated control information, to the OLAP engine. If you are using the Hyperion Essbase Spreadsheet Add-in, you see this data displayed on the sheet.
6. If you are using the Spreadsheet Add-in, and you modify the data, at this point you would issue the Send command.
7. The Storage Manager associates the transaction with an entry in its Transaction Control Table.
8. After the operation is complete on the OLAP engine side, the OLAP engine notifies the Essbase Storage Manager about the update, and the Storage Manager updates internal data structures accordingly.
9. Steps 4 through 8 repeat as often as necessary to complete the operation.

10. The transaction ends. If Essbase encountered an error during transaction processing, it aborts the transaction. If no errors were encountered, Essbase commits the transaction. See "Committing Data" on page 41-8 for details on the differences in commit behavior under committed and uncommitted access.

Essbase issues a message to notify the client that the transaction is complete; for example, `"TOTAL CALC ELAPSED TIME..."`

Transaction Handling under Uncommitted Access

Under uncommitted access, it is possible to access uncommitted data when multiple transactions are active at the same time. Transaction results are unpredictable under uncommitted access.

Notes:

- If you choose Essbase | Retrieve in the Spreadsheet Add-in, and you have Pre-Image Access enabled, Essbase displays a message if another, concurrent transaction has updated your data since your last Retrieve operation.

  For information about specifying Pre-Image Access, see Chapter 40, Specifying Storage Manager Settings.

- Under uncommitted access, if you have defined a commit threshold, Essbase might need to break down a single database operation into multiple transactions. See "Committing Data with Uncommitted Access" on page 41-8 for information on commit thresholds.

Locking Data

Essbase issues Write (exclusive) locks for blocks that are updated or deleted, and Read (shared) locks for blocks that need to be accessed but not updated. By issuing the appropriate locks, Essbase ensures that data changed by one operation cannot be corrupted by a concurrent update on the same data block.

Note: The Essbase Storage Manager locks data blocks, not objects. For information about locking and unlocking outlines and other objects, see "Locking and Unlocking Objects" on page 46-14.

Table 41-1, Basic Lock Types

Lock	Description
Write (exclusive) lock	Prevents any other transaction from accessing the locked data block. Used for all data block updates, including spreadsheet Lock and Send.
Read (shared) lock	Allows other transactions read-only access to the locked data block.

The following table shows the locks Essbase issues for various types of operations.

Table 41-2, Locking by Higher-Level Functions

Type of Operation	Lock Issued
Spreadsheet retrieve	Read (shared) lock on each data block.
Retrieve and lock	Write (exclusive) lock on all affected blocks. A subsequent Send command commits the data.
Calculate derived block	Write lock on block being calculated. As each block is calculated, any blocks containing that block's children acquire Read locks.
Data load	Write lock.
Restructure	Write lock.

Essbase handles locking differently, depending on whether committed or uncommitted access is enabled. For information on access type, see "What Is a Transaction?" on page 41-1.

Locking Under Uncommitted Access

Under uncommitted access, Essbase issues short-term Write locks; Essbase locks blocks only as it is updating them (contrast with committed access, when Essbase holds locks until a transaction completes).

Table 41-3 illustrates locking behavior in Essbase when more than one transaction is contending for a lock on the same data. In the example in Table 41-3, transaction Tx1 is already running, and transaction Tx2 is requesting access to the same data.

Table 41-3, Locking Behavior with Uncommitted Access

Request made by Tx2	If Tx1 holds Read lock	If Tx1 holds Write lock
Read lock	Tx2 gets Read lock	Tx2 gets Read lock
Write lock	Tx2 gets Write lock	Tx2 waits for Tx1 to release the lock

Locking Under Committed Access

Essbase issues short-term Read locks and long-term Write locks. A transaction locks and holds each block it updates until the transaction completes and then commits the block(s).

Table 41-4, illustrates locking behavior in Essbase when more than one transaction is contending for a lock on the same data. In the example in Table 41-4, transaction Tx1 is already running, and transaction Tx2 is requesting access to the same data.

Note that access to locked blocks depends on what options are enabled.

- **Pre-Image Access** lets you view data (in Read-only mode) when the blocks are locked by another transaction; you view data as it existed when it was last committed (before the other transaction started). Enabling Pre-Image access does not limit you to Read access; if you need Write access, or if you do not enable Pre-Image Access, you can use the Wait or Time-out option.
- **Wait** or **Time-out** lets you specify how long your transaction should wait for locked data blocks to become available. If you choose 0 seconds, Essbase issues a Time-out error if the blocks are not immediately available to your transaction.

  For information on setting these options, see Chapter 40, Specifying Storage Manager Settings.

Table 41-4, Locking Behavior Under Committed Access

		Tx1 holds Read lock; Tx2 requests Read lock	**Tx1 holds Read lock; Tx2 requests Write lock**	**Tx1 holds Write lock; Tx2 requests Read lock**	**Tx1 holds Write lock; Tx2 requests Write lock**
Pre-Image Access Enabled	Wait (Time-out) period specified	Tx2 gets Read lock.	Tx2 waits for Tx1 to release lock.	Tx2 gets Pre-Image access.	Tx2 waits for Tx1 to release lock.
	No wait (Time-out) period specified	Tx2 gets Read lock.	Essbase issues time-out message.	Tx2 gets Pre-Image access.	Essbase issues time-out message.
No Pre-Image Access	Wait (Time-out) period specified	Tx2 gets Read lock.	Tx2 waits for Tx1 to release lock.	Tx2 waits for Tx1 to release lock.	Tx2 waits for Tx1 to release lock.
	No wait (Time-out) period specified	Tx2 gets Read lock.	Essbase issues time-out message.	Essbase issues time-out message.	Essbase issues time-out message.

Concurrency Under Committed Access

Occasionally, a deadlock situation can result when two transactions are locking or waiting for access to the same blocks. For example, if transaction Tx1 needs to update data block B1 first and then block B2, it would lock B1 and attempt to lock B2. Meanwhile, if transaction Tx2 needs to update data block B2 first and then block B1, Tx2 would lock B2 and then attempt to lock B1. Tx1 is waiting for B2, and Tx2 locked B2 and is waiting for B1. Neither transaction can complete under these conditions.

Essbase transactions periodically perform deadlock detection. If a transaction is waiting to acquire a lock, it checks periodically. If the transaction detects a deadlock on a block it needs, the transaction aborts, and Essbase issues an error message.

If you try to update a lock that is locked to another user, depending on your Wait or Time-out setting, Essbase behaves as follows:

- If Wait is set to Indefinite, your transaction waits to acquire the needed locks.
- If Wait is set to 0 (Immediate), and if the required blocks are not immediately available, Essbase displays an error message.
- If Wait is set to a user-specified number of seconds, and the time has expired, Essbase displays an error message.

If your request times out, try the operation again.

For information on setting these options, see Chapter 40, Specifying Storage Manager Settings.

Committing Data

Essbase automatically commits any kind of update operation—data load, calculation, restructure, or spreadsheet operation—that modifies the data blocks.

Essbase handles commit operations differently under committed and uncommitted access.

You cannot explicitly commit data under either type of access; there is no COMMIT command.

Committing Data with Committed Access

As Essbase completes an operation, it commits the transaction. There is no explicit commit operation for the user to perform; Essbase handles it automatically.

Committing Data with Uncommitted Access

The following conditions cause Essbase to commit a transaction:

- An operation is completed.
- The number of modified data blocks reaches a user-defined threshold level. To set this level, choose Database | Settings in the Application Manager, click the **Transaction** tab, and use the **Synchronization Point** group. For more information, see Chapter 40, Specifying Storage Manager Settings.
- The number of loaded data rows reaches a user-defined threshold level. To set this level, choose Database | Settings in the Application Manager, click the **Transaction** tab, and use the **Synchronization Point** group. For more information, see Chapter 40, Specifying Storage Manager Settings.

If the user-defined threshold is exceeded during an operation, Essbase commits the data processed to that point and begins a new transaction to continue the operation. Essbase creates new transactions to continue the operation as many times as necessary.

Data Redundancy

To ensure data integrity, the Essbase Storage Manager temporarily retains redundant (duplicate) information. Some of this redundancy is due to the Safe Write feature; when data is modified, Essbase writes the data to a new location, still saving the old data until a commit takes place.

To accommodate redundant information, allow disk space for double the size of your database.

Essbase maintains a file called *dbname*.ESM, in which it stores crucial control information. The *dbname*.ESM file holds pointers to both the new and the old data block locations, and a main switch that determines which to use in an index search. The switch points to old data until Essbase successfully commits the data. At that point, Essbase sets the switch to point to the new data. This is known as the Safe Write feature.

Warning: The *databasename*.ESM file, the index file, and the data file contain information crucial for data recovery. Never alter or delete these files.

Transaction Rollback

If a server crash or other server interruption occurs while there are active transactions running, the Storage Manager rolls them back when the server is restarted.

Rollback with Committed Access

Essbase rolls back all database updates from aborted transactions, as well as any transaction that was in progress when the server stopped. Thus, Essbase ensures that the data is the same as it was before the aborted transaction began.

Rollback with Uncommitted Access

If the server crashes, Essbase rolls back all database updates from the point of the last successful commit. Some of the updates from an aborted transaction might have committed. Whether transactions successfully committed their updates the way users expected depends on the order in which overlapping transactions updated and committed data.

If a transaction is aborted, Essbase commits only the data it had finished processing when the transaction aborted.

What to Do When a Transaction Does Not Complete

Table 41-5 shows what action to take if a transaction did not complete, either because it aborted or due to a server interruption.

Table 41-5, Repeating Operations if the Transaction Stopped Before Completing

Operation	Action to take
Calculation	Start over.
Calc script	Check the server log and see how far the calc script got, and decide whether to start over or continue.
Normal data load	Start over.
Arithmetic data load (loads that perform operations on values in the database, such as adding 10 to each value)	Start where the data load left off.
All other operations	Start over.

Checking Structural Integrity

The VALIDATE command performs several types of structural integrity checks:

- Verifies the structural integrity of free space information in the index.
- Compares the index key in the index page with the index key in the corresponding data block.
- Restructures data blocks whose restructure was deferred with incremental restructuring. For more information, see "Incremental Restructuring" on page 39-16.
- Checks every block in the database to make sure each value is a valid floating point number.
- Verifies the structural integrity of the linked reporting objects (LRO) catalog.

As Essbase encounters mismatches, it records error messages in the VALIDATE error log file. You can specify a file name for error logging, or Essbase will prompt you for this information if you did not provide it. The VALIDATE utility continues running until it has checked the entire database.

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

During index free space validation, the VALIDATE command verifies the structural integrity of free space information in the index. If integrity errors exist, Essbase records them in the VALIDATE log file. The file that you specified on the VALIDATE command holds the error log.

If VALIDATE detects integrity errors regarding the index free space information, your database must be rebuilt. You can rebuild by:

- Restoring the database from a recent system backup
- Exporting data from the database; creating a new, empty database; and loading the exported data into the new database
- Restructuring the database

VALIDATE also compares index and data block keys. The Essbase index contains an index key for every data block. The corresponding data block itself contains the same key. The VALIDATE utility uses this key as a basis for comparison. You might want to run VALIDATE periodically to ensure data consistency.

Even without using VALIDATE, Essbase automatically performs certain validity checking whenever a read operation is performed, to ensure that the index is properly synchronized with the data.

For every read operation, Essbase:

- Compares the index key in the index page with the index key in the corresponding data block
- Checks other header information in the block

If Essbase encounters a mismatch, it displays an error message.

Part VIII Designing and Building Currency Applications

Part VIII describes how to design and build Hyperion Essbase OLAP Server currency applications. Part VIII contains the following chapter:

- Chapter 42, Designing and Building Currency Conversion Applications, introduces you to currency applications and describes how to create one by creating the main database outline, creating the currency database outline, and linking the two outlines. It also describes how to calculate currency conversions.

Chapter 42 Designing and Building Currency Conversion Applications

Organizations with offices in different countries generally do business in the currency of the host country (known as the *local currency*). World and regional headquarters of such organizations must convert data entered in local currencies to a *common currency* for consolidation and analysis. The Hyperion Essbase Currency Conversion option is designed to meet the needs of this complex business problem. This option can be licensed as an "add-on" to the Hyperion Essbase OLAP Server.

The Currency Conversion option includes a sample currency application that contains two databases: INTERNTL and XCHGRATE. This chapter provides background information, step-by-step instructions for building the sample currency application, and procedures for calculating and reporting currency conversions after you have built your own currency application.

This chapter contains the following topics:

Note: Your Essbase server installation includes the option to install the Sample databases, INTERNTL and XCHGRATE. If you don't have access to these databases, contact your Essbase administrator. For information about installing sample applications, see the *Installation Notes*.

Overview of the Business Problem

This chapter focuses on solving the business problems that arise as the sample company, The Beverage Company (TBC), expands its business outside the United States. This chapter builds on the business scenario introduced in Chapter 5, Designing a Single-Server Application, as TBC adds the following markets:

- Three locations in Canada: Toronto, Vancouver, and Montreal
- Four locations in Europe: the UK, Germany, France, and Spain

In addition, TBC adds a new member, US, which is a consolidation of data from the United States regions: East, West, South, and Central.

Data for each TBC market location is captured in local currency. Dollar values are derived by applying exchange rates to local values. These values must then be converted to a common currency (in this case, US$).

TBC needs to analyze actual data in two ways:

- Actuals are converted at actual exchange rates.
- Actuals are converted at budget exchange rates to analyze variances due to exchange rates.

When all actuals have been processed, budget data is converted with budget exchange rates.

Overview of the Currency Application's Structure

Currency conversion applications are defined by linking a currency database to a main application database, as illustrated in Figure 42-1. The *Main Database* contains the full database outline and associated data values; the *Currency Database* contains only exchange rates and other currency-related information. In the example provided in this chapter, TBC's Main Database is INTERNTL; its Currency Database is XCHGRATE. On your server, these databases are in the Sample application.

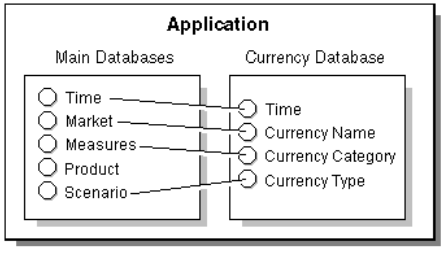

Figure 42-1, Currency Application Consists of Two Databases

Contents of the Main Database

The Main Database can be from 3 to *n* dimensions in size. At a minimum, the database must contain the following dimensions:

- **A dimension tagged as Time.** In the TBC example this dimension is Year.
- **A dimension tagged as Accounts.** Within an Accounts dimension, individual members may be tagged with different categories of exchange rates. This allows you to apply different exchange rates to various accounts or measures.

  For example, P&L accounts may use exchange rates that differ from those used with balance sheet accounts. In addition, some accounts may not require conversion. For example, members such as Units, Headcount, and Margin% require no conversion.
- **A market-related dimension tagged as Country.** In a market dimension, each member has an associated currency name. In the TBC example, the currency name US$ is defined for the Market dimension tagged Country, because all local currencies for the new markets must be converted to US$, the company's common currency.

  Because many members can have the same currency name, the number of currency names is typically less than the total number of members in the dimension. As shown in Table 42-1, TBC uses only six currency names for its 15 Market dimension members. The children of the member Europe all use a different currency and, therefore, must be assigned individual currency names. However, in the case of the Country dimension and the US member and its four cities, all locations use the same currency. The same is true of the Canada member and its three cities. When the children of a given member share a single currency, you only need to define a currency name for the parent member.

Table 42-1, TBC Currency Names

Dimension/Member	Currency Name
Market - Country	US$
US	
East	
West	
South	
Central	
Canada	CN$
Toronto	
Vancouver	
Montreal	
Europe	
UK	Pound
Germany	Mark
France	Franc
Spain	Peseta

A typical Main Database also contains an *optional fourth dimension*: **A dimension defined with a currency partition.** Databases are usually partitioned in a Scenario dimension. Creating a currency partition allows Essbase to internally track currency relationships and previously-converted values of the main and exchange rates databases.

Note: A currency conversion partition applies only to the Currency Conversion option. It is not related to the Hyperion Essbase Partitioning option that enables data to be shared between databases using a replicated, linked, or transparent partition.

The *Spreadsheet Add-in User's Guide* provides examples of ad hoc currency reporting capabilities. Report scripts also let you define reports that convert data when the report is displayed, as discussed under "Converting Data to a Different Currency in Reports" on page 42-26.

Contents of the Currency Database

The Currency Database requires the following three dimensions:

- **A Currency Time dimension.** Exchange rates are usually defined for each time period. Each member of the time dimension in the Main Database must be defined in the Currency Database. Values by time period in the Main Database are usually converted to the exchange rates of their respective time period from the Currency Database (although you can convert data values against any period's exchange rate).
- **Currency Name dimension.** This dimension contains the names of currencies relevant to the markets (or countries) defined in the Main Database. Each currency name defined in the Main Database must also exist in the Currency Database. The currency names define the country-to-exchange rate mapping when conversion occurs.
- **Currency Category dimension.** This dimension contains all the account categories of the Main Database. The category names define the exchange rate mapping when a conversion occurs. In the TBC example, the account categories included are P&L (Profit & Loss) and B/S (Balance Sheet).

A currency database typically includes an *optional fourth dimension*: **A Currency Type dimension** which contains members that identify various currency scenario types**.** Typically an application has different exchange rates for scenarios, such as actual, budget, and forecast. Members of the currency-type dimension are not directly mapped to members of the Main Database. Therefore, member names in this dimension are not required to match member names of the Main Database.

Conversion Methodologies

Different currency applications have different conversion requirements. Essbase supports two currency database types, each with a different conversion method:

- **A database containing only converted data values.** Some applications require only converted values to be stored in the Main Database. Local values are entered and the conversion operation overwrites the local values with a common currency. This method assumes that there is no requirement for reporting or analyzing local currencies.

  Because this operation overwrites data, you must load local values and recalculate the data each time you perform a conversion. This method is useful only when you want to perform a single (not ongoing) conversion.

- **A database containing both local and converted data.** Most applications require data to be stored in both local and master values. This method permits reporting and analyzing local data. In addition, data modifications and recalculations are easier to control. To use this method, you must define a currency partition (see "Modify the Scenario Dimension" on page 42-13). TBC uses this method for defining their currency application.

Either of these two methods may also require a currency conversion to be applied at report time. This allows you to analyze various exchange rate scenarios without actually storing data in the database. The currency conversion module allows you to perform ad hoc conversions with the Spreadsheet Add-in, which is discussed in the *Spreadsheet Add-in User's Guide*, or with a report script, as discussed under "Converting Data to a Different Currency in Reports" on page 42-26.

Steps for Creating a Currency Conversion Application

To accommodate the new markets TBC has added and to provide Essbase with the required currency-related information, TBC has modified their existing Essbase database outline to create a currency conversion application. You can use the TBC model to create your own currency conversion application, revising the steps as needed to fit your specific requirements.

The TBC Currency Conversion application was created using the following steps:

1. "Creating the Main Database Outline" on page 42-7
    - "Open the Existing Database Outline" on page 42-7
    - "Modify the Measures Dimension" on page 42-8
    - "Modify the Market Dimension" on page 42-10
    - "Modify the Scenario Dimension" on page 42-13
    - "Save the Main Database Outline Changes" on page 42-16
2. "Creating the Currency Database Outline" on page 42-16
    - "Generate the Currency Database Outline" on page 42-17
    - "Review the Contents of the Currency Database Outline" on page 42-19
    - "Add New Members to the CurType Dimension" on page 42-20
    - "Save the Currency Database Outline Changes" on page 42-20
3. "Linking the Main and Currency Databases" on page 42-21

Creating the Main Database Outline

To create the Main Database outline (INTERNTL), you need to open your existing Essbase database outline, make changes to its contents, and then save the outline for use in your Currency Conversion application.

Open the Existing Database Outline

Begin creating the Main Database outline by opening TBC's existing outline, as follows:

1. Start Hyperion Essbase Application Manager and select Server | Connect.
2. Choose the Essbase server that contains the Sample Application.
3. Select Sample from the **Applications** list box and open the database outline for the database INTERNTL, as shown in Figure 42-2.

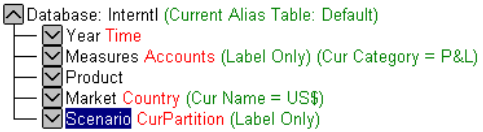

Figure 42-2, Sample Database INTERNTL Essbase Outline

TBC has modified the Measures, Market, and Scenario dimensions. The Year and Product dimensions require no changes because they have no information specifically related to currencies.

Note: The Year dimension *must* be tagged as the Time dimension.

Modify the Measures Dimension

To create their Currency Conversion Main Database, TBC modified the Measures dimension using the following steps. You can follow along and revise the procedures as necessary to create your own currency application.

1. Tag Measures as an Accounts dimension.

   You must have a dimension tagged as Accounts in your Main Database. To meet varying conversion and exchange rate requirements, individual members within the Accounts dimension can then be tagged with different categories of exchange rates.

   All descendants of the Profit member, for example, use a special currency rate that applies to Profit and Loss accounts. Therefore, they are defined with a currency category of P&L.

2. Define the Profit member with a currency category name, as follows:

   a. Select the Profit member and click the **Data Dictionary** button,  , on the toolbar. The **Member Specification** dialog box appears.

   b. Select the **Category** option button from the **Currency Conversion** group, and enter the category name of P&L (for Profit and Loss).

Note: Each descendant of a member inherits the currency category tag of its ancestor. A member or sub-branch of members can also have its own category defined.

Because the Measures dimension shows data for Profit, it must also have the P&L currency category name applied.

3. Define the Measures dimension with a currency category name, as follows:

   a. Select **Prev** to move to the previous member in the dimension and select **Category** from the **Currency Conversion** group. The name P&L appears from the previous entry.

   b. Click OK to close the dialog box.

   All descendants of the Inventory member use a special rate that applies to balance sheet accounts. Define them with a currency category of B/S (for Balance Sheet).

4. Define the Inventory member with a currency category name, as follows:

   a. Select the Inventory member and click the **Data Dictionary** button,  , on the toolbar. The **Member Specification** dialog box appears.

   b. Select **Category** from the **Currency Conversion** group, and enter the category name of B/S.

   c. Click OK to close the dialog box.

   Ratio members do not require a currency conversion, because the local and converted values yield the same result.

5. Define the Ratio member as not requiring a conversion:

   a. Select the Ratios member, and press the **Data Dictionary** button,  , on the toolbar. The **Member Specification** dialog box appears.

   b. Select **No Conversion** from the **Currency Conversion** group.

Note: The children of a member defined with No Conversion do not inherit the No Conversion definition. Each member must be defined individually.

6. Define the children of the Ratio member (Margin % and Profit %) as not requiring a conversion:

   a. Select **Next** to move down the member list to Margin %.

   b. Define Margin % with the No Conversion option.

   c. Select **Next** to move down the member list to Profit %.

   d. Define Profit % with the No Conversion option.

   e. Click OK to close the dialog box.

Figure 42-3 shows the resulting Measures dimension outline after making the modifications necessary for the TBC Currency Conversion application.

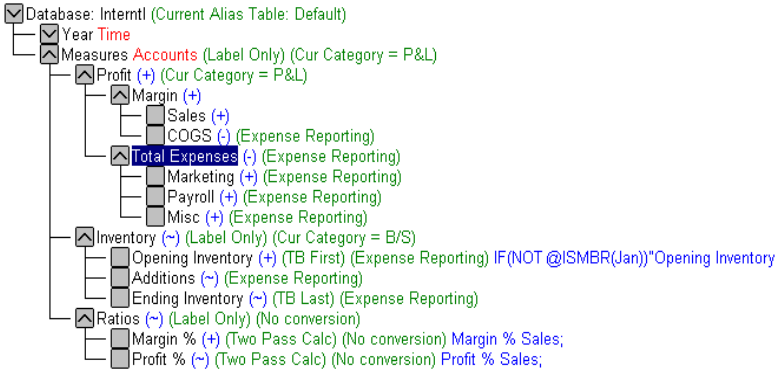

Figure 42-3, TBC Measures Main Database Outline

Modify the Market Dimension

TBC needed to make the following modifications to the Market dimension to accommodate their newly added markets and create their Currency Conversion application Main Database. Procedures for making these changes follow the bullet list.

- Tag the Market dimension as Country. Remember, having a Country tag enables you to define the common currency to be used for converting local currency values. In the TBC example, local currency values for their new markets are converted to US$, the currency name defined for the Country dimension.
- Add the US member, which consolidates data from existing United States region members East, West, South, and Central.
- Add the Canada member and its children, Toronto, Vancouver, and Montreal.
- Add the Europe member and its children, UK, Germany, France, and Spain.
- Define currency names for each of the newly added dimensions/members, as necessary. Remember, when the children of a given member share a single currency, you only need to define a currency name for the parent member.

Modify the Market dimension as follows:

1. Tag Market as a Country dimension by selecting the Market dimension and clicking **Country**,  , on the toolbar.

   The existing members (East, West, South, and Central) belong in TBC's newly added US market. To make this outline change, TBC added the US member with the existing members as descendants.

2. Add the new US member:

   a. Select the Market member and click the **Add Child** button,  .

   b. Type the name US in the **Member Edit** box and press Enter twice.

3. Make existing members East, West, South, and Central, children of US by clicking the **Add Child** button for each member.

4. Add Canada and its respective cities to the Market dimension:

   a. Select the US member and click the **Add Sibling** button,  .

   b. Type the name Canada in the edit box and press Enter twice.

   c. Select the Canada member and click the **Add Child** button.

   d. Add the Toronto, Vancouver, and Montreal members and press Enter twice.

5. Add Europe and its respective countries to the Market dimension:

   a. Select the Canada member and press the **Add Sibling** button.

   b. Type the name Europe in the edit box and press Enter twice. Select the Europe member and click the **Add Child** button.

   c. Add the UK, Germany, France, and Spain members and press Enter twice.

You must now define currency names for each appropriate member. Because TBC required the database to be converted to US dollars, they defined the Market dimension with the US$ currency name.

6. Define currency names for the Market dimension and its members, as appropriate:

   a. Select the Market dimension and click the **Data Dictionary** button,  . The **Member Specification** dialog box appears.

   b. Enter US$ in the **Currency Name** type in box.

   c. Click OK to close the dialog box.

   All descendants of the Market dimension inherit the US$ currency name (unless a child branch has already been defined with another currency). Because members in the US branch use US$ as their local currency, there is no need to define a currency name for these members.

   Like the Market dimension, all descendants of Canada also use the same currency (CN$). You need to define a currency name for the Canada member only and its children inherit the CN$ currency name.

7. Define a currency name for the Canada member:

   a. Select Canada and click the **Data Dictionary** button,  . The **Member Specification** dialog box appears.

   b. Type CN$ in the **Currency Name** box.

   c. Click OK to close the dialog box.

   The children of Europe all use different local currencies. Unlike the Market dimension and the Canada member, you must define each Europe member with an individual currency name.

8. Define individual currency names for children of the Europe member:

   a. Select the UK member name and click the **Data Dictionary** button,  . The **Member Specification** dialog box appears.

   b. Type `Pound` in the **Currency Name** box.

   c. Click **Next** to define the currency name for next country (Germany).

   d. Type `Mark` in the **Currency Name** box.

   e. Click **Next** to define the currency name for next country (France)

   f. Type `Franc` in the **Currency Name** box.

   g. Click **Next** to define the currency name for next country (Spain).

   h. Type `Peseta` in the **Currency Name** box.

   i. Click OK to close the dialog box.

Figure 42-4 shows the resulting Market dimension outline after making the modifications necessary for the TBC Currency Conversion application.

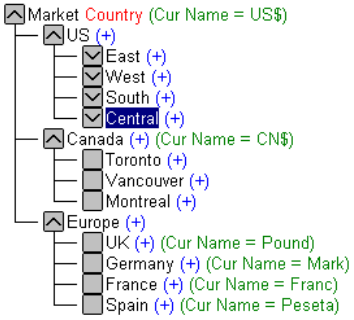

Figure 42-4, TBC Market Dimension Database Outline

Modify the Scenario Dimension

In the TBC Main Database (INTERNTL), the Scenario dimension contains members for both local and converted values. TBC needed to make the following modifications to the Scenario dimension to create their Currency Conversion Application Main Database:

- Create a currency partition. This enabled TBC to store both converted and non-converted values in their Main Database (INTERNTL).
- Define the converted members under a Dollars parent. This enabled TBC to track actuals converted at actual exchange rates.
- Add a new member called Actual @ Bud XChg. This enabled TBC to track actuals converted at separate budget exchange rates to analyze variances due to exchange rates of their newly added markets.
- Define members that can store data in local currency values (Actual and Budget).
- Apply the Exclude member from consolidation tag to the Local, Bud member, because it doesn't store data.
- Tag the Local member as a label, because it doesn't store data.
- Tag the Scenario dimension as a label, because it also does not store data.

Modify the Scenario dimension as follows:

1. Set a Currency Partition tag on the Scenario dimension:
   a. Select the Scenario dimension name and click the **CurPartition** button,  .
   b. Define the converted members under a parent called Dollars.
   c. Add the member Dollars as a child of Scenario. Select the dimension Scenario and press the **Add Child** button.
   d. Type Dollars in the member edit box and press Enter twice.
   e. Move the Actual, Budget, Variance, and Variance % members as children of Dollars.

   The Dollars member does not store data and must, therefore, be tagged as a label.
2. Tag the Dollars member as a label by selecting it and clicking the **Label** button,  .

   TBC also wants to track actuals using the budget exchange rate. The Scenario dimension requires a new member to handle this data.
3. Add a new member, Actual @ Bud XChg, to the Scenario dimension Actual member:
   a. Select the Actual member and click the **Add Sibling** button.
   b. Enter the member `Actual @ Bud XChg` and press Enter twice.

   The newly added Actual @ Bud XChg member does not consolidate into the Dollars member and must, therefore, be tagged appropriately.
4. Apply the **Exclude member from consolidation** tag to the Actual @ Bud XChg member by pressing the **No Consolidation** button,  .
5. Define members that can store the data in local currency values:
   a. Add the member Local as a sibling of Dollars. Select the member name Dollars and press the **Add Child** button.
   b. Enter the name `Local` in the **Member Edit** box and press Enter twice.

   Data only needs to be captured in local currency for Actual and Budget. These are the only children to define for Local.

6. Define Local member children to capture local data:

   a. Select the Local member and click the **Add Child** button.

   b. Enter the member names `Act` and `Bud` and press Enter twice.

   Values for the Act member consolidate to Local, but values for the Bud member do not.

7. Apply the **Exclude member from consolidation** tag to the Bud member by pressing the **No Consolidation** button,  .

8. Tag the Local member (which does not store data) as a Label by selecting it and clicking the **Label** button.

9. Tag the Scenario dimension (which does not contain any consolidated results) as a Label by selecting it and clicking the **Label** button.

   Figure 42-5 shows the resulting Scenario dimension outline after making the modifications necessary for the TBC Currency Conversion application.

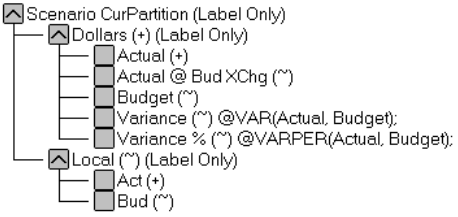

Figure 42-5, Scenario Dimension Database Outline

Save the Main Database Outline Changes

Save the outline changes by clicking the **Save** button. If your database contains data values, Essbase restructures the database to reflect changes to the outline. When you click **Save**, the **Restructure Database** dialog box shown in Figure 42-6 appears.

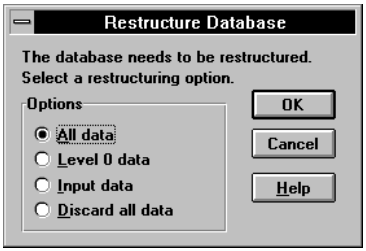

Figure 42-6, Restructure Database Dialog Box

When an existing outline is updated, you have several choices that let you restructure data values. Because no data has been loaded into the model, click OK to continue the outline update.

Creating the Currency Database Outline

Once you have verified and saved the Main Database outline, you can generate the Currency outline. The Currency outline contains dimensions, members, currency names, and currency categories previously defined in your Main Database outline. It is basically structured and ready to use after being generated but may require additions to make it complete.

Generate the Currency Database Outline

Follow these steps to generate the Currency outline:

1. Press the **Generate Currency Outline** button,  .

   If the Main outline is missing any required dimension tags (such as Time, Accounts, or Country), Essbase displays an error box like the one shown in Figure 42-7.

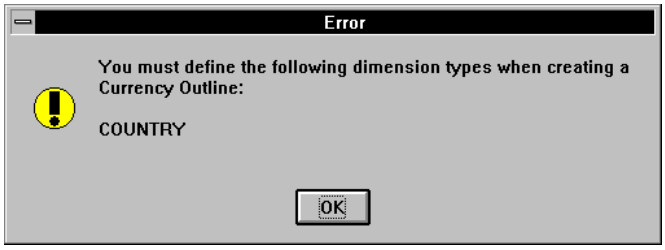

Figure 42-7, Error Dialog Box Displayed When No Country Dimension Is Defined

2. If the **Error** dialog box is displayed after you attempt to generate the Currency outline, do the following:

   a. Note the errors and click OK.

   b. Tag the dimensions as required.

   c. Attempt to generate the Currency outline again. If all tags are correctly defined, the dialog box shown in Figure 42-8 appears.

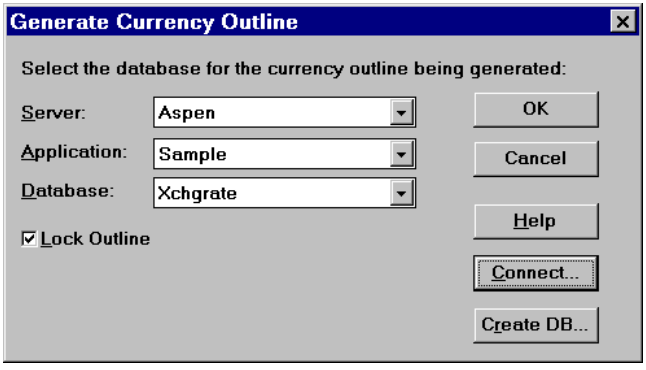

Figure 42-8, Generate Currency Outline Dialog Box

The **Generate Currency Outline** dialog box lets you define the database for which the outline is generated. The dialog box contains the following controls:

- The **Server** list box, which lets you select a Essbase Server
- The **Application** list box, which lets you select an application
- The **Database** list box, which lets you select a database (only Currency Databases appear in the list)
- The **Connect** button, which lets you log in to a different server if necessary
- The **Create DB** button, which lets you create a new Currency Database

If you need to log in to a different server, click the **Connect** button. The dialog box shown in Figure 42-9 appears. If you do not need to log in to a different server, go to Step 5.

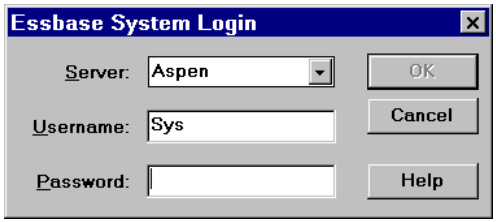

Figure 42-9, Essbase System Login Dialog Box

3. Select the correct server, enter your user name and password, and click OK to close the dialog box and return to the **Generate Currency Outline** dialog box.
4. Click the **Create DB** button. The dialog box shown in Figure 42-10 appears.

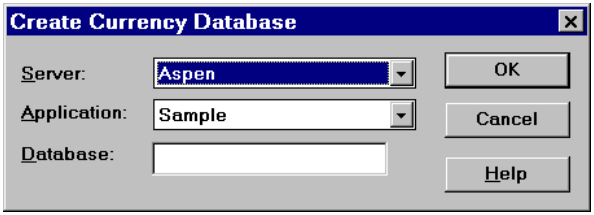

Figure 42-10, Create Currency Database Dialog Box

5. Type the database name in the **Database** text box. In the TBC example, we use the database name XCHGRATE.
6. Click OK to create the database.
7. When the **Generate Currency Outline** dialog box reappears, click OK to generate the outline.

   The outline for the Currency Database XCHGRATE appears in the Outline Editor as shown in Figure 42-11.

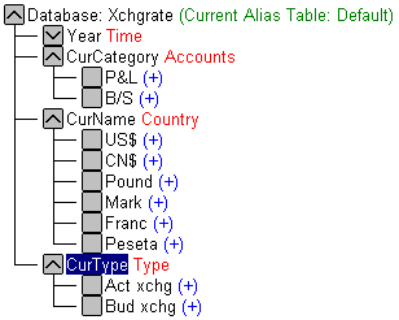

Figure 42-11, Currency Database Outline

Review the Contents of the Currency Database Outline

After generating your Currency Database outline, review its contents and note the following items that are automatically created based on the contents of the Main Database that you defined previously:

- The Year dimension is replicated from the Main Database. This allows you to define exchange rates for each time period.
- The currency category names that were defined on some members of the Measures dimension in the Main Database become members of the CurCategory dimension.
- The currency names from the Market dimension become members of the CurName dimension.
- The CurType dimension, which contains no members, is also created. This dimension is not directly mapped to the Main Database.

Add New Members to the CurType Dimension

After you have generated the Currency Database, you can add members to any dimension. Because the TBC Currency Database (XCHGRATE) contained different exchange rates for actual and budget, they added two new members to the CurType dimension to apply to different scenarios in the Main Database:

1. Add the Act xchg member to the CurType dimension.
2. Add the Bud xchg member to the CurType dimension.

   Exchange rates from these two members are applied to different scenarios from the Main Database, as follows:

   - Actual (from the Main Database) is converted with a CurType member called Act xchg.
   - Actual @ Bud Xchg (from the Main Database) is converted with a CurType member called Bud xchg.
   - Budget (from the Main Database) is converted with a CurType member called Bud xchg.

Note: For details about the currency conversion calculation process, see "Calculating Currency Conversions" on page 42-23.

Save the Currency Database Outline Changes

When you have reviewed the newly generated Currency Database outline and have made any necessary additions, you need to save your changes.

1. Save changes made to the Currency Database outline by pressing the **Save** button. The dialog box shown in Figure 42-12 appears.

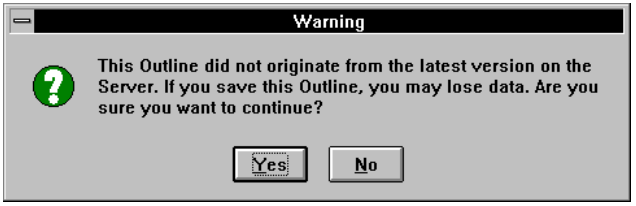

Figure 42-12, Using the Latest Version on the Server Warning

Because the newly-generated outline contains all new members, you are given a choice of continuing or canceling the operation.

2. Click Yes to continue. The **Restructure Database** dialog box shown in Figure 42-13 appears.

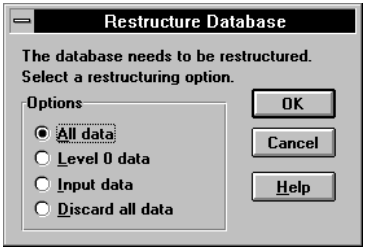

Figure 42-13, Restructure Database Dialog Box

When an existing outline is updated, you have several choices that let you restructure data values.

3. Because no data has been loaded into the model, click OK to continue the outline update.
4. Close the Outline Editor for the Main and Currency Databases.

Linking the Main and Currency Databases

To perform a currency conversion calculation, Essbase must recognize a link between the Main and Currency databases. Generating a currency outline does not automatically link a Main Database with a Currency Database.

To link a Currency Database to the Main Database:

1. Select the Main Database from the application's database list.
2. Choose Database | Settings. The **General** page of the **Database Settings** dialog box shown in Figure 42-14 appears.

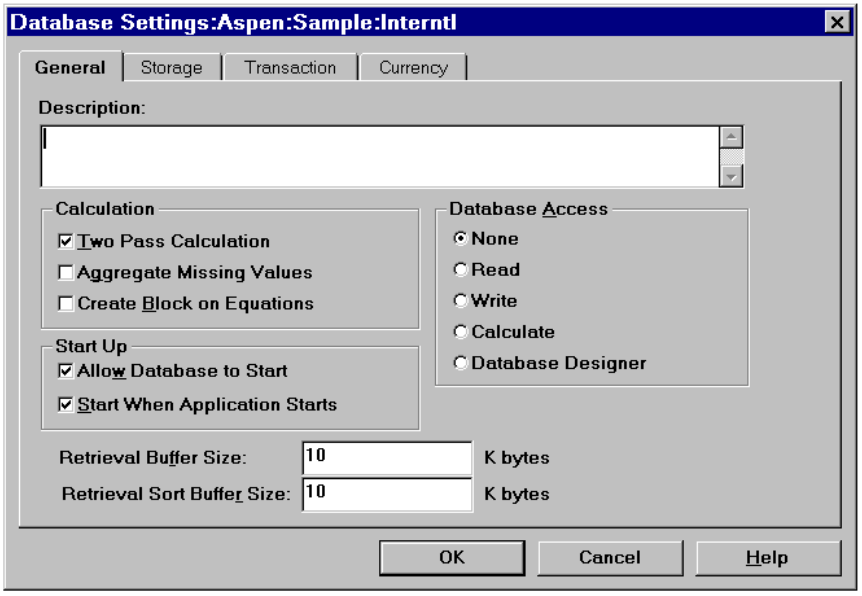

Figure 42-14, Database Settings Dialog Box General Page

3. Select the **Currency** page. The **Currency** page shown in Figure 42-15 appears.

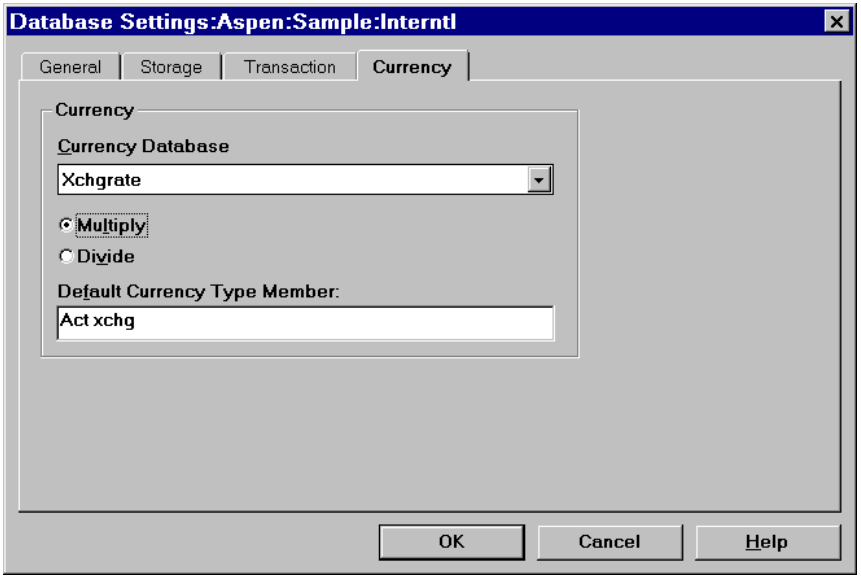

Figure 42-15, Database Settings Dialog Box Currency Page

4. Select a database from the **Currency Database** list box.

   The **Multiply** and **Divide** options are enabled. Use these option buttons to define the conversion calculation method to be used. The **Multiply** option button multiplies each local data value by the exchange rate. The **Divide** option button divides each local data value by the exchange rate.

5. Select the **Multiply** option.

   The **Default Currency Type Member** text box lets you define a currency type member to use as a default for currency conversion calculations.

6. Type `Act xchg` to assign this as the default.

7. Click OK twice to close the dialog box and save these settings.

After the databases are defined and linked, the next step is to load data into the Main Database and exchange rates into the Currency Database. After values have been entered, a currency conversion is calculated. This is accomplished by running a calc script (see "Calculating Currency Conversions" on page 42-23).

Calculating Currency Conversions

You convert data values from a local currency to a common, converted currency using the CCONV *currExchMbr* command in a calc script. For example, you might convert data from a variety of European currencies into US$.

Note: You cannot use the CCONV command to convert data in a transparent partition.

You convert the data values back to the original, local currencies using the CCONV TOLOCALRATE *CurType* command.

You can convert all or part of your main database using the rates defined in your Currency database. You can overwrite the local values with the converted values, or you can keep both the local and converted values in your main database, depending on your tracking and reporting needs.

Overwriting Local Values with Converted Values

If you want to overwrite the local values, you do *not* need to create a CURPARTITION dimension in your main database. Use the CCONV command in a calc script to convert all the data in your database.

The following calc script converts the values in the database to US$.

```
CCONV US$;
CALC ALL;
```

If required, you can specify a currency name that contains the required exchange rate. The following calc script converts the values in the database to US$, using the exchange rate for Jan as defined in the Currency database.

```
CCONV Jan->US$;
CALC ALL;
```

The CALC ALL command is required in the examples shown, because the CCONV command only converts currencies. It does not consolidate or calculate members in your database.

The following calc script converts the values back to their original values in their local currencies using the "Act xchg" rate:

```
CCONV TOLOCALRATE "Act xchg";
CALC ALL;
```

For more information on the CCONV command, see the online *Technical Reference* in your DOCS directory.

Note: You cannot use the FIX command unless you are using a CURPARTITION dimension and the CCTRACK setting is TRUE in the ESSBASE.CFG file.

Keeping Local and Converted Values

You can keep both local and converted values in your database. In your main database you need to define the members that store the local and converted values. You do this by creating a CurPartition dimension (see "Modify the Scenario Dimension" on page 42-13). The CurPartition dimension has two partitions, one for local values and one for converted values.

You then need to create a calc script to complete the following steps:

1. Use the DATACOPY command to copy data from the local to the converted partition.
2. Use the FIX command to calculate only the converted partition and use the CCONV command to convert the data.

Note: When using a CurPartition dimension, you must FIX on a member of this dimension to use the CCONV command.

3. Use the CALC command to recalculate your database.

The following example is based on the Sample INTERNTL database and corresponding Sample XCHGRATE Currency database. Figure 42-16 shows the Currency Partition from the Sample INTERNTL database.

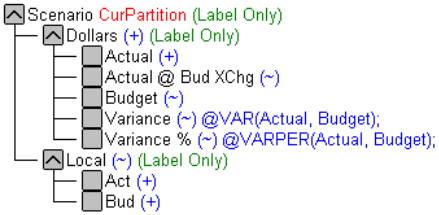

Figure 42-16, Calculating Local and Converted Currency Conversions

The following calc script performs three currency conversions for Actual, Budget and Actual @ Bud Xchg data values.

```
/* Copy data from the local partition to the master partition (for
converted values) */
DATACOPY Act TO Actual;
DATACOPY Bud TO Budget;

/* Convert the Actual data values using the "Act xchg" rate */

FIX(Actual)
       CCONV "Act xchg"->US$;
ENDFIX

/* Convert the Budget data values using the "Bud xchg" rate */
FIX(Budget)
       CCONV "Bud xchg"->US$;
ENDFIX

/* Convert the "Actual @ Bud XChg" data values using the "Bud xchg"
rate */
FIX("Actual @ Bud XChg")
       CCONV "Bud xchg"->US$;
ENDFIX

/* Recalculate the database */
CALC ALL;
CALC TWOPASS;
```

The following calc script converts the Actual and Budget values back to their original values in their local currencies:

```
FIX(Actual)
CCONV TOLOCALRATE "Act xchg";
ENDFIX
FIX(Budget)
CCONV TOLOCALRATE "Bud xchg";
ENDFIX
CALC ALL;
```

Note: When you convert currencies using the CCONV command, the resulting data blocks are marked as *dirty* for the purposes of Intelligent Calculation. This means that Essbase recalculates all the converted blocks when you recalculate your database. For more information on Intelligent Calculation, see Chapter 33, Optimizing Your Calculation Using Intelligent Calculation.

Converting Data to a Different Currency in Reports

You can calculate currency conversions in report scripts, using the CURRENCY *target currency* command to set the output currency and currency type. For the syntax and definitions of Report Writer commands, see the online *Technical Reference* in your DOCS directory.

Note: Essbase cannot perform "on the fly" currency conversions across transparent databases. If you have two transparent partition databases that are calculated using different conversions, you cannot calculate currency conversions in reports.

The following Sample report contains first quarter Budget Sales for colas, using the January exchange rate for the Peseta currency.

```
                    Illinois Sales Budget

                       Jan           Feb       Mar
                       ======== ======== ========
 100-10                3             3         3
 100-20                2             2         2
 100-30                #Missing #Missing #Missing
 100                   5             5         5
             Currency: Jan->Peseta->Act xchg

             Currency: Jan->Peseta->Act xchg
                       Illinois Sales Budget
                       Jan           Feb       Mar
                       ======== ======== ========
 100-10                3             3         3
 100-20                2             2         2
 100-30                #Missing #Missing #Missing
 100                   5             5         5
```

Use the following script to create the Sample currency conversion report:

```
<Page (Market, Measures, Scenario)
{SupCurHeading}
Illinois Sales Budget
       <Column (Year)
       <child Qtr1
<Currency "Jan->Peseta->Act xchg"
<Ichild Colas
!
{CurHead}
Illinois Sales Budget
       <Column (Year)
       <child Qtr1
!
```

Effects of CCTRACK Parameter on Conversion Calculations

The CCTRACK setting in your ESSBASE.CFG file controls whether exchange rates are tracked as currency conversions are calculated.

When CCTRACK is TRUE, Essbase tracks exchange rates that are applied to data as conversions are calculated, allowing conversion to occur at report time through the Spreadsheet Add-in or the Essbase Report Writer.

Setting CCTRACK to FALSE turns off the tracking system and has the following results:

- A CCONV command assumes the data to be converted is always in local currency.
- Currency report options assume that the data to be converted is always in local currency. This means that even if the data has already been converted in the database, it is reconverted at report time and can yield inaccurate results.
- The restrictions on operations such as FIX or DATACOPY are removed for conversion operations. For example, when you have a currency partition and CCTRACK is set to TRUE, you can't use the CCONV command outside of a FIX. Conversely, when you do not have a currency partition and CCTRACK is set to TRUE, Essbase does not allow you to calculate without a FIX. Setting CCTRACK to FALSE eliminates these restrictions.

For more information, refer to online *Technical Reference* in your DOCS directory.

Part IX Maintaining Hyperion Essbase Applications

Part IX describes how to maintain existing Hyperion Essbase OLAP Server applications, including how to use ESSCMD; run Essbase, applications, and databases; monitor performance; work with Essbase files in cross-platform environments; and back up and recover databases. Part IX contains the following chapters:

- Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, describes how to use ESSCMD's interactive and batch processing modes. It contains detailed script and batch files.
- Chapter 44, Running Essbase, Applications, and Databases, describes how to use the Agent to start and stop Essbase servers, use the server console, start and stop Essbase databases, view a list of users, view a list of available ports, and describes how the Agent handles client/server communications.
- Chapter 45, Monitoring Performance Using Diagnostics, describes how to use diagnostic tools to analyze problems, solve problems, check Essbase configuration, and analyze Essbase performance.
- Chapter 46, Working with Essbase Files and Cross-Platform Environments, introduces you to the kinds of files that Essbase uses, describes operations you can perform on Essbase files, and discusses some of the issues you face when working with Essbase files on different platforms.
- Chapter 47, Backing Up Data and Recovering Databases, describes when Essbase performs commit operations, how to recover a database, how to back up a database, and other ways Essbase protects your data.

Chapter 43 Performing Interactive and Batch Operations Using ESSCMD

Your Hyperion Essbase OLAP Server includes ESSCMD, a command-line interface that performs operations interactively or through a batch or script file.

This chapter describes how to use ESSCMD's interactive and batch processing modes, and provides sample script and batch files. It includes the following topics:

For the syntax and usage of each ESSCMD command, see the online *Technical Reference* in your `DOCS` directory.

ESSCMD Basics

With ESSCMD, you can execute server operations at the command line, in either interactive or batch mode.

Interactive mode lets you interactively enter commands at the ESSCMD command line and receive responses. Interactive mode is convenient for short operations that require few commands, checking for information on the fly, and error checking.

Batch processing mode lets you automate your routine server maintenance and diagnostic tasks. You can write a script or batch file and run it from the command line. Batch processing mode is convenient if you frequently use a particular series of commands, or if your task requires many commands.

Syntax Guidelines

In general, you use the same syntax for entering ESSCMD commands as you do for other Essbase commands. However, there are differences between ESSCMD's interactive and batch processing modes in the requirements for quotation marks and the semicolon statement terminator. Use the guidelines in this section when creating script or batch files.

Quotation Marks

Quotation marks ("") enclose character parameters and responses to commands.

- In interactive ESSCMD, using quotes is optional. Be sure to use quotes when a parameter has an embedded space. For example:

  ```
  CALC "Calc All;"
  ```

- In an ESSCMD script file, always enclose all character parameters and responses to commands in quotes. For example:

  ```
  LOGIN "Local" "TomT" "Password";
  ```

- You do not have to enclose numeric parameters and responses in quotes.
- You cannot place quotes within quotes.

Semicolon Statement Terminator

The ; (semicolon) statement terminator signals the end of a command. For example:

```
SELECT "SAMPLE" "BASIC";
```

- In interactive ESSCMD, pressing the Enter key signals ESSCMD that the command is complete. The statement terminator is optional.
- In an ESSCMD script file, you should use the terminator, even though it is optional, if a command has many parameters. This is especially important in order to signal the end of the parameter list if some of the parameters are optional.

  If you omit some optional parameters and do not use a semicolon to end the list, ESSCMD looks for the remaining values in the next command in the file, leading to unpredictable results.

The SETAPPSTATE and SETDBSTATE commands, defined in the online *Technical Reference* in your `DOCS` directory, are examples of commands which you should terminate with a semicolon to prevent any confusion in processing.

Note: All syntax examples in this chapter use quotation marks and semicolon terminators.

Running ESSCMD on Different Operating System Platforms

ESSCMD operates independently of any other Essbase client interface, including the Application Manager, Spreadsheet Add-in, or custom-built application programs.

ESSCMD is available on the following platforms:

- Windows NT server
- HP-UX server
- AIX server
- Solaris server
- Windows 95 and Windows 98 server

Canceling ESSCMD Operations

When running ESSCMD, you can cancel an asynchronous operation, such as a calculation, export, or restructure operation, by pressing and holding the Esc key until ESSCMD responds.

Referencing Files

Some commands require that you precede object or file names with a numeric parameter, from 1 to 4, that tells Essbase where to look for the object or file. The parameter directs ESSCMD to look for files in other applications, databases, or systems.

The following table lists each value for the numeric parameter (*numeric*), the file location to which it applies, and the information that ESSCMD requests when you use each parameter setting. *appName* is the application name and *dbName* is the database name.

numeric	File	Essbase prompts user for:
1	Local or client-based file	Windows NT: Files in the `\ESSBASE\CLIENT\appName\dbName` directory UNIX: Files in the `essbase/client/appName/dbName` directory
2	Remote or server-based file	Windows NT: Files in the `\ESSBASE\APP\appName\dbName` directory UNIX: Files in the `essbase/app/appName/dbName` directory
3	File	Fully-qualified path to the file, unless file is in the current ESSCMD directory
4	SQL table	Full network and database information for the SQL table

For example, the LOADDATA command can load a data file that resides on the client or the Essbase server. The command requires the numeric parameter to tell Essbase where to look for the data file. This example causes ESSCMD to prompt for the fully-qualified path name of the file to load:

```
LOADDATA 3
```

File extensions are usually optional in both interactive and batch processing modes, except when using commands that require a numeric parameter that indicates the location of files:

- If you use file option 3 (File), you must enter the file extension in both interactive and batch processing modes.
- If the object is in the directory from which you started ESSCMD, you do not need to enter a path.

Multi-User Considerations

Because ESSCMD supports multiple login instances on an Essbase server, you can access more than one database in a single session. Even when you log in to multiple databases, you use only one port on your server license.

Case-Sensitivity

Case-sensitivity varies by operating system:

- Windows NT is not case-sensitive. You can enter ESSCMD commands and file-names in uppercase or lowercase letters, or in any combination of the two.
- UNIX is case-sensitive. You must enter file names in the correct case or UNIX does not recognize them. However, you can enter ESSCMD command names and parameters in uppercase or lowercase.

  For more information about entering UNIX file names, see the *Installation Notes.*

Getting Help

- To display a list of all currently available ESSCMD commands, enter:

  ```
  HELP ?
  ```

- To see a description of any ESSCMD command, see the online *Technical Reference* in your `DOCS` directory.

Quick Start to ESSCMD

Before you start ESSCMD, make sure that the following items are properly installed and running:

- Essbase server
- Communications protocol (Named Pipes or TCP/IP) on the Essbase server

For information on protocols supported by Essbase, see the *Installation Notes*.

Starting and Quitting ESSCMD

The Essbase server installation places the `ESSCMD.EXE` file (`ESSCMD` on UNIX platforms) in the `BIN` directory of your application server.

Once you start the application, a command prompt like this one appears:

```
:::[#]->
```

where # is the value of the active login instance. Each subsequent, successful login increments this value by one. When you start ESSCMD, the instance number is zero (0).

To start ESSCMD:

Enter `ESSCMD` at the operating system command prompt. ESSCMD runs within the operating system command line window.

To quit ESSCMD:

Enter `EXIT` at the prompt and press Enter.

ESSCMD disconnects from the application server and returns you to the operating system command prompt.

Using Interactive Mode

In interactive mode, you enter commands and respond to prompts. This is useful when you are performing simple tasks that require few commands. If you are performing more complex tasks that require many commands, consider creating a script file or batch file; see "Using Script and Batch Files for Batch Processing" on page 43-8 for information.

For syntax conventions when working in interactive mode, see "Syntax Guidelines" on page 43-2.

Logging Into the Server

After starting ESSCMD, you must connect to the Essbase server so that you can enter commands. Follow these steps:

1. At the ESSCMD prompt, log in to the server with the LOGIN command.
2. Enter the Essbase server name. When you connect from the server console, the server name depends on your network setup. For example, the name could be LOCAL.
3. Enter your user name.

4. Enter your password.

   The ESSCMD prompt appears as follows:

   `local:::userName[1]->`

   where *userName* is your login name.

   You can enter any valid ESSCMD command. For a complete listing of commands, type `HELP`.

Note: To load an application into memory and select a database on the Essbase server, use the SELECT command to select a database from an application that resides on the server.

The ESSCMD prompt appears as follows:

`local:appName:dbName:userName[1]->`

where:

- *appName* is the name of the application.
- *dbName* is the name of the database to which you are connected.

Entering Commands

There are two ways to enter commands in interactive mode. Choose either of the following methods to enter commands:

- Type the command and press Enter.

  ESSCMD prompts you for each of the command parameters. For example, the SELECT command has two parameters, as shown in the command syntax:

  `SELECT "appName" "dbName";`

  If you enter only `SELECT` and press Enter, ESSCMD prompts you for the first parameter, the application name (`appName`). After you enter the application name and press Enter, ESSCMD prompts you for the database name (`dbName`).

- Type the commands and all parameters, then press Enter.

  Using SELECT as the example, you would type:

  ```
  SELECT "Sample" "Basic";
  ```

Whichever method you use, the interactive prompt now reflects the application and database names. For example, the following prompt tells you that the Sample application and Basic database are selected:

```
local:Sample:Basic:User[1]->
```

In this case, you can enter other commands without the application or database name parameters that it normally requires.

Canceling Operations

While ESSCMD is running, you can cancel an asynchronous operation, such as a calculation, export, or restructure operation, by pressing and holding the Esc key until ESSCMD responds.

Warning: Do not pause or suspend your system while Essbase is processing a command. Pausing the system may prevent Essbase from correctly completing the command.

Using Script and Batch Files for Batch Processing

If you use a series of commands frequently or you must enter many commands to complete a task, consider automating the task with a script or batch file. These files are useful for batch data loads and complex calculations.

For syntax conventions when working in batch processing mode, see "Syntax Guidelines" on page 43-2.

- A *script file* contains ESSCMD commands. You can run a script file from the operating system command line or from within an operating system batch file, and the script file is processed by ESSCMD. A script file has a `*.SCR` file extension.
- A *batch file* is an operating system file that calls multiple ESSCMD scripts, and can also include operating system commands. You can use a batch file to run multiple sessions of ESSCMD. You can run a batch file on the Essbase server from the operating system prompt; the file is processed by the operating system. On Windows NT, batch files have `*.BAT` file extensions.

Note: On UNIX, a batch or script file is written as a shell script. A shell script usually has the file extension `.sh` (Bourne or Korn shell) or `.csh` (C shell).

When you run a script or batch file, ESSCMD executes the commands in order until it reaches the end of the file.

Existing script or batch files might be affected by changes to some commands. To ensure that your files work in Essbase Version 5, check on your use of changed or deleted commands. See the *Start Here* booklet for information about new and changed commands.

Writing Script Files

ESSCMD script files automate an often-used or lengthy series of commands. Each script file must be a complete ESSCMD session, with login, application and database selection, logout, and termination commands.

To define a script file:

1. Enter ESSCMD commands in any editor that saves data in ASCII text.
2. Save the file with the `.SCR` ESSCMD script file extension.

For example, the following script file, `TEST.SCR`, was created in Notepad:

```
LOGIN "LOCAL" "TOMT" "PASSWORD";
SELECT "SAMPLE" "BASIC";
GETDBSTATE;
EXIT;
```

When run from the operating system command line, this script logs TomT into the Essbase Local server, selects the Sample application and Basic database, gets database statistics, and quits the ESSCMD session.

Running Script Files

To Run Script Files in ESSCMD

Enter the following command at the Essbase server prompt:

`ESSCMD` *`scriptFileName`*`.SCR`

Replace *`scriptFileName`* with the name of the script file. For example:

```
ESSCMD TEST.SCR
```

Handling Command Errors in a Script File

ESSCMD's error-handling features provide error checking and handling for your script files. You can write error-handling commands into your script file to check for errors and, if necessary, branch to an appropriate error-handling response.

After each ESSCMD command is executed, a number is stored in an internal buffer. If the command executes successfully, `0` is returned to the buffer; If the command is unsuccessful, the error number is stored in the buffer; this is called *nonzero status*.

For error checking within an ESSCMD script file, ESSCMD provides the following error-handling commands:

- IFERROR checks the previously executed command for a non-zero return status (failure to execute). If the status is not zero, processing skips all subsequent commands and jumps to a user-specified point in the file, where it resumes. The script file can branch to an error-handling routine or the end of the file.
- RESETSTATUS reverts all saved status values to 0 (zero) in preparation for more status checking.
- GOTO forces unconditional branching to a user-specified point in the file, whether or not an error occurred.

In this `LOAD.SCR` example file, if a command does not execute successfully, ESSCMD branches to the end of the file to avoid completing other operations.

```
LOGIN "local" "User1" "password" "Sample" "Basic";
LOADDATA 2 "calcdat";
IFERROR "Error";
CALC "Calc All;";
IFERROR "Error";
RUNREPT 2 "Myreport";
IFERROR "Error";
[possible other commands]
EXIT;

:Error

EXIT;
```

Note: You can use the OUTPUT command to log errors to a text file.

Sample Script Files

The following script files demonstrate common Essbase batch operations. All samples are based on the Sample Basic database that comes with your Essbase program. The scripts for these examples are available in your `\ESSBASE\APP\SAMPLE\BASIC` directory. On UNIX systems, the examples are available from `/home/essbase/app/Sample/Basic`.

Importing and Calculating a Data Sample File

Suppose you need a file that executes the following actions:

- Logs in to the Essbase server.
- Selects an application and database.
- Prevents other users from logging on and making changes to the database.
- Imports data from a text file.
- Calculates the database.
- Exits ESSCMD.

The following script file does the job:

```
LOGIN "Poplar" "TomT" "Password";
SELECT "Sample" "Basic";
DISABLELOGIN;
IMPORT 2 "ACTUALS" 4 "Y" 2 "ACTUAL" "N";
CALCDEFAULT;
ENABLELOGIN;
EXIT;
```

On Windows NT, this script file, `SAMPLE1.SCR`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory. On UNIX platforms, `SAMPLE1.SCR` is in your `/home/essbase/app/Sample/Basic` directory.

Updating a SQL Script, Importing, and Calculating a Data Sample File

Suppose you need a script file that executes the following actions:

- Logs in to the Essbase server.
- Selects an application and database.
- Prevents other users from logging on and making changes to the database.
- Updates the outline from an SQL data source.
- Imports data from SQL.
- Calculates the database.
- Exits ESSCMD.

The following script file does the job:

```
LOGIN "Poplar" "TomT" "Password";
SELECT "Sample" "Basic";
DISABLELOGIN;
BUILDDIM 2 "PRODRUL" 4 "PRODTBL" 4 "PROD.ERR";
IMPORT 4 "TOMT" "PASSWORD" 2 "ACTUAL" "N";
CALCDEFAULT;
EXIT;
```

On Windows NT, this script file, `SAMPLE2.SCR`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory. On UNIX systems, `SAMPLE2.SCR` is in your `/home/essbase/app/Sample/Basic` directory.

Writing Batch Files

You can write a batch file that runs one or more report scripts, and includes operating system commands. See your operating system instructions to learn the syntax for writing batch files.

For a representative example of a batch file, see "Scheduling Report Printing Example Files" on page 43-14.

Handling Command Errors in Batch Files

For the operating system batch file, you can use ESSCMD command return values to control the flow of scripts that the batch file executes.

An ESSCMD program returns an integer value upon exiting. This value represents the status of the last executed command, usually whether the command succeeded or failed. You can set up your batch file to test for this value, and if the test fails, branch to an error-handling response. This process is similar to creating a script file. For information about handling errors in script files, see "Handling Command Errors in a Script File" on page 43-10.

For example, a batch file could contain three scripts: an ESSCMD batch file that loads data, a calc script that performs calculations on the data, and a report script that reports on the results of the calculation. If the load batch file fails, the calculations and reporting also fail. In this case, it would be best to stop the batch file after the failure of the load file and correct the error that caused the failure before going on. If your batch file tests for the return value of the load process, and this return value indicates failure, the batch file can jump to the end of the file and stop or execute some other error-handling procedure, rather than attempting to calculate data that did not load.

The following example shows a Windows NT operating system batch file and the contents of one of the ESSCMD scripts it runs, `LOAD.SCR`. Because error-checking requirements vary, the syntax in this example may not correspond to that of your operating system. See your operating system documentation for error checking in batch files.

An operating system batch file could contain commands like this:

```
ESSCMD LOAD.SCR
If not %errorlevel%==goto Error
ESSCMD CALC.SCR
If not %errorlevel%==goto Error
ESSCMD REPORT.SCR
If not %errorlevel%==goto Error
Echo All operations completed successfully
EXIT

:Error
Echo There was a problem running the script
```

Scheduling Report Printing Example Files

Suppose you need a file that executes the following actions:

- Logs in to the Essbase server.
- Selects an application and database.
- Assigns reports that output to files for later printing.
- Exits ESSCMD.

The following Windows NT batch file does the job:

```
LOGIN "Poplar" "TomT" "Password";
SELECT "Sample" "Basic";
RUNREPT 2 "REP1" "REP1.OUT";
RUNREPT 2 "REP2" "REP2.OUT";
RUNREPT 2 "REP3" "REP3.OUT";
EXIT;
```

On Windows NT, this script file, `SAMPLE3.BAT`, is available in your `\ESSBASE\APP\SAMPLE\BASIC` directory. On UNIX platforms, `SAMPLE3.BAT` is in your `/home/essbase/app/Sample/Basic` directory.

Chapter 44 Running Essbase, Applications, and Databases

The Hyperion Essbase OLAP Server includes a central organizing sub-system called the Hyperion Essbase Server Agent that performs many important functions. When you load the main Essbase executable file, `ESSBASE.EXE`, the Agent is activated. It is responsible for activities including checking security, establishing connections between the server and its clients, and accessing databases and applications.

The Agent enables you to perform many important administrative functions. This chapter describes the following Agent-related activities:

About the Agent

The Agent allows communication between the Essbase server and its clients. Like a traffic light at a busy intersection, controlling the flow and direction of many cars at once, the Agent controls the flow of activity during Essbase operation. Additionally, the Agent provides ongoing information about all actions performed by users and allows you to monitor the behavior of Essbase.

You have access to the Agent through the *server console*. Access to the Agent is available only on the server machine. After you start Essbase, the server console window appears. The server console is where you view the activities of the Agent and where you enter commands you want the Agent to perform.

Starting and Stopping the Essbase Server

You can start the Essbase server from the server console, from a terminal connected to the server, or as a background process. You can stop the server from the server console, from a terminal connected to the server, or from a batch script.

Starting the Essbase Server in the Foreground

To start the Essbase server, type ESSBASE *password* at the operating system command line, where *password* is the password to access the server. This starts the Essbase Agent (see "Using the Server Console" on page 44-3).

Starting the Essbase Server in the Background

To start the Essbase server in the background on Windows NT, Windows 98, and UNIX platforms, enter:

ESSBASE *password* -b &

You can start Essbase without using the ampersand (&) at the end of the command, but the command prompt is not returned after the server is started.

Stopping the Essbase Server from the Agent

To stop the server and all running applications from the Agent, enter QUIT or EXIT in the Agent window.

Stopping the Essbase Server Remotely

Use the SHUTDOWNSERVER command to shut down the Essbase server, from ESSCMD or from a batch script. You must have Supervisor privilege to use this command. For information on privilege levels, see Chapter 16, Managing Security at Global and User Levels.

Using ESSCMD to Shut Down the Server

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

Using an ESSCMD Batch Script to Shut Down the Server

To shut down a server from an ESSCMD batch script, enter the following command in the batch script:

```
SHUTDOWNSERVER "servername" "username" "password";
```

Where *servername* is the name of the server you wish to shut down; *username* is your user name; and *password* is your password.

Using the Server Console

When Essbase starts on the server, the main process, `ESSBASE.EXE` (`ESSBASE` on UNIX platforms), is the Agent. The Agent can only be accessed from the server machine, which is also called the *console*.

On Windows NT, when the Agent starts, a new icon appears on the desktop, as in the following illustration:

Figure 44-1, Essbase Agent Icon

On Windows 95 and Windows 98, a new icon appears in the Taskbar.

The Agent records its activities, such as user logins, applications starting, and so on, in the Agent log file, `ESSBASE.LOG` (`Essbase.log` on UNIX systems). The log is stored in the `\ESSBASE` directory (`/essbase` on UNIX systems).

You can use the following commands from the Agent:

Command	Function
START *appname*	Starts the specified application.
STOP *appname*	Stops the specified application.
USERS	Displays a list of all users that are connected to the server. The total number of connections is displayed, as well as the number of ports available.
LOGOUTUSER *user*	Disconnects a user from the server and frees a port. This command requires the Essbase system password.
PASSWORD	Changes the system password that is required to start the Essbase server. Changing this password doesn't change the Essbase System Supervisor's password. This command requires the Essbase system password.
VERSION	Displays the server software version number.
HELP	Lists all valid Agent commands and their respective functions.
PORTS	Displays the number of ports that are installed on the server, and how many are in use.
DUMP *filename*	Dumps information from the Essbase security system to a specified file in ASCII format. If you don't supply a path with the file name, the file is saved to the `\ESSBASE\BIN` directory (`/essbase/bin` on UNIX systems). This command requires the Essbase system password.
QUIT/EXIT	Shuts down all open applications and quits Essbase.

You can press Enter when the Agent window is open (Essbase must be started in the foreground) to display a list of all available commands.

Starting and Stopping Applications

When you start an application, Essbase loads the application and any associated databases into memory on the server. This makes data accessible to end users. When you stop an application, Essbase unloads all information and databases from memory on the server.

When you stop an application, transactions may be running on one of the databases. If you issue the STOP command in the Agent, or UNLOADAPP in ESSCMD, the application does not stop if a calculation or data load is in progress. Instead, Essbase displays a message in the Agent window. If you stop the Agent by closing the Agent window or by pressing `Ctrl+C`, Essbase rolls back any transactions that were in progress (see Chapter 41, Ensuring Data Integrity, for information about transaction processing).

Starting an Application

To start an application from the Agent:

Type `START` *`appname`* at the command prompt on the server console.

You can also start an application by:

- Creating a new application, which automatically starts the application.
- Using the Application|Start/Stop command from the Application Manager.

  If the application is not already loaded, a confirmation box appears. Click **Yes** to load the application.

  If the application is already loaded, the confirmation box asks if you want to stop the application. Click **No** to prevent Essbase from stopping the application.
- Starting a database within an application, as discussed under "Starting and Stopping Databases" on page 44-6.
- Saving an outline to the server.

Note: Opening an outline does *not* start an application.

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

Stopping an Application

When you stop an application, transactions may be currently running. If you issue the STOP command in the Agent, or UNLOADAPP in ESSCMD, the application does not stop if a calculation or data load is in progress. Instead, Essbase displays a message in the Agent window. If you stop the Agent by closing the Agent window or by pressing `Ctrl+C`, the next time you start the application, Essbase rolls back any transactions that were in progress (see Chapter 41, Ensuring Data Integrity for information about transaction processes).

To stop an application from the Agent:

Type `STOP` *`appname`* at the command prompt on the server console.

You can also stop an application by using the Application | Start/Stop command from the Application Manager:

- If the application is already loaded, a confirmation box appears. Click **Yes** to stop the application.
- If the application is not already loaded, the confirmation box asks if you want to start the application. Click **No** to prevent Essbase from starting the application.

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

Starting and Stopping Databases

Starting a database means loading the database into memory on the server. Stopping a database unloads all information for the database from memory onto the server.

When you stop a database, transactions may be currently running. If you issue the STOP command in the Agent, or UNLOADDB in ESSCMD, the database does not stop if a calculation or data load is in progress. Instead, Essbase displays a message in the Agent window. If you stop the Agent by closing the Agent window or by pressing `Ctrl+C`, the next time you start the database, Essbase rolls back any transactions that were in progress (see Chapter 41, Ensuring Data Integrity, for information about transaction processes).

Starting a Database

You can start a database by:

- Creating a new database, which automatically starts the database.
- Using the Database | Start/Stop command from the Application Manager.

  If the database is not already loaded, a confirmation box appears. Click **Yes** to load the database.

  If the database is already loaded, the confirmation box asks if you want to stop the database. Click **No** to prevent Essbase from stopping the database.
- Saving an outline to the server.

Note: Opening an outline does *not* start a database.

- Clicking the **Statistics** tab in the **Database Information** dialog box.

Note: When you start a database from an application that is not loaded, the application is loaded along with all its related databases.

You can use the LOADDB, LOGIN, or SELECT command in ESSCMD to perform this task. See the online *Technical Reference* in your `DOCS` directory for more information about each of these commands. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Stopping a Database

To stop an application from the Agent:

Type `STOP` *`appname`* at the command prompt on the server console.

This stops the application that contains the database you want to stop.

You can also stop an application by:

- Using the Database | Start/Stop command from the Application Manager.

  If the database is already loaded, a confirmation box appears. Click **Yes** to stop the database.

  If the database is not already loaded, the confirmation box asks if you want to start the database. Click **No** to prevent Essbase from starting the database.

- Using the Application Start/Stop command from the Application Manager to stop the application that contains the database you want to stop

  If the application is already loaded, a confirmation box appears. Click **Yes** to stop the application.

  If the application is not already loaded, the confirmation box asks if you want to start the application. Click **No** to prevent Essbase from starting the application.

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

Viewing a List of Users and Available Ports

The Agent allows you to view a list of all users that are connected to the server at any given time. Additionally, you can view the total number of ports available, as well as the number of existing connections.

To view a list of all users connected to the server, type the command `USERS` at the command prompt of the server console. The server console displays the following information:

- The names of all users connected to the server
- The total number of ports available
- The total number of existing connections
- The application to which each user is connected
- The database to which each user is connected

You can use the LISTUSERS command in ESSCMD to view a list of users that have been defined on the selected server. Using this command does not return information about the available ports and connections or the application or database users are connected to. See the online *Technical Reference* in your `DOCS` directory for more information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Disconnecting a User From the Server

To disconnect a user from the server, type the command LOGOUTUSER *user* at the command prompt of the server console. The user's port is now free.

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

Changing the System Password

You can change the password that is required to start the Essbase server from the Agent's server console. Changing the system password does not change the connection password that is established for the Essbase System Supervisor.

To change the Essbase system password:

1. Type PASSWORD at the command prompt of the server console.

   Essbase prompts you to enter the old system password, as shown in Figure 44-2.

Figure 44-2, Changing the Essbase System Password

2. Type the old (current) system password.
3. Type the new system password that you want to use.
4. Retype the new system password that you want to use.

   Essbase verifies that the system password has been updated.

For information about changing passwords for users or groups, see Chapter 16, Managing Security at Global and User Levels.

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

Displaying the Server Software Version Number

To see the version number of the Essbase server software, type `VERSION` at the command prompt of the server console.

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

Viewing Ports

To see the number of ports installed on the server, as well as the number of ports in use, type `PORTS` at the command prompt of the server console.

Viewing Security System Information

To output information from the Essbase security system into a specified file in ASCII format, type `DUMP` *`filename`* at the command prompt of the server console. Essbase prompts you for the Essbase system password.

Shutting Down All Open Applications and Quitting Essbase

To shut down all open applications and quit Essbase from the Agent's server console, type `QUIT` or `EXIT` at the command prompt.

Client/Server Communications

The Essbase client/server architecture supports enterprise analysis applications. The server runs a Server Agent (ESSBASE.EXE) process that acts as a traffic coordinator for all user requests to Essbase applications. The application server (ESSSVR) fields data requests from clients. An Essnet library connects Essbase to the network and enables the server and the clients to communicate with each other, as shown in Figure 44-3.

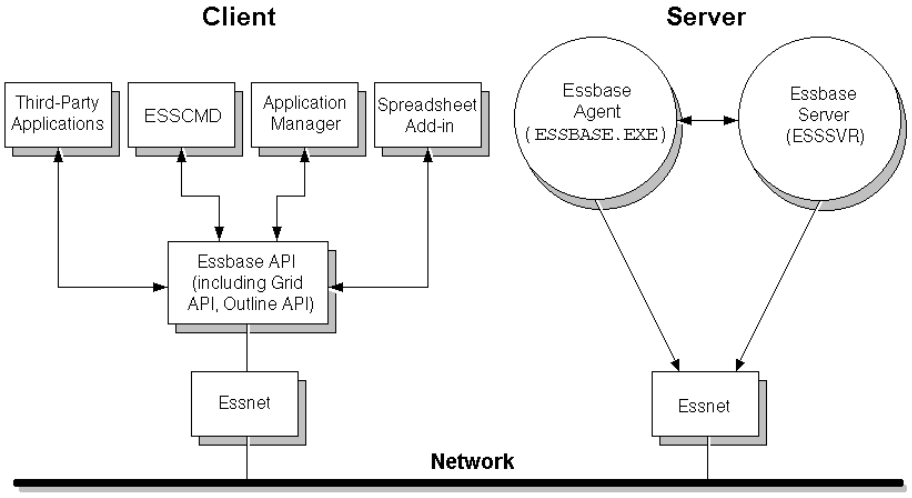

Figure 44-3, Essbase Client/Server Communications

Essbase Client Types

Essbase clients retrieve and analyze data from the server with Lotus 1-2-3, Excel, or a custom application interface. There are three types of Essbase clients:

- The Hyperion Essbase Spreadsheet Add-in, which provides users with seamless data access to the server.
- The Hyperion Essbase Application Manager. The Application Manager user designs, develops, and maintains Hyperion Essbase Applications.
- A custom application built with the Hyperion Essbase Application Programming Interface (API). The online *API Reference* in your DOCS directory provides a complete listing of Essbase API functions, platforms, and supported compilers.

As shown in Figure 44-4, Essbase handles client requests differently, depending on the type of the request. Administrative requests, such as logins and logouts, starting and stopping applications and databases, and viewing user security information, are handled by the Essbase Agent (ESSBASE.EXE). Client requests for data, such as data loads, calculations, spreadsheet reports, and data lock/unlock, are handled by the application server (ESSSVR).

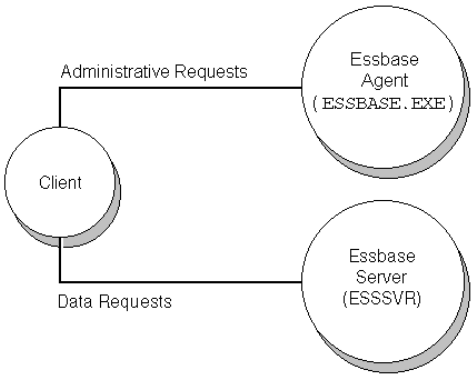

Figure 44-4, Essbase Client Request Handling

Communications Event Flow

When a user logs in to Essbase, the following events occur:

1. The Essbase client logs in, using a predefined address:
   - For TCP/IP connections, the address is port number `1423`.
   - For Named Pipes connections, the address is `\\pipe\essbase0`.
2. The user selects an application/database combination.
3. The Agent compares the requested application with applications currently running. If the specified application is already running, the Agent does not need to do anything. If the requested application is not already running, the Agent initiates startup, and the following events occur:
   a. The Agent assigns a dynamic port for the application server or creates a dynamic name for a named pipe.
   b. The application server returns the port number for the application to the Agent and to the client. Port number information is stored at run time in the Essbase Application Programming Interface (API).

4. The Agent sets the application server as active with the current user and security information. When the client later sends a data request to the application server, the security information captured by the API is embedded in the request.
5. The Agent sends the client query to the application server (ESSSVR).
6. For every request that the client makes, the application server causes the following events to occur:
   a. Connect
   b. Send request
   c. Get response
   d. Send data
   e. Receive data
   f. Disconnect

Multithreading

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

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

When the Essbase Agent receives a client request, it initiates an inter-process communication (IPC). This process starts the requested application if it is not already running and assigns a port for the requested application. The server then returns a port number to the Agent and the client. When this activity concludes, the Agent thread finishes the IPC with the Agent and, if initialization is successful, loops and waits to receive the next request from the Agent.

Essbase launches a static number of threads to handle multiple users. The number of threads is based on the number of licensed ports you purchase, as shown in Table 44-1. The number of ports represents the number of concurrent connections Essbase supports. Essbase provides one reserve port for the System Administrator to use to log out one or more users if all ports are in use.

Table 44-1, Licensed Ports and Multithreading

Number of Licensed Ports	Number of Threads
5 or fewer ports	5
10 or fewer ports	10
20 or more ports	20

You can set the number of threads for the Agent or the server in the ESSBASE.CFG file. For complete details on setting the number of thread for the Agent and the server, refer to the AGENTTHREADS and SERVERTHREADS settings in the online *Technical Reference* in your DOCS directory.

Monitoring Applications

Each application that is loaded is an open task or process in the operating system. Under Windows NT, Windows 95, and Windows 98 platforms, the application appears in an Essbase server window. On UNIX platforms, the application server is often a background process. When the application starts, ESSBASE.EXE (ESSBASE on UNIX systems) starts the ESSSVR process.

The application server records its activities, such as writing data to an application log file, *appname*.LOG, in the \ESSBASE\APP*appname* directory. You can open and view this text file when you need to troubleshoot problems or view application activity.

On Windows NT, Windows 95, and Windows 98, you can also view application activities as they occur in the Essbase server window. On UNIX, you can route all messages to a file with the `tail -f` *logfile* command.

On Windows NT, Windows 95, and Windows 98 platforms, when the Agent starts an application, a new icon appears on the desktop, as in the following illustration:

Figure 44-5, Essbase Application Server Icon

On Windows 95 and Windows 98, a new icon appears in the Taskbar.

You can double-click the icon to view the server window.

Chapter 45 Monitoring Performance Using Diagnostics

This chapter describes the Hyperion Essbase OLAP Server diagnostic tools available to you and tells where to find log files that can help you analyze and solve problems. Use this section not only when problems occur, but any time you want to check or analyze Essbase configuration or performance.

This chapter contains the following sections:

Overview of Diagnostic Tools

Essbase application, server and database information dialog boxes in the Hyperion Essbase Application Manager are useful when you are preparing to migrate data, add users, analyze performance problems, or perform other administrative tasks. These dialog boxes are available from the Server, Application, and Database menus in the Application Manager. For details about accessing specific information in dialog boxes and the pages they contain, see Table 45-1.

Note: For information about a specific field in any dialog box, display the dialog box and press the Help button.

Viewing Application Information

The **Application Information** dialog box shown in Figure 45-1 provides information about your application and its databases.

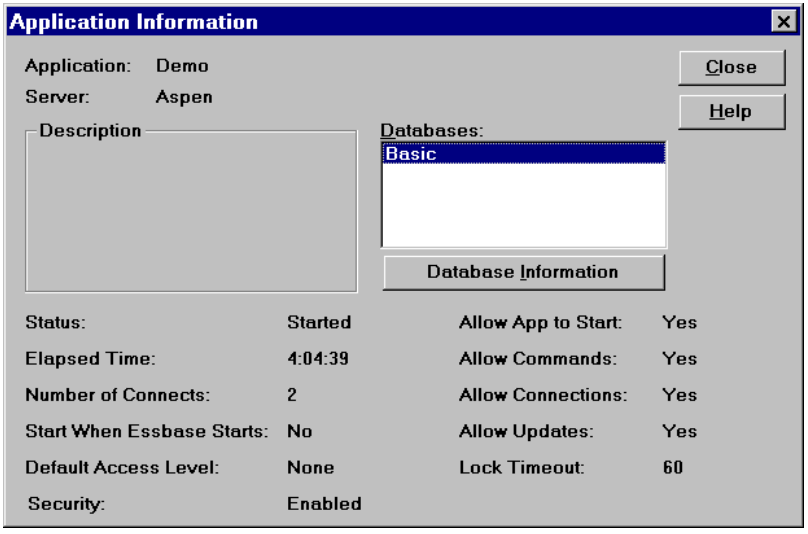

Figure 45-1, The Application Information Dialog Box

Use the **Application Information** dialog box to verify which databases are running in the application and to check access, security, and start-up information.

You can also display the **Database Information** dialog box (see "Viewing Database Information" on page 45-9) from the **Application Information** dialog box by selecting a database in the list box and clicking the **Database Information** button.

The settings and other information in the **Application Information** dialog box are for information only. If you want to customize your application, you can change the settings in the **Application Settings** dialog box, as described in Chapter 16, Managing Security at Global and User Levels.

Click the Help button for information on each label.

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

Viewing Server Information

This dialog box provides information, grouped by tabs, on your server license, configuration, operating system, disk drives, and applications.

Essbase displays information on a "snapshot" basis; to see the latest information, click on the **Refresh** button for a new snapshot. The **Refresh** button refreshes every page in the dialog box, not just the current page.

The settings and other information in the **Server Information** dialog box are for information only and reflect activity on the server. The dialog box doesn't let you change settings; its purpose is to help you identify activities or operations that may be affecting performance.

Viewing License Information

Use the **License Info** page to see information about the Essbase installation; for example, you can verify whether Essbase is correctly installed and which options are installed.

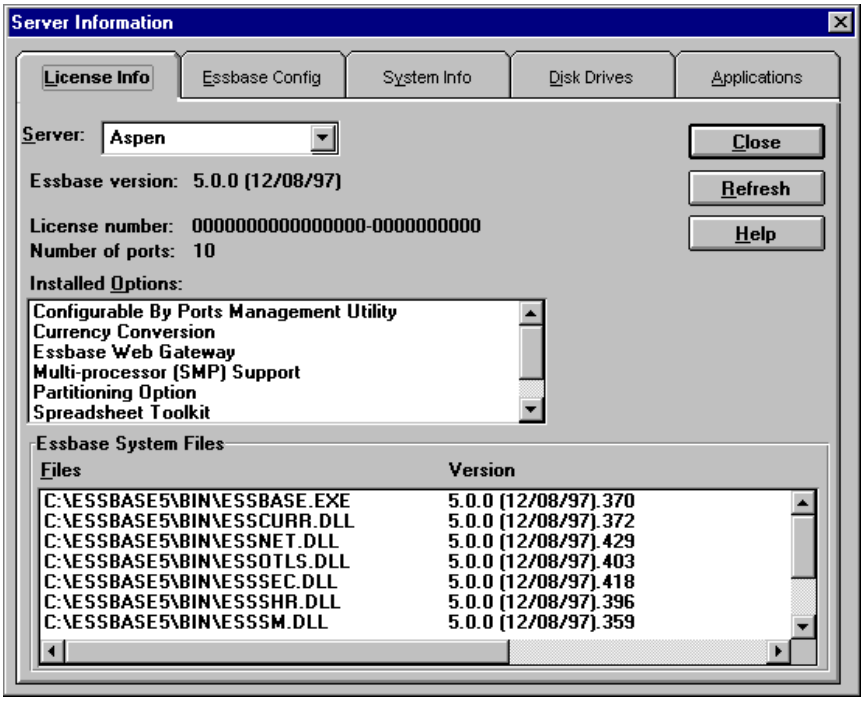

Figure 45-2, License Info Page of the Server Information Dialog Box

Click the Help button for information on each label.

Viewing Configuration Information

Use the **Essbase Config** page to see information about Essbase status and configuration. It reflects the ESSBASE.CFG server file settings, if you have created such a file; if you have not, the **Global ESSBASE.CFG Settings** list box is blank. See the online *Technical Reference* in your DOCS directory for information on how to create this file.

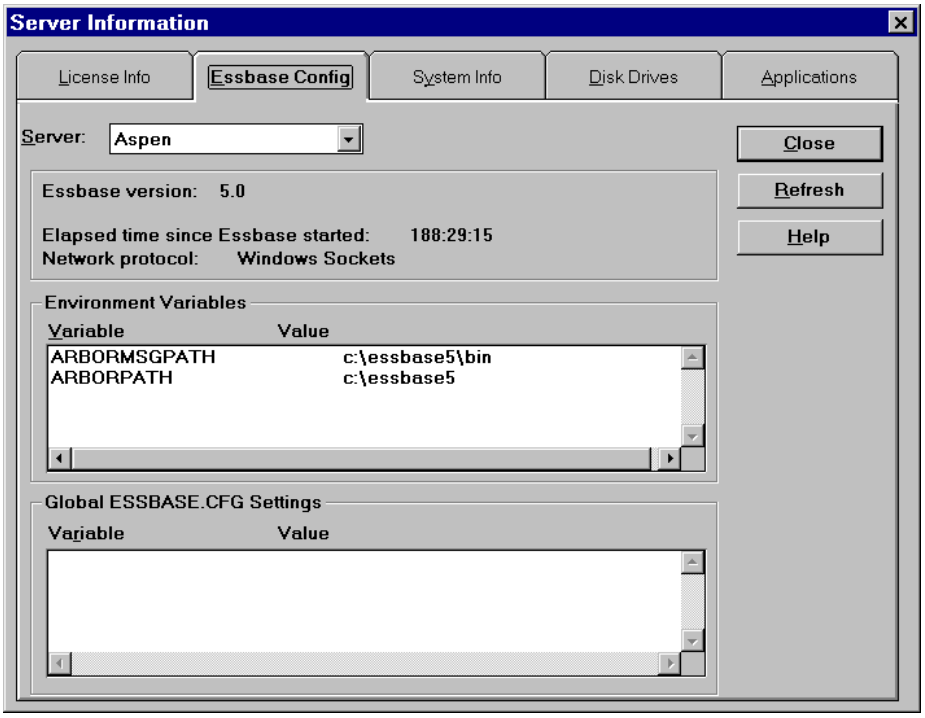

Figure 45-3, Essbase Config Page of the Server Information Dialog Box

The **Environment Variables** list box displays your operating system environment variables, as defined during installation. You can use this list box to verify path information.

Click the Help button for information on each label.

Viewing System Information

Use the **System Info** page to get information about the operating system and resource usage. This page lets you determine whether machine or operating system constraints are impacting Essbase performance.

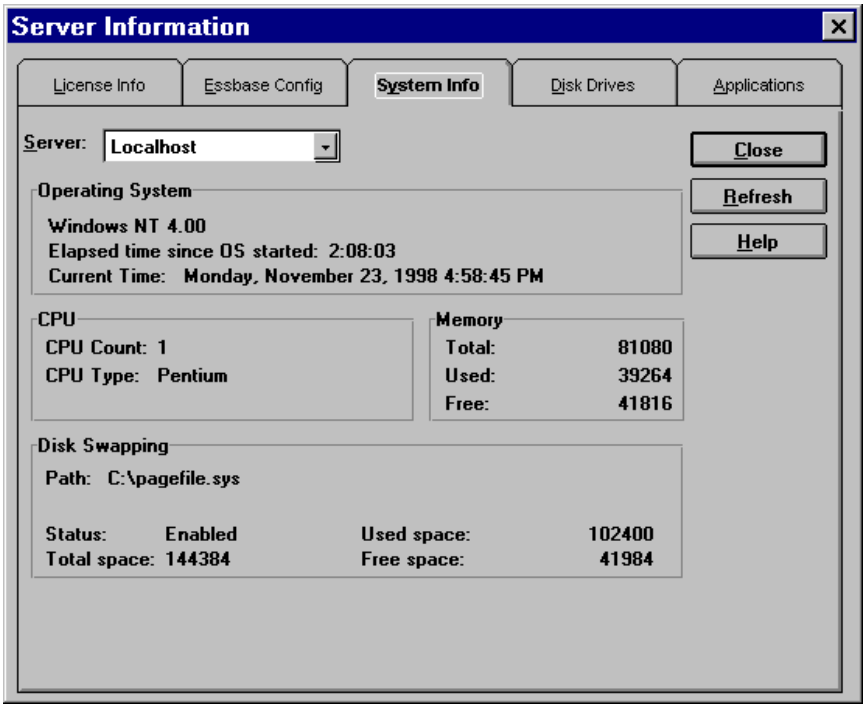

Figure 45-4, System Info Page of the Server Information Dialog Box

For example, suppose Essbase is running too slowly. Check the **Memory** and **Disk Swapping** groups to see how much free space you have. Assume that the **Free space** value in **Disk Swapping** is very low.

Take the following action, depending your platform:

- On Windows 95, Windows 98 and Windows NT, increase the maximum virtual memory, or add more volumes for disk swapping.
- On UNIX, increase your swap space, or add another swapping device.

Click the Help button for information on each label.

Viewing Disk Drive Information

The **Disk Drives** page contains information about disk drive types, disk drive usage, and file system types. Use this information to determine whether lack of disk space or incompatible file system types are impacting Essbase performance.

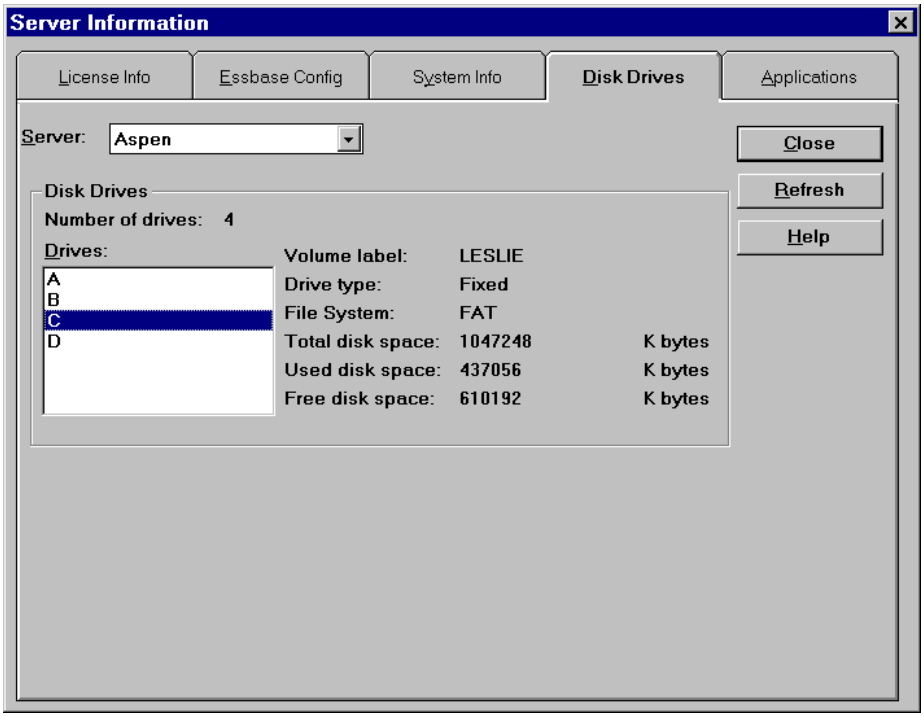

Figure 45-5, Disk Drives Page of the Server Information Dialog Box

For example, if information on the **System Info** page indicates that swap space is low, you can use the **Disk Drives** page to see which drives have space available.

You can also use the **Disk Drives** page to see which drives have space available when you want to allocate space on another drive. See "Specifying Disk Volumes" on page 40-16 for information about disk spanning using the Disk Volumes setting.

Click the Help button for information on each label.

Viewing Application Status Information

Use the **Applications** page to see which applications and databases are loaded.

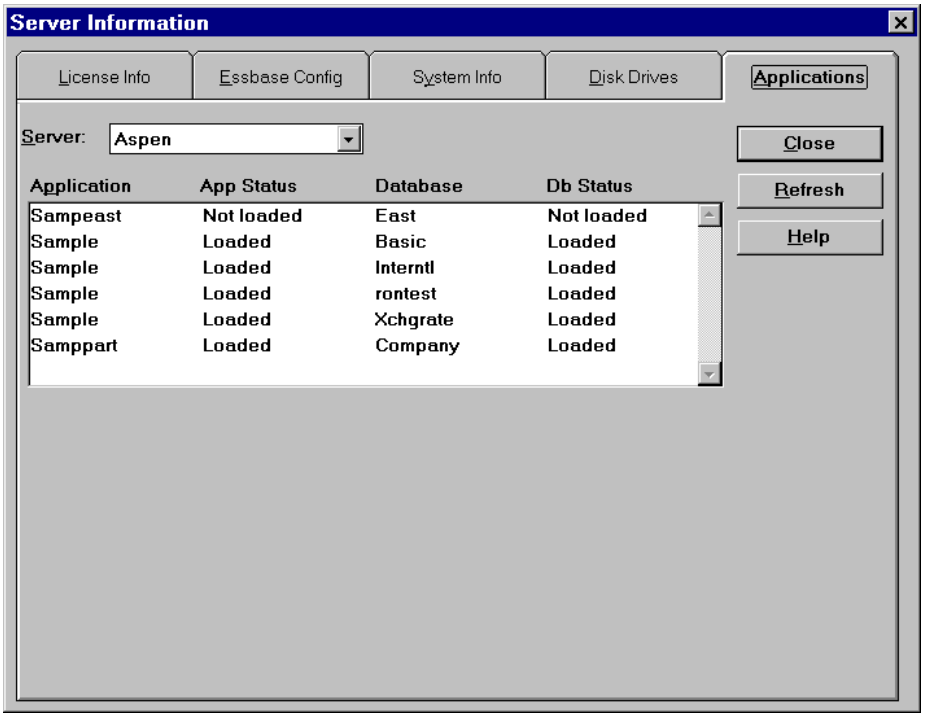

Figure 45-6, Applications Page of the Server Information Dialog Box

Essbase lists only those applications that you are authorized to use.

Click the Help button for information on each label.

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

Viewing Database Information

The **Database Information** dialog box provides information, grouped by tabs, such as database storage, database statistics, and lock contention. To see the **Database Information** dialog box, select a database, then choose Database | Information from the Application Manager.

The server name, application name and database name are displayed on every page. Essbase displays information on a "snapshot" basis; to see the latest information, click on the **Refresh** button for a new snapshot. The **Refresh** button is available on every dialog box page. Each time you click it, the **Refresh** button refreshes every page in the dialog box, not just the current page.

The settings and other information in the **Database Information** dialog box are for information only and reflect activity on the server. The dialog box doesn't let you change settings; its purpose is to help you identify activities or operations that may be affecting performance.

Click the Help button for information on each label.

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

For information on changing your database settings, see Chapter 40, Specifying Storage Manager Settings.

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

Viewing General Database Information

Use the **General** page to check database status and start-up information, cache sizes, and the data compression setting.

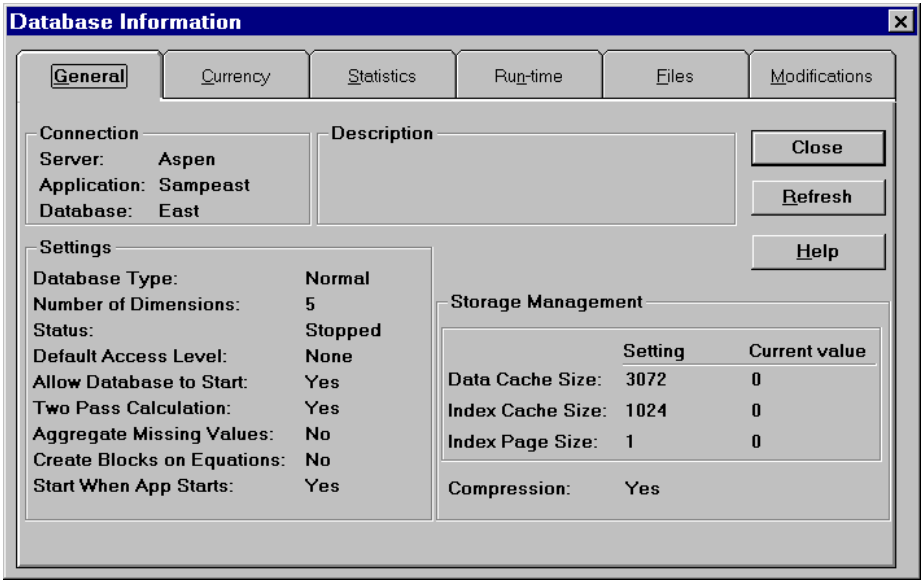

Figure 45-7, General Page of the Database Information Dialog Box

The **Compression** field indicates whether data compression is enabled (`Yes`) or disabled (`No`). If this value is `Yes`, Essbase compresses data blocks as it writes them out to disk. This field does not indicate the type of data compression that is enabled. See Chapter 40, Specifying Storage Manager Settings, for information about specifying data compression.

Click the Help button for information on each label.

You can use the GETDBSTATE command in ESSCMD to perform this task. Using this command also provides currency database information that is on a separate (**Currency**) page of the **Database Information** dialog box that you access from the Application Manager desktop. See the online *Technical Reference* in your `DOCS` directory for more information about this command. See Chapter 43, Performing Interactive and Batch Operations Using ESSCMD, for information about ESSCMD.

Viewing Currency Database Information

Use the **Currency** page to examine the configuration of the currency database associated with the current database, if the current database is linked to a currency database.

Note: Hyperion Essbase Currency Conversion is an optional module, licensed separately. It might not be installed on your server.

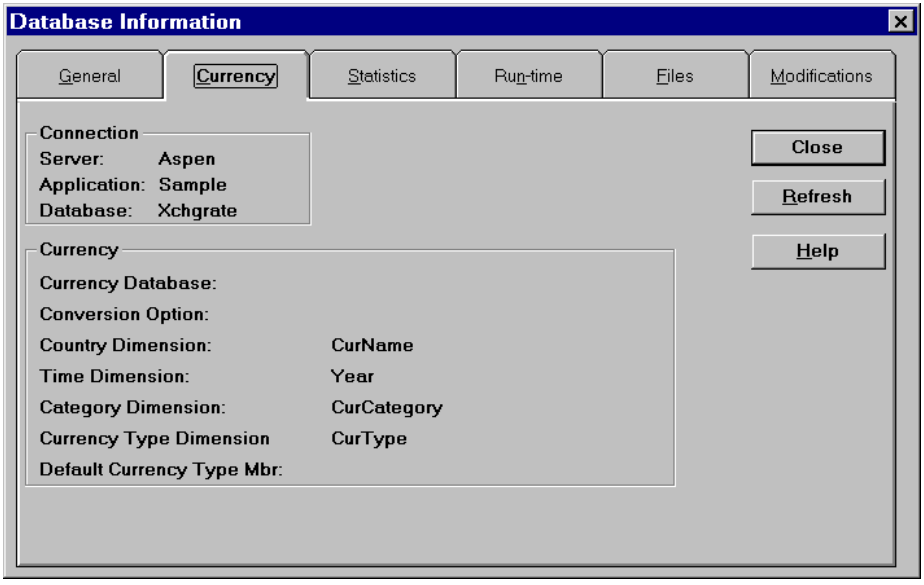

Figure 45-8, Currency Page of the Database Information Dialog Box

Click the Help button for information on each label.

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

Viewing Database Statistics

Use the **Statistics** page for dimension information and to see characteristics of data blocks in your database.

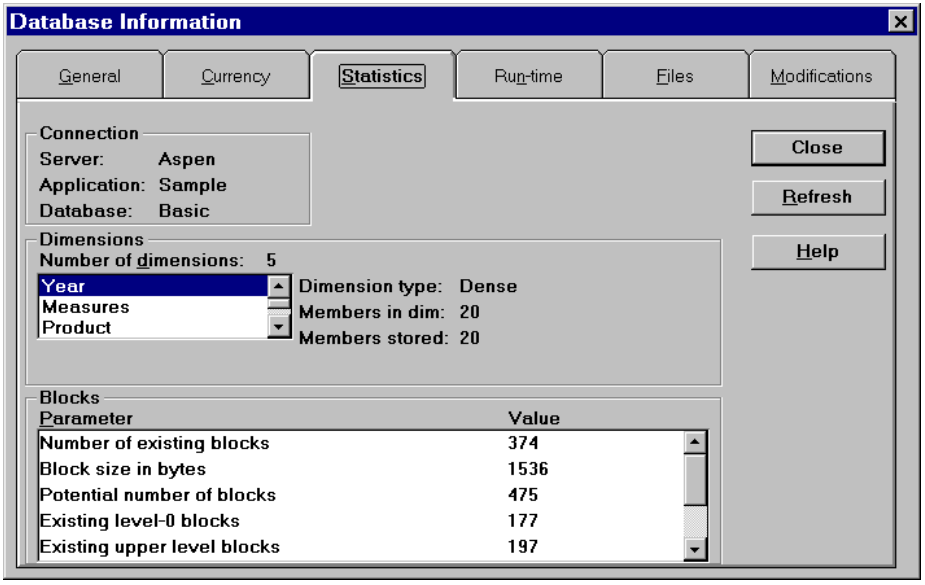

Figure 45-9, Statistics Page of the Database Information Dialog Box

To check on block density and usage, scroll down to the **Block Density** and **Percentage of maximum blocks existing** fields.

Block density is the average percentage of cells containing data in a datablock; **Percentage of maximum blocks existing** is the percentage of existing blocks compared to the number of potential blocks.

If you have a low block density and a high percentage of maximum blocks, you might want to increase the number of dense dimensions. See Chapter 4, Basic Architectural Elements, for background information on dimension types.

To check block size, look at the **Block size in bytes** field. This is the size of the expanded block, whether or not you have data compression enabled. Ideally, this value should be between eight kilobytes and 64 kilobytes. To alter block size, you need to change your dense/sparse configuration.

The **Compression Ratio** indicates the ratio of the compressed block size to the expanded block size. For more information about data compression, see Chapter 39, Introducing the Essbase Storage Manager, and Chapter 40, Specifying Storage Manager Settings.

If the database is not started when you choose the **Statistics** tab, Essbase prompts you to start the database.

Click the Help button for information on each label.

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

Viewing Run-Time Information

The **Run-time** page displays run-time information such as lock contention, insufficient cache size (buffer shortage), "hit" ratio on index searches (percentage of searches that did not involve fetching from disk), and the number of read and write operations.

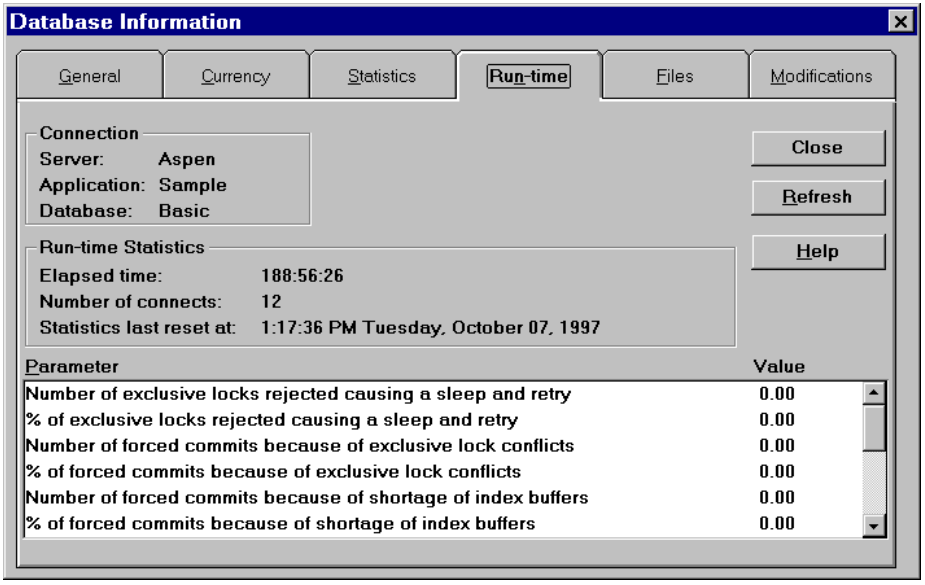

Figure 45-10, Run-Time Page of the Database Information Dialog Box

For example, suppose you want to know if your index cache size is large enough. One indicator to check is the **Hit ratio on Index Cache** field (you need to scroll down in the **Parameter** list box to see it). This value indicates success rate of Essbase in locating index data in memory without having to fetch it from the disk. Ideally, this value should be near 100 percent. If the number of hits is very low, the index cache (buffer) is too small.For information on setting the index cache size, see "Using the Settings Dialogs in the Application Manager" on page 40-5.

If the database is not started when you choose the **Run-Time** tab, Essbase prompts you to start the database.

Click the Help button for information on each label.

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

Viewing Database File Information

The **Files** page lists the index and page files associated with the current application and database.

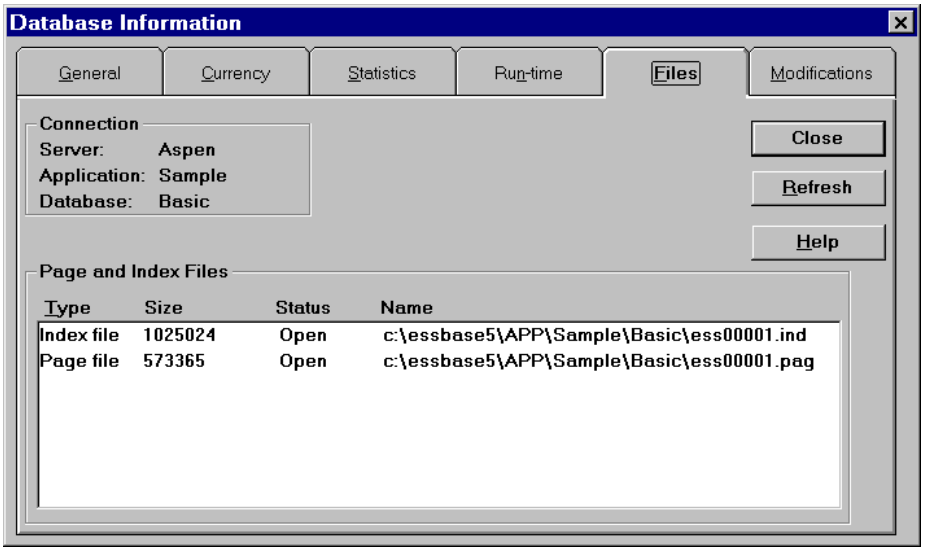

Figure 45-11, Files Page of the Database Information Dialog Box

This page lets you see where index file or page (data) files are stored, and whether they span multiple volumes.

Note: If the database in the **Database** field is started, the index files and data (page) files should be open. If they are closed, first click the **Refresh** button. If the files are still closed, a problem may have occurred starting the database. Check your application and server log files (see "The Application Event Log File" on page 45-27 and "The Server Event Log File" on page 45-30) to see whether you can determine the problem. If you cannot, call the technical support department of your software provider. You can also check the **Applications** page of the **Server Information** dialog box (see "Viewing Application Status Information" on page 45-8) to see if the database is started.

Click the Help button for information on each label.

Viewing a Record of Database Modifications

Use the **Modifications** page to see information about the last successful data load, calculation, and outline update operations for the current application.

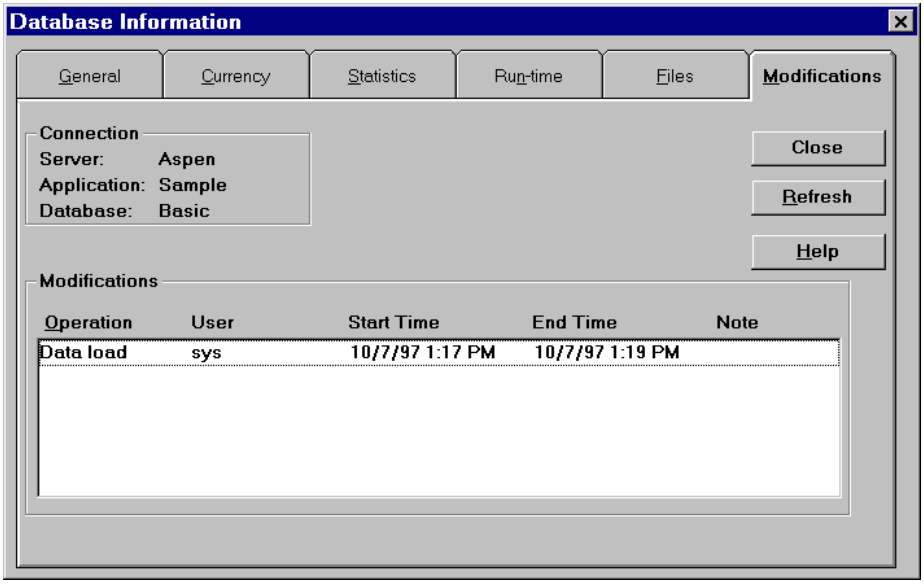

Figure 45-12, Modifications Page of the Database Information Dialog Box

Note: **Start Time** and **End Time** include the necessary preparation Essbase does prior to the operation, such as locking the data. To see the duration of the actual operation itself, see the "Elapsed time" entry in the application log. "The Application Event Log File" on page 45-27 describes this log and where to find it.

Click the Help button for information on each label.

Quick Reference to Diagnostic Tools

Table 45-1 provides detailed procedures for accessing the correct dialog boxes to perform common server, database, and application maintenance tasks. Use this table to look up the proper menu commands and dialog box pages to choose when you want to view specific information displayed in the **Server Information**, **Application Information**, or **Database Information** dialog boxes.

Table 45-1, Quick Reference to Diagnostic Tools

To do this...	Open this dialog box/page
To see a list of the **servers you are currently connected to**	Choose Server|Information to display the **Server Information** dialog box. Choose the **License Info** page to view the **Server** list box.
To view the **version of Essbase** that is currently running on the server	Choose Server|Information to display the **Server Information** dialog box. Choose the **License Info** page to see the **Version** label.
To check the **license number** of your copy of Essbase	Choose Server|Information to display the **Server Information** dialog box. Choose the **License Info** page to view the **License number** label.
To see the **total number of ports** that can connect to the server	Choose Server|Information to display the **Server Information** dialog box. Choose the **License Info** page to view the **Number of ports** label.
To see a **list of all features** in place when Essbase was installed	Choose Server|Information to display the **Server Information** dialog box. Choose the **License Info** page to view the features list under the **Installed Options** label.
To see a **list of the system files** currently in the server memory, as well as their locations and version numbers	Choose Server|Information to display the **Server Information** dialog box. Choose the **License Info** page to view the **Essbase System Files** list box.
To see **elapsed time since Essbase started**	Choose Server|Information to display the **Server Information** dialog box. Choose the **Essbase Config** page to view elapsed time.
To check the **type of network protocol** the server is using	Choose Server|Information to display the **Server Information** dialog box. Choose the **Essbase Config** page to see the **Network protocol** label.
To view **environment variables**, including source path information	Choose Server|Information to display the **Server Information** dialog box. Choose the **Essbase Config** page to see the **Environment Variables** list box.

Table 45-1, Quick Reference to Diagnostic Tools

To do this...	Open this dialog box/page
To view the **version of the operating system** on which Essbase is running	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to view the **Operating System** label.
To see **the time that has elapsed since the operating system was started or rebooted** (in hours:minutes:seconds format)	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to view the **Operating System** label.
To see **the time and date** the Server Information was retrieved, according to the server	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to view the **Operating System** label.
To see the **number and type of CPUs** in the system on which the server is operating	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to view the **CPU** option group.
To see the **amount of available memory** on the operating system	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to see the **Memory option** group.
To see the **disk swap path**	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to view the **Path** label in the **Disk Swapping** option group.
To check if **disk swapping is enabled**	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to view the **Status** label in the **Disk Swapping** option group.
To see the **total swap space**	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to view the **Total space** label in the **Disk Swapping** option group.
To see the **swap file size**	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to view the **Used space** label in the **Disk Swapping** option group.
To see **available swap space**	Choose Server|Information to display the **Server Information** dialog box. Choose the **System Info** page to view the **Free space** label in the **Disk Swapping** option group.
To view the **drive type** (Fixed, Removable, RAM, Remote, or Unknown)	Choose Server|Information to display the **Server Information** dialog box. Choose the **Disk Drives** page and select the drive in the **Drive** list box. The **Drive type** label displays the drive type.

Table 45-1, Quick Reference to Diagnostic Tools

To do this...	Open this dialog box/page
To view the **file system used by a drive** (FAT, HPFS, NTFS, etc.)	Choose Server|Information to display the **Server Information** dialog box. Choose the **Disk Drives** page and select the drive in the **Drive** list box. The **File System** label displays the type of file system used by the drive.
To view the **total amount of space on a drive**	Choose Server|Information to display the **Server Information** dialog box. Choose the **Disk Drives** page and select the drive in the **Drive** list box. The **Total space** label displays the total disk space.
To view the **amount of space being used** on a drive	Choose Server|Information to display the **Server Information** dialog box. Choose the **Disk Drives** page and select the drive in the **Drive** list box. The **Used disk space** label displays the amount of space currently being used by the selected drive.
To view the **amount of space available** on a drive	Choose Server|Information to display the **Server Information** dialog box. Choose the **Disk Drives** page and select the drive in the **Drive** list box. The **Free disk space** label displays the amount of space currently available on the selected drive.
To see **which applications and databases are loaded**	Choose Server|Information to display the **Server Information** dialog box. Choose the **Applications** page. The **Application** list box displays the following information: • A list of the applications that you are authorized to use • The status of the listed applications (Loaded, Not Loaded, Loading, and Unloading) • A list of available databases • The status of each database (Loaded, Not Loaded, Loading, and Unloading)
To view your **database connections**	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The server, application and database currently loaded are displayed in the **Connection** group.
To view your **database type**	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. **Database Type** is displayed in the **Settings** group.

Table 45-1, Quick Reference to Diagnostic Tools

To do this...	Open this dialog box/page
To view the **number of dimensions** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The **Number of Dimensions** option is displayed in the **Settings** group.
To view the **status of a database**	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. **Status** (for example, **Stopped**) is displayed in the **Settings** group.
To view the **default access level to a database**	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The **Default Access Level** is displayed in the **Settings** group.
To see **whether users may start the database**	Choose Database|Information to display th**e Database Information** dialog box. Choose the **General** page. The **Allow Database to Start** option is displayed in the **Settings** group.
To see **if your database aggregates missing values**	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The **Aggregate Missing Values** option is displayed in the **Settings** group.
To see **if your database creates blocks on equations**	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The **Create Blocks on Equations** option is displayed in the **Settings** group.
To see **if a database is set to start automatically** when an application is started	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The **Start When App Starts** option is displayed in the **Settings** group
To view the **data cache size** in kilobytes of a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The **Setting** and **Current Value** of the data cache are displayed following the **Data Cache Size** option in the **Storage Management** option group.
To view the **index cache size** kilobytes of a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The **Setting** and **Current Value** of the index cache are displayed following the **Index Cache Size** option in the **Storage Management** option group.

Table 45-1, Quick Reference to Diagnostic Tools

To do this...	Open this dialog box/page
To view the **index page size** kilobytes of a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The **Setting** and **Current Value** of the index pages are displayed following the **Index Page Size** option in the **Storage Management** option group.
To see **if the compression option is enabled** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **General** page. The **Compression** option is displayed in the **Storage Management** option group.
To see **if you're connected to a currency database**	Choose Database|Information to display the **Database Information** dialog box. Choose the **Currency** page. See **Currency Database** in the **Currency** option group
To see **if the currency conversion option is on** in a currency database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Currency** page. See **Conversion Option** in the **Currency** option group.
To see **if a dimension is set to a specific country** in a currency database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Currency** page. See **Country Dimension** in the **Currency** option group.
To **view the time dimension** setting in a currency database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Currency** page. See **Time Dimension** in the **Currency** option group.
To **view the category dimension** setting in a currency database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Currency** page. See **Category Dimension** in the **Currency** option group.
To **view the currency partition dimension** setting	Choose Database|Information to display the **Database Information** dialog box. Choose the **Currency** page. See **Currency Partition Dimension** in the **Currency** option group.
To see the **default type of currency** a member is set to in a currency database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Currency** page. See **Default Currency Type Mbr** in the **Currency** option group.

Table 45-1, Quick Reference to Diagnostic Tools

To do this...	Open this dialog box/page
To view the **number of dimensions** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. See **Number of Dimensions** in the **Dimensions** option group.
To view a **list of the dimensions** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. All of the dimensions in the database are listed in the list box in the **Dimensions** option group.
To see **if a dimension is designated as sparse or dense**	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. Choose the specific dimension from the list box in the **Dimensions** option group. The dimension type (either **Sparse** or **Dense**) is displayed following the **Dimension type** label.
To see the **number of members** in a dimension	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. Choose the specific dimension from the list box in the **Dimensions** option group. The number of members in the dimension is displayed following the **Dimension type** label.
To see the **number of stored members** in a dimension	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. Choose the specific dimension from the list box in the **Dimensions** option group. The number of stored members in the dimension is displayed following the **Members in dim** label.
To view the **number of existing blocks** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. In the **Blocks** option group, the **Number of existing blocks** label is displayed in the list box beneath the **Parameter** heading. The number of blocks is displayed in the same row, beneath the **Value** heading.

Table 45-1, Quick Reference to Diagnostic Tools

To do this...	Open this dialog box/page
To view the **block size in bytes**	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. In the **Blocks** option group, the **Block size in bytes** label is displayed in the list box beneath the **Parameter** heading. The block size is displayed in the same row, beneath the **Value** heading. **Note:** Block size varies widely from block to block.
To view the **potential number of blocks** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. In the **Blocks** option group, the **Potential number of blocks** label is displayed in the list box beneath the **Parameter** heading. The number of potential blocks is displayed in the same row, beneath the **Value** heading.
To view the **number of existing level-0 blocks** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. In the **Blocks** option group, the **Existing level-0 blocks** label is displayed in the list box beneath the **Parameter** heading. The number of existing level-0 blocks is displayed in the same row, beneath the **Value** heading.
To view the **number of existing upper-level blocks** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. In the **Blocks** option group, the **Existing upper level blocks** label is displayed in the list box beneath the **Parameter** heading. The number of existing upper-level blocks is displayed in the same row, beneath the **Value** heading.
To view **block density** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. In the **Blocks** option group, the **Block density** label is displayed in the list box beneath the **Parameter** heading. Block density is displayed in the same row, beneath the **Value** heading. **Note:** Density varies widely from block to block.

Table 45-1, Quick Reference to Diagnostic Tools

To do this...	Open this dialog box/page
To view the **percentage of the maximum number of blocks** existing in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. In the Blocks option group, the **Percentage of maximum blocks existing** label is displayed in the list box beneath the **Parameter** heading. The percentage of maximum blocks is displayed in the same row, beneath the **Value** heading.
To view the **compression ratio of blocks** in a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Statistics** page. In the **Blocks** option group, the **Compression Ratio** label is displayed in the list box beneath the **Parameter** heading. The compression ratio is displayed in the same row, beneath the **Value** heading.
To view the **elapsed run-time that has elapsed since the database was started**, according to the serve (in hours:minutes: seconds format).	Choose Database|Information to display the **Database Information** dialog box. Choose the **Run-time** page. See **Elapsed time** in the **Run-time Statistics** option group.
To view the **number of users connected** to the database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Run-time** page. See the **Number of connects** label in the **Run-time Statistics** option group.
To view the **number and % of exclusive locks rejected causing a sleep and retry** (number of exclusive locks placed on database blocks that were rejected due to lock contention) and the **number and % of forced commits because of exclusive lock conflicts** (the number of times Essbase wrote modified blocks to disk because more than one user attempted an update operation on the same blocks).	Choose Database|Information to display the **Database Information** dialog box. Choose the **Run-time** page. See the **Parameter** list box.

Table 45-1, Quick Reference to Diagnostic Tools

To do this...	Open this dialog box/page
To view the **size and status of a data (page) file or an index file**	Choose Database|Information to display the **Database Information** dialog box. Choose the **Files** page. In the **Page and Index Files** option group, the file type (page or index) is displayed under the **Type** heading; the size of the file is displayed under **Size**; the status of the file (open or closed) is displayed under **Status**; and the name of the file is displayed under **Name**.
To see **details about modifications** to a database	Choose Database|Information to display the **Database Information** dialog box. Choose the **Modifications** page. The **Operation** (type of modification), the **User**, the **Start Time** and the **End Time** are all displayed in the list box in the **Modifications** option group.

Exception Error Handling

The Essbase Exception Handler gathers information that can help the technical support department of your software provider to pinpoint where and why an error has occurred. Essbase maintains two exception error logs, one for the application and one for the server.

- "The Application Event Log File" on page 45-27
- "The Server Event Log File" on page 45-30

Examples of events that can cause a server crash include:

- Access violation
- Illegal instruction
- Exceeding array boundaries

When an exception occurs, Essbase displays an error message on the server console. The message states "`Exception occurred`," followed by information about the name and location of the appropriate exception error log. The **Process Type** field in the log header indicates whether the exception occurred in the application or on the server.

Error Log Information Contents

The exception error log provides the following information. The information is the same in the application log as in the server log except where noted.

- Exception information
- Machine registers
- Stack and trace dumps
- Application-wide configuration
- Operating system resources
- System-wide configuration
- `ESSBASE.CFG` values
- License information (server log only)
- Client request activity

- Application information (application log only)
  - File information
  - Database information
  - Database statistics
- Address of the exception pinpoints where the crash occurred

Error Log Names and Locations

The Essbase Exception Handler writes the information into the Exception Error Log on the local disk in an ASCII file as follows:

- If the server crashed, the report is written in the directory specified by `ARBORPATH`; for example, `D:\ESSBASE\log00001.XCP`.
- If the application crashed and the application name is unknown, the report is written into the `APP` directory in the directory specified by `ARBORPATH`; for example, `D:\ESSBASE\APP\log00001.XCP`.
- If the application crashed and the application name is known, but the database name is unknown, the report is written to the appropriate application directory; for example, `D:\ESSBASE\APP\SAMPLE\log00001.XCP`.
- If the application crashed and both the application and database names are known, the report is written to the appropriate database directory; for example `D:\ESSBASE\APP\SAMPLE\BASIC\log00001.XCP`.
- If the exception error generates a trace (`.TRC`) file, the Essbase Exception Handler stores the file in a dump directory that you must define (see "Creating a Directory for Trace Files" on page 45-32).

The Exception Error Log filename is normally `log00001.XCP`. The `ExceptionLogOverwrite` parameter in the `ESSBASE.CFG` file (see the online *Technical Reference* in your `DOCS` directory) controls whether Essbase overwrites the log file.

- When `ExceptionLogOverwrite` is TRUE (the default), Essbase overwrites the previous error log report with the new one.
- When `ExceptionLogOverwrite` is FALSE, Essbase creates a new file instead of overwriting the previous one.

  Subsequent reports are numbered sequentially; for example, if `log00001.XCP` exists, the next report has the filename `log00002.XCP`, the next has `log00003.XCP`, and so on.

The Application Event Log File

Essbase records all application-related activity, including calculations and database restructuring, in a log file on the server. The log file resides in the application directory; for example, `\ESSBASE\APP\`*`application_name`*. It is called *`application_name`*`.LOG`, where *`application_name`* is the name of the current application.

Viewing the Application Log File

The Application Manager provides a facility to view the log file. To view the log file:

1. Connect to the appropriate server, if you have not done so (choose Server | Connect).
2. From the Application Desktop window, select the application whose log file you want to review. Applications are listed in the **Applications** list box.
3. Choose Application | View Event Log. The **View Log File** dialog box appears.

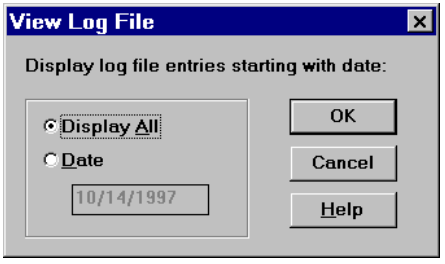

Figure 45-13, View (Application) Log File Dialog Box

4. You can view the entire log file or view only those log entries beyond a specific date. To view the log from a specific date to the present, click the **Date** option and enter the starting date for log file entries.

5. Click OK to view the log. The **Log Viewer** appears, as shown in Figure 45-14. It allows you to scroll through all recorded activities.

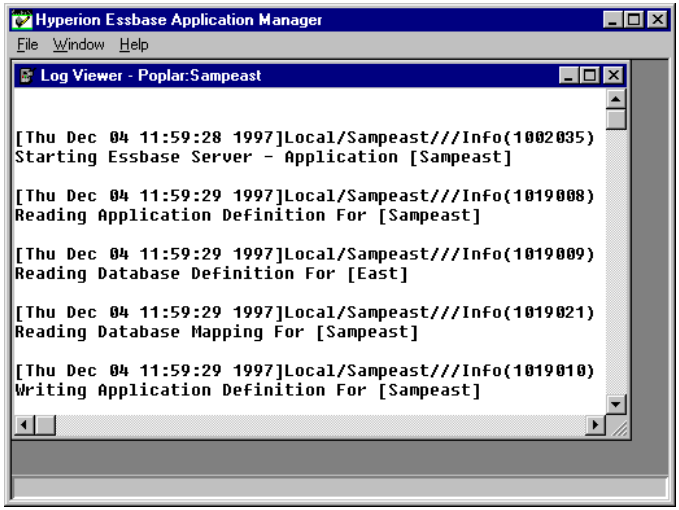

Figure 45-14, A Sample Application Log

Note: Any activities that take place while you review the log are not reflected in the **Log Viewer**; the **Log Viewer** only presents a snapshot of the log file *at the time of your request.*

6. To close the viewer, choose the File | Close.

Deleting the Application Log File

The log file uses disk space on the server. As additional activities are recorded, the log file grows in size. Occasionally, you may need to delete a log file before it grows too large.

To have Essbase automatically clear out the log file whenever a message is written to it, enter `CLEARLOGFILE TRUE` in the `ESSBASE.CFG` server file. When `CLEARLOGFILE` is `FALSE` (the default setting), Essbase appends new messages without overwriting existing messages. For information about the `ESSBASE.CFG` file and the syntax to use for this setting, see the online *Technical Reference* in your `DOCS` directory.

To delete the log file:

1. Connect to the appropriate server, if you have not done so (choose Server | Connect).
2. From the Application Desktop window, select the application whose log file you want to delete. Applications are listed in the **Applications** list box.
3. Choose Application | Delete Event Log.
4. Essbase prompts you to confirm the deletion. Click Yes to delete the log.

   You might want to back up the log file before deleting it. You can store the log file on any disk, diskette, or tape that has sufficient available storage space. You must back up the log file from the Server console.

Warning: CLEARLOGFILE affects all applications on your server.

These menu commands are available only if you have at least Application Designer privileges for this application.

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

Note: The DELETELOG command deletes only the log file for the application you specify.

Performance Considerations

Because the application log file grows in size as activities are recorded, you may want to control which activities are logged in the file.

For example, the TIMINGMESSAGES parameter in the `ESSBASE.CFG` file lets you control whether Essbase logs the duration of each spreadsheet and report query in the application log file. If this parameter is disabled, Essbase does not log query duration times in the log file. In addition to controlling the size of the application log file, setting TIMINGMESSAGES to FALSE prevents Essbase from having to communicate with the operating system to obtain query start and finish times. As a result, query execution time may improve in environments with many concurrent users. For information about other `.CFG` parameters that may affect performance or log file size, see the online *Technical Reference* in your `DOCS` directory.

The Server Event Log File

All server-related activity is recorded in a log file on the server. The log file resides in the ARBORPATH directory and is called ESSBASE.LOG. This log file records information pertaining to all applications in the server.

Viewing the Server Log File

The Application Manager provides a facility to view the server event log file. To view the log file:

1. Connect to the server whose log file you want to review, if you have not done so (choose Server | Connect).
2. Choose Server | View Event Log. The **View Log File** dialog box appears:

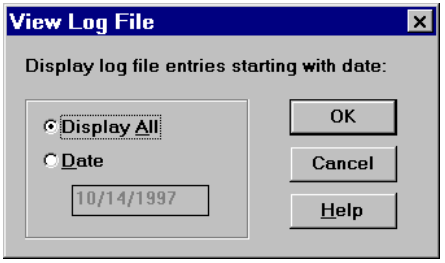

Figure 45-15, View (Server) Log File Dialog Box

3. You can view the entire log file or view only those log entries beyond a specific date. To view the log from a specific date to the present, click the **Date** option and enter the starting date for log file entries.
4. Click OK to view the log. The **Log Viewer** shown in Figure 45-16 appears. It allows you to scroll through all recorded activities.

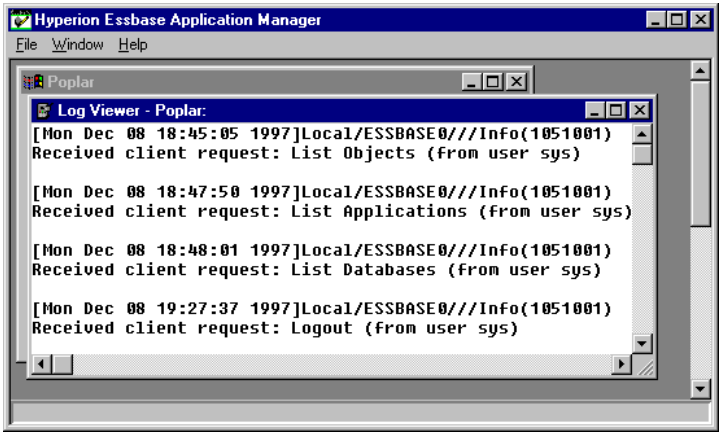

Figure 45-16, A Sample Server Log

Note: Any activities that take place while you review the log are not reflected in the **Log Viewer**; the **Log Viewer** presents a snapshot of the log file *at the time of your request*.

Only users with Supervisor authority can view the Server Log file.

5. To close the viewer, choose File | Close.

Deleting the Server Log File

The log file uses disk space on the server. As additional activities are recorded, the log file grows in size. Occasionally, you may need to delete a log file before it grows too large.

To have Essbase automatically clear out the log file whenever a message is written to it, enter `CLEARLOGFILE TRUE` in the `ESSBASE.CFG` server file. When `CLEARLOGFILE` is `FALSE` (the default setting), Essbase appends new messages without overwriting existing messages. For information about the `ESSBASE.CFG` file and the syntax to use for this setting, see the online *Technical Reference* in your `DOCS` directory.

To delete the log file:

1. Connect to the server whose log file you want to delete, if you have not done so (choose Server | Connect).
2. Choose Server | Delete Event Log.

3. Essbase prompts you to confirm the deletion. Click Yes to delete the log.

Note: You might want to back up the log file before deleting it. You can store the log file on any disk, diskette, or tape that has sufficient available storage space. You must back up the log file from the Server console.

Only users with Supervisor authority can view the Server Log file.

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

Creating a Directory for Trace Files

On rare occasions, an internal Essbase diagnostic utility may generate a trace (`.TRC`) file when an exception error occurs. When this file is generated, Essbase creates an application event log file entry that contains the trace file name and location, and indicates an error number identifying the exception. To capture trace files, you need to identify a directory for Essbase to store them in.

You have two choices:

- Define a location using an `ARBORDUMPPATH` environment variable. Use this option if you have limited space on your main Essbase server machine.
- Create a new `\ARBORPATH\DUMP` directory on your Essbase server machine. Use this option if you want to store trace files on your main Essbase server machine.

Note: Both methods are server-wide settings and only need to be defined once for each Essbase server installation. If you define `ARBORDUMPPATH` and also create a `DUMP` directory, Essbase uses the `ARBORDUMPPATH` directory to store your trace files.

Defining an ARBORDUMPPATH Environment Variable

To define the `ARBORDUMPPATH` environment variable, decide where you want to store the trace files, such as `D:\TRACE`. Then use your operating system to define the variable:

On Windows NT, Windows 95, or Windows 98: `SET ARBORDUMPPATH=D:\TRACE`

On UNIX: `setenv ARBORDUMPPATH "/trace"`

Be sure that the Essbase server has appropriate access to write files into this directory.

Defining a DUMP Directory

If you don't create an ARBORDUMPPATH environment variable, you need to create an \ARBORPATH\DUMP directory on your main Essbase server machine.

To create a new DUMP directory on Windows NT, Windows 95, or Windows 98:

1. Change directory to your Essbase %ARBORPATH% directory.

   For example: `CD C:\ESSBASE`

2. Create a directory called DUMP.

   For example: `MD DUMP`

To create a new dump directory on UNIX:

1. Change directory to your Essbase $ARBORPATH directory.

   For example: `cd /essbase`

2. Create a directory called dump.

   For example: `mkdir essbase/dump`

Trace File Notification

When an exception error occurs that generates a trace (.TRC) file, the application event log file indicates that the file was created and identifies its location (see "Error Log Names and Locations" on page 45-26).

If you have an ARBORDUMPPATH environment variable or an \ARBORPATH\DUMP directory, Essbase creates a trace file named with a long string of numbers separated by underscores, ending with .TRC.

Note: If your operating system doesn't support long file names, the .TRC file name might be truncated.

If you have no ARBORDUMPPATH or \ARBORPATH\DUMP directory, no trace files are created. You can still benefit by reporting the error number from the application log file to the Technical Support department of your software provider.

The Outline Change Log File

You can set Essbase to provide an outline change log that saves detailed modification information to an ASCII text file. You can review the outline change log at any time to see a history of all changes made to an outline since the time it was originally created. This information can be useful to you in the event you ever need to roll back an outline to a previous version. To activate the outline change log, add the following to your `ESSBASE.CFG` file:

```
OUTLINECHANGELOG TRUE
```

Each Essbase database contains a separate outline change log file in the same location as the database. Essbase names the log file with the same name as the corresponding database, followed by the extension `.OLG` (*`database_name`*`.OLG`). The file is stored in the database directory on the server.

The data format of the outline change log file is as follows:

- Date and time of outline modification
- Name of the user who made the change
- Type of change the user made
- Details describing the type of change made

Outline Change Detail that Appears in the Log

When a user makes and saves changes to an outline, Essbase logs modification information into the outline change log as an individual record. Each change record begins and ends with the identifying information shown in Figure 45-17:

```
[Begin Outline Change for app0/Db1, Thu Sep 19 11:29:54, 1998, By User1]

Added new dimension "Overseas"
dimension type - sparse
dimension tag - (none)
Left sibling - None
Status - Store Data
Level Number - 0
Generation Number - 1

[End Outline Change for app0/Dbl, Thu Sep 18 19:00:25, 1998, by User1]
[Begin Outline Change for app0/Dbl, Fri Sep 19 11:29:54, 1998, By User1]

Number of member changes for the dimension "Year" : 2

Added new member "day1" to "Jan" :
Left sibling - None
Unary calc Symbol - Add
Level Number - 0
Generation Number - 4

Added new member "day2" to "Jan" :
Left sibling - "day1"
Status - Store Data
Unary calc Symbol - Add
Level Number - 0
Generation Number - 4

[End Outline Change for app0/Db1, Fri Sep 19 11:29:56, 1998, By User1]
```

Figure 45-17, Sample Outline Change Log

The detailed change information the outline change log provides depends on the type of change the user makes to the outline, as detailed in Table 45-2. See "Impact of Common Outline Changes" on page 39-18 for information on the restructuring impact of each type of outline change.

Table 45-2, Outline Change Log Contents

Type of Change	Information Included
Add a dimension	• Name of dimension • Type of dimension (dense or sparse) • Dimension tag (if any) • Name of dimension's left (immediately preceding) sibling[1] • Dimension's level number • Dimension's generation number
Delete a dimension	• Name of dimension
Update a dimension	• Name of dimension • Dimension tag • Dimension type (dense or sparse) • Level name changes (if applicable) • Generation name changes (if applicable)
Rename a dimension	• Old name of dimension • New name of dimension
Move a dimension to a new position	• Name of dimension • Old location, including dimension's left (immediately preceding) sibling[1] • New location, including dimension's left (immediately preceding) sibling[1]
Add a member to a dimension	• Name of new member or members • Unary calc symbol for member • Level number of member • Generation number of member • Status of member (Store, Share) • Member alias (if applicable) • Member's Account Type • Member's user-defined attributes (if applicable) • Calc formula for member
Update a member of a dimension	• Name of member updated • Member attributes that were updated

Table 45-2, Outline Change Log Contents

Type of Change	Information Included
Rename a member of a dimension	• Old name of member • New name of member
Move a member of a dimension to a new position	• Name of member moved • New location • Names of parent and left sibling[1] in new location

1. The outline change log program reads outline information from left to right; if you are looking at an outline, *left sibling* is sibling directly above (to left of) newly added dimension or member. This rule does not apply if immediately preceding dimension or member is a *parent*. If a newly added (or moved) member is the first child of its parent, or if the member is the first dimension in the outline, outline change log identifies the dimension or member as "None."

Turning the Outline Change Log Function On and Off

To activate the outline change log function, you need to make the following entry in the ESSBASE.CFG file:

```
OUTLINECHANGELOG TRUE
```

After you have turned the function on, Essbase updates the outline change log automatically each time a user makes changes to an Essbase database outline.

Note: During a restructure, Essbase holds outline change information in memory until all updates have been made to the outline change log. Turning on the outline change log might, therefore, affect your restructure performance. "Restructuring Considerations" on page 39-17 discusses other conditions that affect restructure performance.

If you want to disable changes from being made to the outline change log file or to remove the log from the ESSBASE.CFG file, change the setting in the ESSBASE.CFG file to:

```
OUTLINECHANGELOG FALSE
```

Setting the Outline Change Log File Size

The default file size assigned for the outline change log is 64000 bytes. When the file reaches this size, Essbase copies the contents of the file to a backup file with the same name as the outline change log file, but with an .OLB extension (*database_name*.OLB). The outline change log is cleared, and any additional log entries are written to the outline change log file. Each time the outline change log file reaches its maximum file size, Essbase replaces the backup file with a backup of the current outline change log. This cycle repeats as Essbase continues to add log entries to the outline change log.

If you want to set the outline change log file size larger than the default of 64000 bytes, you need to make the following entry in the ESSBASE.CFG file:

```
OUTLINECHANGELOGFILESIZE n
```

where *n* is a number interpreted in bytes; for example, entering 20000 sets the file size to 20000 bytes; entering 9216 sets the file size to 9 kilobytes. The minimum file size is 8092 bytes; the maximum is 2 megabytes. The default, minimum, and maximum file sizes for the backup file are the same as those specified for the outline change log file. For example, if you use the OUTLINECHANGELOGFILESIZE setting to change the maximum size of the outline change log file to 2 megabytes, you automatically set the .OLB backup file to the same maximum size.

Locking and Unlocking Objects

Essbase uses a check-out facility for server-based objects to ensure that no more than one user attempts to modify an object at one time. By default, whenever a user opens a server-based object, Essbase locks the object. If a second user attempts to open the same object, a message appears indicating the object is already locked and who has locked it.

Overriding a File Lock

You can override the lock error by unchecking **Lock file** in the Application Desktop window before attempting to open an object. This essentially opens the file in Read-only mode and you cannot save any changes you make.

Unlocking Objects

There are two ways to unlock a previously locked object:

- Save the locked file to the server.
- Choose File | Unlock. This unlocks a file that you previously locked but did not save to the server.

Notes:

- Users can unlock only those objects that they originally locked. A supervisor can unlock files locked by other users.
- File locking is not the same as block locking. For information about how Essbase locks data blocks, see Chapter 41, Ensuring Data Integrity.

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

Monitoring Applications

Each application that is loaded is an open task or process in the operating system. Under Windows NT, Windows 95, and Windows 98 platforms, the application appears in an Essbase server window. On UNIX platforms, the application server is often a background process. When the application starts, `ESSBASE.EXE` (`ESSBASE` on UNIX systems) starts the `ESSSVR` process.

The application server records its activities, such as writing data to an application log file, *`appname`*`.LOG`, in the `\ESSBASE\APP\`*`appname`* directory. You can open and view this text file when you need to troubleshoot problems or view application activity. On Windows NT, Windows 95, and Windows 98, you can also view application activities as they occur in the Essbase server window. On UNIX, you can route all messages to a file with the `tail -f` *`logfile`* command, where *`logfile`* is the name of a file that you specify.

On Windows NT, Windows 95, and Windows 98 platforms, when the Agent starts an application, a new icon appears on the desktop, as shown in Figure 45-18.

Figure 45-18, Essbase Application Server Icon

On Windows 95 and Windows 98, a new icon appears in the Taskbar.

You can double-click the icon to view the server window.

Server Error Message Categories

Table 45-3 provides error message categories for each error number range shown in the first column. When you receive an error message, use this table to look up the Essbase component that the error is related to.

Table 45-3, Server Error Message Categories

Error Message Number Range	Essbase Component that Generated Error
1001000-1001999	Report Writer Extractor
1002000-1002999	General Server
1003000-1003999	Storage Manager
1004000-1004999	Not currently used
1005000-1005999	Backup/Export/validate
1006000-1006999	Data cache
1007000-1001999	Member structure
1008000-1008999	System calls/portable layer/ASD/Agent
1009000-1009999	Restoring ASCII data
1010000-1010999	Internal (block numbering)
1011000-1011999	Internal (utilities)
1012000-1012999	Calculator
1013000-1013999	Requestor
1014000-1014999	Lock Manager
1015000-1015999	Alias Table
1016000-1016999	Report Writer

Table 45-3, Server Error Message Categories

Error Message Number Range	Essbase Component that Generated Error
1017000-1017999	Currency
1018000-1018999	Message database
1019000-1019999	Database objects
1020000-102999	Spreadsheet Extractor
1021000-1021999	SQL Interface
1022000-1022000	Security
1023000-1023999	Partitioning
1024000-1024999	Query Extractor
1030000-1030999	Application Program Interface (API)
1040000-1040999	Network
1041000-1041999	Network—Named Pipes
1042000-1042999	Network—TCP
1050000-1050999	Agent
1060000-1060999	Outline API
1070000-1070999	Index cache
1080000-1080999	Transaction manager
1090000-1090999	Rules file processing
1100000-110999	Grid API
1110000-1110999	Hyperion Web Gateway (HWG)
1140000-1140999	Linked Reporting Objects

Chapter 46 Working with Essbase Files and Cross-Platform Environments

This chapter introduces the file types Hyperion Essbase uses, describes operations you can perform on Essbase files, and discusses some of the issues you face when working with Essbase files on different platforms.

This chapter contains the following sections:

About Essbase File Types

The following table lists the file types Essbase uses to store applications and databases:

Table 46-1, Essbase File Types for Applications and Databases

File Extension	Description
`alg`	Spreadsheet audit historical information
`apb`	Backup of application file
`app`	Application file, defining the name, location, and other application settings
`arc`	Archive file
`atx`	Spreadsheet audit transaction
`chg`	Outline synchronization change file
`csc`	Essbase calc script
`db`	Database file, defining the name, location, and other database settings
`dbb`	Backup of database file
`dbf`	dBASE data file

Table 46-1, Essbase File Types for Applications and Databases

File Extension	Description
ddb	Partitioning definition file
ddm	Temporary Partitioning file
ddn	Temporary Partitioning file
esm	Storage Manager file, containing control information used for database recovery
esr	Temporary database root file
esn	Temporary Storage Manager file
ind	Essbase index file
inn	Temporary Essbase index file
log	Server or application log file
lro	Linked Reporting Object file that is linked to a data cell
lst	Cascade table of contents or list of files to back up
mdx	dBASE multiple index file
ocl	Database change log
ocn	Incremental restructuring file
oco	Incremental restructuring file
olb	Backup of outline change log
olg	Outline change log
otl	Essbase outline file
otm	Temporary Essbase outline file
otn	Temporary Essbase outline file
oto	Incremental restructuring file
pag	Essbase database data (page) file
pan	Temporary Essbase database data (page) file
rep	Essbase report script
rul	Essbase data load rules file

Table 46-1, Essbase File Types for Applications and Databases

File Extension	Description
scr	Essbase ESSCMD script
sel	Saved member select file
str	Exported Essbase outline file
tct	Essbase database transaction control file
tcu	Temporary database transaction control file
trc	Trace file, created only if a dump file exists in the Essbase directory and server crash occurs
txt	ASCII text file, such as a data file to load or a text document to link as a linked reporting object
wk4	Lotus 1-2-3 spreadsheet file
xcp	Exception error log file
xls	Microsoft Excel spreadsheet file

The following table lists the types of Essbase files stored in the `\ESSBASE\BIN` directory:

Table 46-2, Essbase File Types in the \ESSBASE\BIN Directory

File Extension	Description
12a	Lotus 1-2-3 Spreadsheet Add-in
bak	Backup of security file
bnd	Microsoft ODBC file for SQL server installation using a DB2 database
cfg	Essbase configuration file
cnt	Online help contents file
cpl	Microsoft ODBC driver for Windows NT and Windows 95 or 98
dll	Microsoft Windows Dynamic Link Library
exe	Executable file
hlp	Online help file

Table 46-2, Essbase File Types in the \ESSBASE\BIN Directory

File Extension	Description
ico	Icon bitmap image
lck	Essbase lock file
lic	License information file for ODBC
mdb	Message database file
sec	Security file
sl	HP-UX shared library file
so	Solaris shared library file
xll	Microsoft Excel Spreadsheet Add-in

The following table lists the types of Essbase files stored in the \ESSBASE\API sub-directories:

Table 46-3, Essbase File Types in the \ESSBASE\API Directory

File Extension	Description
a	UNIX static library file
bas	Microsoft Visual Basic program source file, containing header definitions for the API
h	C/C++ header file, containing header definitions for the API
lib	C/C++ program library
lwp	LAN Workplace network library
np	Named Pipes network library
tcp	TCP/IP network library

Operating on Applications, Databases, and Related Objects

The Essbase Application Manager lets you copy, rename, delete, and lock:

- Applications
- Databases
- Related objects, such as outlines, calc scripts, report scripts, rules files, and data sources

Essbase stores each application as a collection of files in the Essbase application directory (for example, `\ESSBASE\APP\`*`appname`*`)` on the server. Essbase stores each database as a collection of files on the server's database directory (for example, `\ESSBASE\APP\`*`appname`*`\`*`dbname`*).

Files that are related to applications and databases are called *objects*. The common types of objects are outlines, calc scripts, report scripts, rules files and data sources. You typically store objects on the server in the *`appname`* or *`dbname`* directory or on the client machine under the `\ESSBASE\CLIENT\`*`appname`* or *`dbname`* directory.

In addition to the standard object types, Essbase also supports the following objects:

- Linked reporting objects are files or cell notes associated with specific data cells in a database. For more information, read Chapter 11, Linking Objects to Your Essbase Data.
- Visual Basic programming objects (`.OCX`) for creating OLAP-aware modules. This feature is available with Hyperion Objects, licensed separately.

Copying an Application

When you copy an application, Essbase copies all files associated with the application to the destination application. Before copying, make sure you have enough disk space to contain a full copy of the application, databases, and related files.

To copy an application in the Application Manager:

1. Connect to the server on which the application resides.
2. Select the application from the Application Desktop window.
3. Choose Application | Copy. The following dialog box appears:

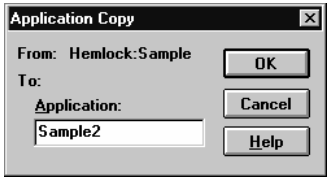

Figure 46-1, Application Copy Dialog Box

4. Enter the name of the new application in the **Application** text box.
5. Click OK. Essbase creates a new application and database directory structure on the server under the `\ESSBASE\APP` directory.

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

Renaming an Application

When you rename an application, all objects within the application with the same name as the application remain untouched.

To rename an application in the Application Manager:

1. Connect to the server on which the application resides.
2. Select the application from the Application Desktop window.
3. Choose Application | Rename. The following dialog box appears:

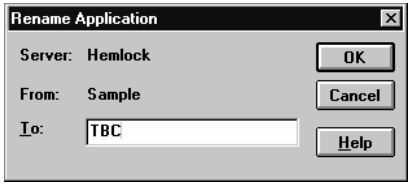

Figure 46-2, Rename Application Dialog Box

4. Enter the new name for the application in the **To** text box. The name can be up to 8 alphanumeric characters.
5. Click OK. Essbase renames the application and directory name under the `\ESSBASE\APP` directory.

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

Deleting an Application

When you delete an application, all objects within the application are also deleted. The `\ESSBASE\APP\`*`appname`* directory and all files located in the directory are deleted.

To delete an application in the Application Manager:

1. Connect to the server on which the application resides.
2. Select the application from the Application Desktop window.
3. Choose Application | Delete. The following dialog box appears:

Figure 46-3, Confirm Delete Dialog Box for Applications

4. Click Yes or No.

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

Copying a Database

When you copy a database, all files associated with the database are also copied to the destination application. Before copying, make sure you have enough disk space to contain a full copy of the databases and related files.

To copy a database:

1. Connect to the server on which the database resides.
2. Select the application that contains the database from the Application Desktop window.
3. Select the appropriate database in the selected application.
4. Choose Database | Copy. The following dialog box appears:

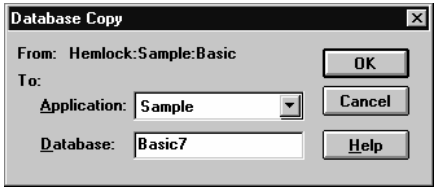

Figure 46-4, Database Copy Dialog Box

5. To copy the database to another database within the same application, type a new database name in the **Database** text box and click OK.
6. To copy the database to another database in a different application, the other application must already exist. Choose the application name from the **Application** list box. Then type the name of the new database and click OK.

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

Renaming a Database

When you rename a database, the outline file (`.OTL`) is also renamed. All other objects in the database with the same name as the database are unchanged.

To rename a database:

1. Connect to the server on which the application resides.
2. Select the application that contains the database from the Application Desktop window.
3. Select the appropriate database in the selected application.
4. Choose Database | Rename. The following dialog box appears:

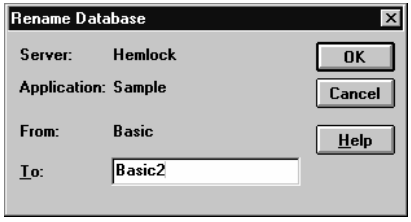

Figure 46-5, Rename Database Dialog Box

5. Enter the new name for the database in the **To** text box.
6. Click OK. Essbase renames the database and directory name under the `\ESSBASE\APP\`*`appname`* directory.

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

Deleting a Database

When you delete a database, all objects within the database are also deleted. The `\ESSBASE\APP\`*`appname`*`\`*`dbname`* directory and all files located in the directory are deleted.

To delete a database:

1. Connect to the server on which the application that contains the database resides.
2. Select the application that contains the database from the Application Desktop window.
3. Select the appropriate database in the selected application.
4. Choose Database | Delete. The following dialog box appears:

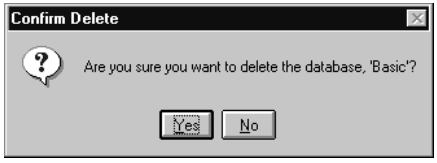

Figure 46-6, Confirm Delete Dialog Box for Databases

5. Click Yes or No.

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

Copying Objects

Objects related to databases or applications are outlines, calc scripts, report scripts, rules files, and data sources. You can copy any of these objects, except outlines, to another application, database, server, or client location.

To copy an object:

1. Connect to the server on which the object resides.
2. Select the application from the Application Desktop window.
3. Select the appropriate database in the selected application.
4. In the Application Desktop window, click the button corresponding to the object file.
5. Choose File | Copy. The following dialog box appears:

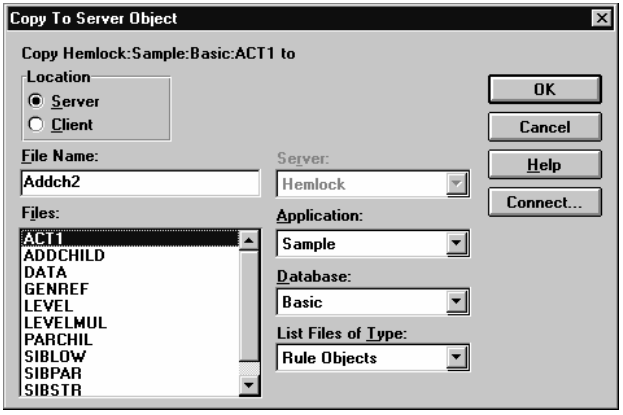

Figure 46-7, Copy to Server Object Dialog Box

6. Enter the name in the **File Name** text box.
7. Select the server, application, and database names corresponding to the destination of the new object.
8. Click OK.

Renaming Objects

You can rename any object, except outlines. Outlines always have the same name as a database, so you need to rename the database to rename the outline.

To rename an object:

1. Connect to the server on which the object resides.
2. Select the application from the Application Desktop window.
3. Select the appropriate database in the selected application.
4. In the Application Desktop window, click the button corresponding to the object file.
5. Choose File | Rename. The following dialog box appears:

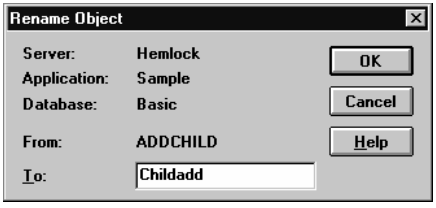

Figure 46-8, Rename Object Dialog Box

6. Enter the name in the **To** text box.
7. Click OK.

Deleting Objects

You can delete any object, except outlines. Outlines are a required part of a database, so you need to delete the database to delete the outline.

To delete an object:

1. Connect to the server on which the object resides.
2. Select the application from the Application Desktop window.
3. Select the appropriate database in the selected application.
4. In the Application Desktop window, click the button corresponding to the object file.
5. Choose File | Delete. The following dialog box appears:

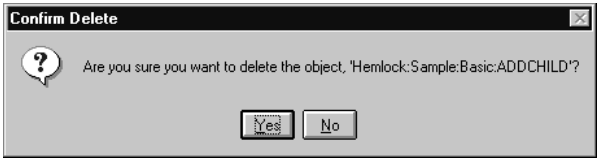

Figure 46-9, Confirm Delete Dialog Box for Objects

6. Click Yes or No.

Locking and Unlocking Objects

Essbase uses a check-out facility for server-based objects to ensure that no more than one user attempts to modify an object at one time. This section describes how to lock and unlock objects.

Note: Locking objects is not the same as locking data blocks. The Essbase Storage Manager handles locking for data blocks, but not for objects. See Chapter 41, Ensuring Data Integrity, for information about locking data blocks.

Locking Objects

By default, whenever a user opens a server-based object, Essbase locks the object. If a second user attempts to open the same object, a message appears indicating the object is already locked and who has locked it.

You can avoid the lock error by unchecking **Lock file** in the Application Desktop window before attempting to open an object. This essentially opens the file in Read-only mode, and you cannot save any changes you make.

Unlocking Objects

There are two ways to unlock a previously locked object:

- Save the locked file to the server.
- Choose File | Unlock. This unlocks a file that you previously locked but did not save to the server.

Note: Users can only unlock objects that they have locked themselves. A user with Supervisor privileges can unlock any object.

Porting Applications Across Platforms

The Hyperion Essbase OLAP Server runs on Windows NT, Windows 95, DEC Alpha NT, and the following UNIX platforms: Solaris, HP-UX, and AIX. The *Installation Notes* book describes how to install and configure the Essbase server on each provided platform.

After you install an application, you might want to port the application to another server which may be running a different operating system. This section describes how to port an installed Essbase application to another Essbase server machine.

Porting Essbase applications across servers involves these steps:

1. Identifying compatible files.
2. Checking file names.
3. Transferring compatible files.
4. Reloading the database.

Indentifying Compatible Files

If you are porting an Essbase application to a server running a different operating system, you need to identify which Essbase files are compatible with the new operating system.

The following file types are compatible between operating systems:

- ASCII text files. The Essbase ASCII text files are calc scripts (`.csc`) and report scripts (`.rep`). Also, your data files might be ASCII text files.
- Data load rules files. These are binary files, but they are compatible between operating systems. Data load rules files have the extension `.rul`.
- Outline files. These are binary files, but they are compatible between operating systems. Outline files have the extension `.otl`.

The following file types are incompatible between operating systems and need to be redefined or reloaded on the new server:

- Database files with the extensions `.db` and `.dbb`
- Data files with the extension `.pag`
- Index files with the extension `.ind`
- Security files with the extension `.sec`
- Application files with the extensions `.app` and `.apb`
- Storage Manager files with the extension `.esm`

Note: If you are using the Linked Reporting Objects feature, you need to re-link any files or cell notes on the new server. For more information, see Chapter 11, Linking Objects to Your Essbase Data.

Checking File Names

When transferring files to a UNIX system, you need to be aware of the case of file names. UNIX is a case-sensitive operating system and files are recognized only if they have the correct case. For example, in certain ESSCMD operations, you need to specify a file name, and the file name must be entered with the correct case.

The Essbase system files use the following naming conventions on UNIX systems:

- Executable files have no extension and are uppercase (`ESSBASE`, `ESSCMD`).
- Static library files have the file extension `.a` and are in lowercase (`libessnet.a`).
- Shared library files have the file extension `.sl` on HP-UX, `.so` on Solaris, and `.a` on AIX. These file names are in lowercase (for example, `libesscur.sl`).
- Security files have the file extension `.sec` and are in lowercase (`essbase.sec`).
- Message database files have the file extension `.mdb` and are in lowercase (`essbase.mdb`).
- Online help files have the file extension `.hlp` and are in lowercase (`esscmd.hlp`).

All Essbase data and application files on UNIX systems are capitalized with *proper* case: the first letter is uppercase, and the remaining letters are lowercase. The following table gives examples of names for different file types:

Table 46-4, File Naming Examples for UNIX

File Type	Example
Database files	`Mydb.db`
Data files	`Mydb.pag`
Index files	`Mydb.ind`
Outline files	`Mydb.otl`
Data load rules files	`Atlanta.rul`
Data files to load	`Atlanta.txt`
Calc scripts	`Mycalc.csc`
Report scripts	`Myrepo.rep`
Archive files	`Mydb.arc`
Application log files	*`Appname`*`.log`

Table 46-5 lists several examples of valid and invalid file names on UNIX systems:

Table 46-5, Valid and Invalid File Names on UNIX

Valid File Names	Invalid File Names
`Model.csc`	`MODEL.CSC`
`Monthly.rep`	`Monthly.Rep`
`Forecast.otl`	`forecast.otl`
`Actuals.rul`	`AcTuAlS.rUl`
`My_File.txt`	`My_File.Txt`

Note: The Essbase server does not allow long file names for applications, databases, calc scripts, reports and other database files. All file names for objects you create must conform to the standard 8.3 convention.

Transferring Compatible Files

If the two servers are connected, you can create the application and database directories on the new server and use either FTP (File Transfer Protocol) or Application Manager to transfer the compatible application files. If the servers are not connected, you need to redefine server information on the new server before reloading the database.

Transferring Using FTP

Using FTP, you can transfer files directly between operating systems. You should transfer only the files compatible between operating systems, and you should transfer the files in binary mode.

If you have files on a UNIX server with the wrong case, the Application Manager can see these files but cannot open them. After you use FTP to transfer files, you should rename the files on the server to ensure they are capitalized with proper case. Alternatively, you can rename the file using FTP when you transfer the file:

```
ftp>put oldfile Newfile
```

Transferring Using Application Manager

Using Application Manager, you can transfer files from the client machine to the server by using File | Save As. For example, you can connect to a Windows NT server, open an outline, and then save it to a UNIX server. The file types you can transfer from the Application Manager are outline files, report scripts, calc scripts, and rules files.

When you save a file from Application Manager to a UNIX server, proper case is automatically applied regardless of how you type it.

Redefining Server Information

If the server you are porting to is not connected to the existing server, you need to redefine some Essbase server information on the new server.

To redefine server information, follow these steps:

1. Create users and specify their privileges. To do this, use Application Manager on the new server. See Chapter 16, Managing Security at Global and User Levels, for help.
2. Create the applications and databases that you want to port. To do this, use Application Manager on the new server. See Chapter 7, Creating Applications and Databases, for help.
3. Copy the outline files (`.otl`) for the databases you want to port from the old server to the same directory location on the new server. Make sure the application is not running while you copy these files.
4. Copy compatible files from the old server to the new server. For help, see "Indentifying Compatible Files" on page 46-16.
5. Reload the database. For help, see "Reloading the Database" on page 46-19.

Reloading the Database

Database files, such as `.db`, `.pag`, `.esm`, and `.ind`, are not compatible between operating systems. If you port your application to a server on a different operating system, you need to repopulate the database by reloading the data from a data file and data load rules file (if applicable). One way you can do this is to export the data to an ASCII text file, transfer the text file to the new server, then use it to load data.

Chapter 47 Backing Up Data and Recovering Databases

This chapter tells how to recover a database, how to back up a database, and other ways Hyperion Essbase protects your data.

This chapter includes the following sections:

If you are migrating from a previous release of Essbase, see the *Start Here* booklet.

Backing Up Your Database

A key part of a database maintenance routine is making regular backups of your Essbase data. The most common method of backing up is *archiving*. Archiving makes copies of the Essbase data files exactly as they exist on your system. You can also back up data by *exporting* it. Exporting makes a copy of the data in an ASCII text format. This section describes both backup methods.

Archiving Data

To archive your data, use the Essbase BEGINARCHIVE and ENDARCHIVE commands along with a third-party backup utility of your choice. The BEGINARCHIVE command places your database in Read-only mode and provides a list of files to back up. Users can remain logged in, but they will not be able to modify data during the archive procedure.

Notes:

- If you back up your data without using BEGINARCHIVE, make sure all Essbase applications are closed and all users are logged out during the backup procedure. Otherwise, you risk corrupting the database.
- The file list generated by BEGINARCHIVE includes Linked Reporting Object files and Partitioning database files, if applicable. For more information on Linked Reporting Objects, see Chapter 11, Linking Objects to Your Essbase Data. For information on Partitioning, see Chapter 6, Designing Partitioned Applications.

To archive your data, follow these steps.

1. Issue the BEGINARCHIVE command in ESSCMD.

   BEGINARCHIVE does the following:

   - Commits any modified data blocks.
   - Switches the database into Read-only mode.
   - Creates a file containing a list of files that need to be backed up. By default, the file is called `archive.lst`. See the BEGINARCHIVE command in the online *Technical Reference* in your `DOCS` directory for more information on `archive.lst`.

   If a user tries to modify data during the archive procedure, an error message informs the user that data is in Read-only mode for archiving. No user can update the database until the ENDARCHIVE command is issued. (See step 3.)

   Read-only mode persists until you issue ENDARCHIVE, even if the database is shut down or restarted.

2. Use a third-party backup utility to back up the files listed in `archive.lst`. These files are described in "Recovering from a Crashed Database" on page 47-4.

   Be sure to back up data on every disk volume Essbase uses. For information about data storage on multiple volumes, see Chapter 39, Introducing the Essbase Storage Manager.

3. Complete the archiving process by issuing the ENDARCHIVE command in ESSCMD.

   ENDARCHIVE does the following:

   - Puts the database into Read-write mode.
   - Re-opens database files in Exclusive mode.

Notes:

- You must issue an ENDARCHIVE command to put the database back into Read-write mode, even if you shut down and restart the database. The ENDARCHIVE command does not restart the database.
- If you try to cancel the backup after issuing BEGINARCHIVE and receive a "can't cancel" message, most likely the system is in the final stage of writing items to the drive and has reached the point where the operation cannot be cancelled.

Exporting Data

You can back up data by exporting instead of archiving. The EXPORT utility copies data to an ASCII text file that you specify; it does not compress the data. The exported file contains data only and does not include control, outline, or security information. The maximum size of the exported file is 2 gigabytes.

Archiving is the recommended method of backing up. However, you might use the EXPORT utility when:

- Transferring data across platforms
- Backing up only a certain portion of the data; for example, level 0 blocks
- You prefer an exported file in text format, rather than binary format

Note: You can also export data by creating reports. For more information on how to do this, see Chapter 38, Copying Data Subsets and Exporting Data to Other Programs.

Using the EXPORT Utility

To export data, choose Database | Export in the Application Manager or issue EXPORT in ESSCMD.

The EXPORT utility lets you specify the amount of data to export:

- All data
- Level 0 blocks only (blocks containing only Level 0 sparse member combinations)
- Data from input blocks only (blocks containing data from a data load)

Reloading Exported Data

When you reload data that has been exported, Essbase marks the data as input data. If you reload data exported from level 0 blocks or input blocks, you must recalculate the database after reloading. When you recalculate the database, Essbase recalculates every data block.

If you export *all* data in a database, then reload, Essbase marks all blocks in the database as input blocks. Consequently, if you try to clear data by choosing Database | Clear Data | Non-Input Blocks from the Application Manager, no data is cleared because the database contains no non-input blocks.

Note: To clean a database after exporting all and reloading, you can run the following calc script:

```
Set ClearUpdateStatus Only;
CalcAll;
```

Recovering from a Crashed Database

After a server interruption such as a crash, Essbase recovers your database, restoring data as it existed at the time of the last successful commit. Essbase also recovers and consolidates free fragments (unused addressable units in the data blocks).

Essbase recovers data as soon as you restart the server after a server interruption. The size of the index determines recovery time.

Only a media failure (faulty disk, disk failure, or head crash) would require you to restore from archived files. See "Archiving Data" on page 47-1.

When Essbase recovers a database, it uses the following files from the database as they existed before the crash or other interruption:

File Name	Description
ESS*xxxxx*.IND	An index file. When data is first loaded, Essbase stores the index in a series of files of up to 2 gigabytes each. Essbase names the first file ESS000001.IND (Ess000001.ind on UNIX platforms) and numbers each index file consecutively to a maximum of ESS99999.IND (Ess99999.ind on UNIX platforms). Other volumes have the same naming conventions (that is, every volume used has a different file named ESS00001.IND or Ess00001.ind).
ESS*xxxxx*.PAG	A data file. When data is first loaded, Essbase stores the data in a series of files of up to 2 gigabytes each. Essbase names the first file ESS000001.PAG (Ess000001.pag on UNIX platforms) and numbers each data file consecutively to a maximum of ESS99999.PAG (Ess99999.pag on UNIX platforms). Other volumes have the same naming conventions (that is, every volume used has a different file named ESS00001.PAG or Ess00001.pag).
databasename.IND	A free fragment file for data and index free fragments.
databasename.ESM	A main root page file.
databasename.TCT	A transaction control table.

Warning: Do not move, copy, modify, or delete any of the files listed above. Doing so could result in data corruption.

What to Expect if a Server Interruption Occurs

The following table lists types of server interruptions and their results.

Table 47-1, Essbase Recovery Handling

Type of Interruption	Result
Power loss on server machine Operating system crash Server stopped with `Ctrl+C` keys	Server stops. When you restart the server, Essbase reverts to the last successfully committed transaction and recovers free fragments.
Operation cannot complete due to system limitations Memory shortage Out of disk space Application stopped with `Ctrl+C` keys	Essbase displays an error message. You might need to allocate more memory or disk space and restart the server.
Server crash.	Server stops. When you restart the server, Essbase reverts to the last successfully committed transaction. Essbase Exception Manager creates error log file of type `.XCP`.

Table 47-2 shows what you need to do if a server interruption occurs during a transaction. How Essbase recovers from these interruptions depends on your transaction isolation level setting (committed or uncommitted access). See Chapter 41, Ensuring Data Integrity, for more information.

Table 47-2, Recovery Procedures for Server Requests

Type of Lock	Recommended Action
Lock (for spreadsheet update)	Issue the Lock command again.
Send (spreadsheet update)	If Essbase did not log a completion message, or if Essbase issued a network timeout error, repeat the last Send operation.
Calculation	Repeat the last calculation. See Part V, Calculating Your Data.
Data load	Repeat the last data load (see Chapter 22, Performing a Data Load), or load the error log file (see Chapter 23, Debugging and Optimizing Data Loads).

Table 47-2, Recovery Procedures for Server Requests

Arithmetic data load (adding to or subtracting from values in the database)	First, make a backup before loading. You can then restore from the backup if a failure occurs during loading. If you have no backup, you can try to compare the values in the data source with those in the database to determine which values must be cleared and reloaded. See Chapter 21, Manipulating Fields Using a Rules File.
Restructure	Repeat the last operation that caused a restructure. Check for temporary files and delete them.

Note: If the UPDATECALC parameter is set to FALSE, Essbase recalculates the entire database if an interruption occurs during a calculation. (The parameter's default value is TRUE.) For more information on UPDATECALC, see the online *Technical Reference* in your `DOCS` directory.

Spreadsheet Update Logging

For extra protection against data loss, and for spreadsheet audit information, Essbase provides a spreadsheet update logging facility. Enable this facility using the SSAUDIT or SSAUDITR parameter in the `ESSBASE.CFG` server file. For information on the `ESSBASE.CFG` file, see online *Technical Reference* in your `DOCS` directory.

SSAUDITR automatically clears update log files after archiving, when you issue the ENDARCHIVE command. SSAUDIT does not remove log files after archiving. See "Archiving Data" on page 47-1 for information on the ENDARCHIVE command.

You can specify SSAUDIT for all databases on the server or for individual databases. See the online *Technical Reference* in your `DOCS` directory for syntax information.

Essbase handles recovery under normal situations. However, sometimes you might want to manually load the spreadsheet update log file. For instance, if you've restored from a recent backup and don't want to lose changes made since the backup was made, or you experience a media failure, you can recover transactions from the update log file. To do so, use the Essbase command-line facility, ESSCMD, from the server console.

The following ESSCMD command sequence loads the update log file:

```
LOGIN hostnode username password
SELECT application_name database_name
LOADDATA 3 filepath:application_name.ATX
EXIT
```

To simplify the process of loading the update log file, prepare a batch file as described in Chapter 43, Performing Interactive and Batch Operations Using ESSCMD.

Notes:

- When SSAUDIT or SSAUDITR is specified, Essbase logs spreadsheet update transactions chronologically. Essbase uses two files:
  - *audit_database_name*.ATX stores each update transaction as a unit.
  - *audit_database_name*.ALG contains historical information: date, time, size, and starting row of each transaction.
- The spreadsheet update log can get quite large; even if you are using SSAUDITR, Essbase only clears the log after you archive data. If spreadsheet update activities are frequent in your application, you may want to manually delete the log file periodically.
- When a database is started after a shutdown, if spreadsheet logging is enabled, Essbase writes the following message to the database log file:

  Starting Spreadsheet Log *volumename**application_directory**application_name**database_name**database_name*.atx For Database *database_name*

  For example:

  Starting Spreadsheet Log \ESSBASE\app\app1\sample\sample.atx for database sample

- To ensure successful spreadsheet update logging, stop and restart your application after:
  - Any operation that causes a restructure. (See "Restructuring Databases" on page 39-12.)
  - Running any of the following database comands:

    CREATEAPP
    CREATEDB
    COPYDB
    RENAMEDB
- Essbase ensures that if you enable spreadsheet logging, updates cannot take place without being logged. If Essbase cannot write to the update log for any reason, Essbase stops the transaction and issues an error message.

SSAUDIT and SSAUDITR are only available from the ESSBASE.CFG file.

Appendix A Performance-Related Settings

You can customize Essbase at the database level, using either the **Database Settings** dialog box in the Application Manager or ESSCMD.

This section contains information on many performance-related Essbase settings. It focuses on general performance rather than, for example, Calculator, data load, or Report Writer performance.

For information on optimizing Calculator performance, see the following chapters:

- Chapter 32, Optimizing Your Calculations
- Chapter 33, Optimizing Your Calculation Using Intelligent Calculation

For information on optimizing data loads, see:

- "Optimizing Data Loads" on page 23-6

For information on optimizing Report Writer performance, see:

- Chapter 37, Optimizing Your Reports.

This appendix provides general-purpose information and does not account for the wide variety of configuration possibilities. For more detailed information on the Storage Manager, see the following chapters:

- Chapter 14, Sizing Your Database
- Chapter 39, Introducing the Essbase Storage Manager
- Chapter 40, Specifying Storage Manager Settings
- Chapter 41, Ensuring Data Integrity

Note: If you are migrating your database, your new database inherits its previous settings instead of the defaults Essbase uses for a new installation. See the *Start Here* booklet for information about migration.

Table A-1 helps you understand what each setting controls and where to adjust it:

Table A-1, Database Settings and Defaults

Setting	Application Manager Setting	ESSCMD Command	Default Value and Comments
Index cache size	Database|Settings, Storage tab	`SETDBSTATEITEM 12`	1 MB (Default and minimum) Recommendation: Start with 10 MB and adjust as needed. See “Sizing Your Index and Data Caches” on page 14-12.
Data cache size	Database|Settings, Storage tab	`SETDBSTATEITEM 5`	3 MB (Default and minimum) Recommendation: Start with 30 MB and adjust as needed. See “Sizing Your Index and Data Caches” on page 14-12.
Index page size	Database|Settings, Storage tab	`SETDBSTATEITEM 13`	1 KB (Default and minimum) Recommendation: Use the default setting. Essbase accepts any value between 1 KB and 8 KB, but only uses values of 1 KB, 2 KB, 4 KB, or 8 KB. Essbase rounds up the value you enter to an acceptable value if necessary.
Isolation level	Database|Settings, Transaction tab	`SETDBSTATEITEM 18`	Uncommitted. See Chapter 41, Ensuring Data Integrity for more information.
Commit blocks	Database|Settings, Transaction tab	`SETDBSTATEITEM 21`	3000 Effective only when Isolation Level is Uncommitted. Notes: • A zero value means no implicit commit; Essbase commits all affected blocks at the end of a transaction. • If either Commit Block or Commit Row has a non-zero value, Essbase commits data when it reaches the first threshold. For example, if Commit Blocks is 10 but Commit Rows is 0, and you load data, Essbase commits after 10 blocks are updated.

Table A-1, Database Settings and Defaults

Setting	Application Manager Setting	ESSCMD Command	Default Value and Comments
Commit rows	Database|Settings, Transaction tab	SETDBSTATEITEM 22	0 (no implicit commit; Essbase commits all affected blocks at the end of a transaction.) Effective only when Isolation Level is Uncommitted. See Note under Commit blocks.
Wait for write access to locked data block	Database|Settings, Transaction tab	SETDBSTATEITEM 20	Indefinite wait (-1 in ESSCMD). Effective only when Isolation Level is Committed.
Pre-image access	Database|Settings, Transaction tab	SETDBSTATEITEM 19	Enabled. Effective only when Isolation Level is Committed.
Disk volumes: volume name	Database|Settings, Storage tab	SETDBSTATEITEM 23 SETDBSTATEITEM 24	If you do not specify this setting, the Storage Manager uses only the volume that ARBORPATH points to, and fills the entire volume as needed. Replaces DISKVOLUMES *volume_name* in ESSBASE.CFG, which is no longer used except for initial migration.
Disk volumes: partition size	Database|Settings, Storage tab	SETDBSTATEITEM 23 SETDBSTATEITEM 24	Unlimited. (Uses all available space on specified volume.)
Disk volumes: file type	Database|Settings, Storage tab	SETDBSTATEITEM 23	Index and Data.
Disk volumes: maximum file size	Database|Settings, Storage tab	SETDBSTATEITEM 23	2 GB
Retrieval buffer size	Database|Settings, General tab	SETDBSTATEITEM 16	10,240 B (10 KB). Note: In Version 4, this setting was specified as REPTKBYTEBUF in ESSBASE.CFG.

Table A-1, Database Settings and Defaults

Setting	Application Manager Setting	ESSCMD Command	Default Value and Comments
Retrieval sort buffer size	Database|Settings, General tab	SETDBSTATEITEM 17	10,240 B (10 KB). ***Note:*** In Version 4, this setting was specified as REPTKBYTESORTBUF in ESSBASE.CFG.
Data compression	Database|Settings, Storage tab	SETDBSTATEITEM 14 SETDBSTATEITEM 15	Bitmap compression enabled. In Application Manager, the same setting lets you choose bitmap or RLE (Run-Length Encoding) compression, or no compression. In ESSCMD, use SETDBSTATEITEM 14 to enable or disable compression, and SETDBSTATEITEM 15 to control compression type.

Glossary

!	*See* bang character.
#Missing	*See* missing data.
Accounts dimension	A dimension type that makes accounting intelligence available. You can only tag one dimension as Accounts, although you do not need to have an Accounts dimension.
administrator	An individual who installs and maintains the Essbase system including setting up user accounts and security. *See also* database administrator, system administrator.
Advanced Interpretation mode	An option in the Hyperion Essbase Spreadsheet Add-in that lets you define your spreadsheet layout through Zoom operations, Retrieval Wizard operations, or by typing data into the sheet. When you construct a free-form report in Advanced Interpretation mode, Essbase interprets the member names and creates a default view based on the location of the labels.
Agent	The process on the server that starts and stops applications and databases, manages connections from users, and handles user-access security. Referred to as ESSBASE.EXE.
Agent log file	A record of actions performed on the Agent.
aggregate	*See* consolidate.
alias	An alternate name for a dimension or member.
alias table	A table storing alias names for a database.
alternate name	*See* alias.
ancestor	A branch member that has members below it. For example, in the Sample Basic database, Qtr2 and Year are ancestors of April.
AND/OR	A logical expression used in calculation or report scripts to identify a subset of members.
API	Application Programming Interface. The Essbase API is a library of functions you can use in a custom C or Visual Basic program to access the Essbase server.

application	A management structure containing one or more Essbase databases and related files that control many system variables such as memory allocation and autoload parameters.
Application Designer	An individual who designs, creates, and maintains Essbase applications and databases.
application log file	A record of user actions performed on an application.
Application Manager	Essbase software that lets you create and maintain Essbase applications.
Application Programming Interface	*See* API.
Application Server	The main server process (ESSSVR) that runs when an application starts.
ARBORPATH	An environment variable that specifies the Essbase root directory.
area	A predefined set of members and values that make up a partition.
arithmetic data load	A data load that performs operations on values in the database, such as adding 10 to each value.
asymmetric report	A report characterized by groups of nested members that differ by at least one member in the nested group. There can be a difference in the number of members or the names of members.
attribute reporting	The process of defining reports based on attributes of the members in the database outline.
bang character	The bang character (!) terminates a series of report commands, and requests information back from the database. You must terminate a report script with a bang character, or you can use several bang characters within the script.
batch file	Operating system file (*.BAT extension) that can call multiple ESSCMD scripts and run multiple sessions of ESSCMD. Batch files handle batch data loads and complex calculations, and can include commands that run report scripts. You can run a batch file on the server from the operating system prompt.
batch processing mode	A method of using ESSCMD that lets you write a batch or script file, which you can use to automate your routine server maintenance and diagnostic tasks. You can write a script or batch file and run it from the command line.
block	The primary storage unit within Essbase. A data block is a multi-dimensional array representing the cells of all dense dimensions.
bottom-up	Calculations performed from the bottom member of the outline to the top member of the outline. Also, data that is copied or moved from a departmental/organizational cube to a consolidated model.

build method	A method used to modify database outlines based on the format of data in data source files.
calc script	*See* calculation script.
calculation	The process of consolidating or running a calc script on a database. *See also* consolidation, calculation script.
calculation script	A text file containing a set of instructions telling Essbase how to calculate a database. Also called a *calc script*.
cascade	The process of creating multiple reports for a subset of member values.
cell	A unit of data representing the intersection of each dimension in a multidimensional database. Also, the intersection of a row and a column in a spreadsheet.
cell note	A text annotation of up to 599 characters for a cell in an Essbase database. Cell notes are a type of linked reporting object.
change log	*See* Outline Change log.
child	A member that has a parent above it in the database outline. A child may have peers (siblings) that exist at the same layer of the database outline.
client	A client interface, such as Hyperion Essbase Spreadsheet Add-in software, a custom API program, or the Hyperion Essbase Application Manager. Also, a workstation connected to a server via a local area network.
client log file	An ASCII file that records all messages, actions, and errors generated by a client.
column	A vertical list of fields.
column heading	A part of a report that lists members across a page. You can define columns that report on data from more than one dimension, which results in nested column headings. All data values in a column have the members listed in the column heading as a common attribute. *See also* nested column headings.
committed access	A Storage Manager Isolation Level setting that affects how Essbase handles transactions. Under committed access, concurrent transactions hold long-term Write locks and yield predictable results.
consolidation	A method of computing the data relationships for all parent/child combinations within a dimension. A consolidation is typically additive, but can be any type of calculation, For example, if the dimension Year consists of the members Qtr1, Qtr2, Qtr3, and Qtr4, its consolidation would be Year. The terms “aggregate” and “roll-up” also describe the consolidation process.

currency conversion	A factor that converts currency values in an Essbase database from one country's currency into another's, optionally without altering the original data.
currency partition	A dimension type that separates local currency members for a base currency defined in your application. Also identifies currency types such as Actual, Budget, and Forecast.
database	A repository of data within Essbase that contains a multidimensional data storage array. Each database consists of a storage structure definition (a database outline), data, security definitions, and optional calculation scripts. An application contains one or more databases.
database administrator	Individual who administers Essbase databases, but may also design, maintain, and create the database.
Database Designer	The highest type of access that can be assigned globally (per database). This type of access allows complete calculate and update access, and the ability to run report and calc scripts.
database filter layer	A layer in the Essbase security plan that defines specific settings for database members down to the cell level.
data block	*See* block.
data cache	A buffer in memory that holds data blocks.
data file	A file containing data blocks; Essbase generates the file on data load and stores it on disk.
data load	The process of populating an Essbase database with data. Loading data establishes actual values for the cells defined by the database's structural outline.
data load rules	A set of operations that Essbase performs on data from an external data source file as it is loaded.
data point	*See* cell.
data source	External data, such as a text file, spreadsheet file, or SQL database, that will be loaded into an Essbase database.
data value	*See* cell, metadata.
DB Designer	*See* Database Designer.
dense dimension	A dimension with a high probability for occupying one or more data points in every combination of dimensions that occurs.
descendant	Any member below a parent.
detail member	*See* leaf member.

dimension	A data category. The Sample Basic database includes such dimensions as Time, Accounts, Product, or Market. In an Essbase database outline, the dimensions represent the highest consolidation level.
dimension build rules	Similar to data load rules, but they also modify the outline based on data in the external data source file.
dirty block	A data block containing cells that have been changed since the last calculation.
disabled user name	A user name which has become inactive, meaning that the user will not be able to log on to the server. User names are automatically disabled when their users exceed limitations specified by a supervisor in the Password Management options for the server settings. A supervisor can choose to disable a user name for other reasons as well.
drill down	*See* Zoom.
dynamic	A method of calculating a member at the moment of data retrieval.
Dynamic Calc And Store members	Members that Essbase calculates when you first retrieve the values. Essbase then stores these values in the database. Subsequent retrievals of Dynamic Calc And Store members do not require calculating.
Dynamic Calc members	Members that Essbase calculates when you retrieve the value. Essbase discards calculated values after the retrieval request is complete.
Dynamic Calculation	A calculation that occurs when you report data on a member that has been tagged as Dynamic Calc. The member's values are calculated at retrieval time instead of being precalculated during batch calculation.
dynamic reference	A pointer to header records in a data source.
Dynamic Time Series	A function that enables you to perform dynamic time-period-to-date reporting.
Dynamic Time Series members	Members that Essbase uses to perform Dynamic Time Series reporting.
ESSCMD	A command-line interface used to perform server operations interactively or through a batch file. *See also* batch file.
EssCell	The Essbase cell retrieve function. An EssCell function is entered into a cell in the Spreadsheet Add-in to retrieve a single database value representing an intersection of specific database members.
extraction commands	A type of Report Writer command that handles the selection, orientation, grouping, and ordering of raw data extracted from the database. These commands begin with the "<" character.

field	A value or item in a data source file that will be loaded into an Essbase database.
file delimiter	One or more characters separating fields in a data source.
filter	A method for controlling access to database cells. A filter is the most detailed level of security, allowing you to define varying access levels different users can have to individual database values.
FlashBack	A Hyperion Essbase Spreadsheet Add-in command that restores the previous database view. This command is similar to a typical Undo command.
formatting command	A type of Report Writer command that allows for customizing the report format and appearance, the creation of new columns, and calculation of columns and rows. These commands are generally contained within curly brace "{}" characters, although some begin with the "<" character. *See also* member-specific report commands.
formula	A combination of operators and functions as well as dimension names, member names, and numeric constants used to calculate relationships between any members of a database.
formula preservation	*See* preserve formulas.
free-form data loading	A method of loading data into the database. Free-form data loading is used when the data source contains enough information to load the data source directly into the database. For example, use free-form data loading when the data is in the natural order used by Essbase, when the dimension, member, or alias names are required, and when the data is read according to the member names Essbase finds.
Free-Form mode	An option in the Hyperion Essbase Spreadsheet Add-in that allows you to type in report script commands in the spreadsheet to create reports.
free-form reporting	Creating reports by typing in members of dimensions or report script commands in the spreadsheet. Free-form reporting is available in both Advanced Interpretation mode and Free-form mode.
function	A predefined routine that returns a value or range of values, or controls formula execution. Essbase provides hundreds of functions within the following categories: mathematical, index, financial, macro, and Boolean.
generation	A description of a hierarchical tree structure that begins at the root, then the branches, and ends with the leaf nodes. The tree structure is a top-down description of an Essbase outline.
generation name	A unique name describing a generation.
Global Access layer	A layer in the Essbase security system used to define settings for an application or database.

global report commands	Global Report Writer commands are executed when they occur in the report script file, and stay in effect until the end of the report file or until another global command replaces them.
header record	One or more records at the top of a data source that describe the contents.
hierarchy	A set of multidimensional relationships often created in an outline in a tree formation. Examples of hierarchy are parent, child, and generation.
index	A method of retrieving Essbase data based on sparse dimensions. Also refers to the index file.
index cache	A buffer in memory that holds index pages.
index entry	A pointer to an intersection of sparse dimensions. Each index entry points to a block on disk, and locates a particular cell within the block by means of an offset.
index file	A file Essbase uses to store data retrieval information. It resides on disk and contains index pages.
index page	A subdivision of an index file containing entries that point to data blocks.
input block	A type of data block that has at least one loaded data value.
input data	Any data that is loaded from a data source and is not generated by calculating the database.
Intelligent Calculation	A calculation method that tracks which data blocks have been updated since the last calculation.
interactive mode	A method of using ESSCMD that lets you enter commands at the command line. Essbase prompts you for input if necessary.
interdimensional irrelevance	A member of a dimension that does not intersect with other dimensions.
Isolation Level	A Storage Manager setting that determines the lock and commit behavior of your database operations. Choices are *committed access* and *uncommitted access*.
jump point	The point at which you can drill across from a data value in one database to a corresponding location in another database.
Keep Only	A Hyperion Essbase Spreadsheet Add-in command that retains only the rows that are highlighted within a spreadsheet.
latest	A key word used within Hyperion Essbase Spreadsheet Add-in or Report Writer to extract data values based on the member defined as the latest period of time.

leaf member	A member that has no children; also referred to as detail member, level 0 member, and leaf node.
leaf node	*See* leaf member.
level	A branch within each dimension. The levels are numbered incrementally from the leaf member (level 0) towards the root.
level 0 block	A data block created for sparse member combinations, when all of the sparse members are level 0 members.
level 0 member	*See* leaf member.
level name	A unique name describing a level.
linked	A connection or jump point from one cell to a cell in another database, a cell note, or a separate file.
linked object	A term that encompasses linked partitions and linked reporting objects.
linked partition	A form of shared partition that provides the ability to link together two different databases with a data cell. When a user clicks on a linked cell in a spreadsheet, for example, Essbase drills across to a second database and a new sheet opens, displaying the dimensions in the second database. The user can then drill down into the available dimensions in the second database.
linked reporting object (LRO)	An external file linked to a data cell in an Essbase database. LROs can be cell notes or files containing text, audio, video, or pictures.
load data	*See* data load.
lock	A method to prevent two people from simultaneously altering the same set of data. The Essbase Storage Manager handles data block locking. When creating partitions, you can also place a lock on the partition definition file to prevent other users from editing its contents while you are working.
log file	An Essbase file that records actions and commands. An application log file records user actions performed on that application. A client log file records client messages, actions, and errors. The Agent log file records actions performed by the server Agent process. The Outline Change log file records changes you make to an Essbase database outline.
mathematical operators	*See* unary operators.
MDDB	A multidimensional database. *See also* multidimensional.
member	A discrete component within a dimension. For example, in the Sample Basic database, Year, Profit, and Diet are members. Members can also be dimensions; for example, Year is a dimension as well as a member.

member select	A feature within Hyperion Essbase Spreadsheet Add-in that allows you to specify a subset of members using selection criteria such as UDA, pattern match, or AND/OR.
member selection report command	A type of Report Writer command that selects ranges of members based on database outline relationships, such as sibling, generation, and level.
member-specific report command	A type of Report Writer formatting command that is executed as it is encountered in the report script. The command affects only the member to which it is associated, and executes the format command before it processes the member. *See also* formatting command.
metadata	The data that describes the data values within a database (for example, dimensions and member names).
Minimum Database Access	An option group that controls the default security to all of an application's databases, using access settings (such as Read or None) applied globally to the application. All users connecting to databases within the application will have at least the access level defined here; however, their own user privileges may be higher.
missing data	A marker in Essbase indicating that the data in this location does not exist, does not contain any meaningful value, or was never entered. Similar to a null value, but not the same as zero.
multidimensional	A method of referencing data through three or more dimensions. An individual record is the intersection of a point for a set of dimensions.
multithreading	A client/server process that enables multiple users to work on the same applications without interfering with each other.
Named Pipes	An API used by Microsoft LAN manager for special node-to-node applications and particularly for access to communications and database servers.
Navigate Without Data	A Hyperion Essbase Spreadsheet Add-in option that enables you to turn off data retrieval. This feature is most useful when your database has Dynamic Calc and Dynamic Calc And Store members.
nested column headings	A column heading containing more than one dimension in the report heading. For example, in the Sample Basic database, a column heading that contains both Year and Scenario members is a nested column.
object	A file related to an application or database. Objects can be outlines, rules files, calc scripts, report scripts, or data sources. They are stored within the application or database subdirectory on the server or client machine.

On-line Analytical Processing (OLAP)	A multidimensional, multi-user, client-server computing environment for users who need to analyze consolidated enterprise data in real-time. OLAP systems feature zooming, data pivoting, complex calculations, trend analyses, and modeling.
operator	*See* unary operator.
outline	The structure that defines all elements of a database within Essbase. It contains definitions of dimensions and members, dense or sparse dimension tags and attributes, calculations, shared members, and alterations to the basic roll-up structure of the database.
Outline Change Log	A record of changes made to the Essbase database outline.
page file	*See* data file.
page heading	A type of report heading that lists members represented on the current page of the report. All data values on the page have the members in the page heading as a common attribute.
paging	A storage scheme that makes use of spare disk space by increasing the available memory. Also referred to as swapping.
parent	A member that has a consolidation branch below it.
partition area	A subcube within a database. A partition is composed of one or more areas. These areas are composed of cells from a particular portion of the database. For replicated and transparent partitions, the number of cells within an area must be the same for the both the data source and the data target to ensure that the two partitions have the same shape. For example, if the data source area contains 18 cells, the data target area must also contain 18 cells to accommodate the number of values in the data source area.
partitioning	The process of defining areas of data that are shared or linked between data models. Partitioning can affect the performance and scalability of Essbase applications.
Partition Manager	An Essbase tool that enables you to easily create and maintain a replicated, linked, or transparent database partition. The Partition Manager includes the Partition Wizard containing a series of pages that step you through the partition creation process.
Password Management	A group of options in the server settings that lets you limit a user's allowed number of login attempts, number of days of inactivity, and number of days using the same password.
pattern matching	The ability to match values with substituted wildcard values such as question marks (?) and asterisks (*).
period	A specific key term used to define a Dynamic Time Series member that identifies a special period of time (for example, semi-annual).

pivot	The ability to alter the perspective of retrieved data. When Essbase first retrieves a dimension, it expands into rows. A user can pivot or rearrange the data to obtain a different viewpoint.
precalculation	The process of calculating the database prior to user retrieval.
preserve formulas	The process of keeping user-created formulas within a spreadsheet while retrieving new data.
record	A row of fields that is read as a unit from a data source.
redundant data	Duplicate data blocks that Essbase retains during transactions until Essbase commits the updated blocks.
Remove Only	A Hyperion Essbase Spreadsheet Add-in command that allows you to remove only the highlighted cells within a spreadsheet.
replicated partition	A portion of a database, defined through the Partition Manager, that lets you propagate updates to data mastered at one site to a copy of data stored at another site. Users are able to access the data as though it were part of the local database.
report	The formatted summary information returned from a database after a report script is run. One or more reports can be generated from a report script.
Report Editor	An ASCII text editor that you use to write report scripts. The Report Editor features a text editing window and customized menus. Saved report scripts have the file name extension `*.REP`.
Report Extractor	An Essbase component that retrieves report data from the Essbase database when you run a report script.
report script	An ASCII file that contains Report Writer commands that generate one or more production reports. Report scripts can be run in batch mode, using the ESSCMD command line interface, or through the Application Manager. The script is a text file that contains data retrieval, formatting, and output instructions.
Report Viewer	An Essbase component that displays the complete report after a report script is run. Saved reports have the file name extension `*.RPT`.
restore	An operation to reload data and/or structural information after a database has been damaged or destroyed. Typically performed after you shut down and restart the database.
restructure	An operation to regenerate or rebuild the database index and, in some cases, data files.
Retrieval Wizard	An Hyperion Essbase Spreadsheet Add-in command that allows you to easily design a report using a wizard tool.
roll-up	*See* consolidate.

root member	The highest member in a branch. Contrast with leaf member.
row heading	A report heading that lists members down a report page under their respective row names. Rows are indented below the dimension name.
script file	A text file (`*.SCR` extension) that contains ESSCMD commands, which Essbase executes in order to the end of the file. You can run a script file from the operating system command line or from within an operating system batch file.
server	A high-capacity, multi-user database server that locates and accesses data values based on their intersection of dimension members. It contains an agent and an application.
server application	*See* server.
server interruption	Any occurrence that stops the server, including a crash, a power outage, a user pressing the `Ctrl+C` keys, or certain errors.
shared member	A member that shares storage space with another member of the same name. This member has an attribute that designates it as shared. Shared members prevent Essbase from making extra calculations on a member that appears in more than one location in the outline.
sibling	A child member at the same branch level as another child member (for example, in the Sample Basic database, East and West are siblings).
sparse dimension	A dimension with a low percentage of available data positions filled. For example, a product not sold in all a company's available markets would be a good sparse candidate.
Spreadsheet Add-in	Essbase software that works with your spreadsheet. The Spreadsheet Add-in is an add-in module to your spreadsheet software.
Storage Manager	A layer of the Essbase server that handles mapping of Essbase data to physical disk storage. The Storage Manager controls lock, index, data, allocation, and transaction management functions.
Styles	*See* visual cues.
subset	A cross-section of data. Subsetting further defines members that meet specific conditions.
subsequent retrievals	Additional retrievals of the same member combinations after the first retrieval is complete.
substitution variable	A variable you can use to represent Essbase-specific values. You set the variable and a corresponding string value on the Essbase server. Substitution variables can then be applied throughout calculation scripts, report scripts, the Hyperion Essbase Spreadsheet Add-in, and the Essbase API.

supervisor	A defined type of user who has full access to all applications, databases, related files, and security mechanisms for a server.
suppress rows	Options that exclude rows containing missing or zero values and underscore characters from spreadsheet reports.
swapping	*See* paging.
Symmetric Multiprocessing (SMP)	A powerful computing process that enables multiple processors to function in parallel, made possible by the 32-bit, multi-threaded Essbase OLAP server application.
symmetric report	A symmetric report is characterized by repeating, identical groups of members.
system administrator	A person who maintains the hardware, software, disk space distribution, and configurations for running Essbase software.
template	A predefined screen designed to retrieve particular data on a regular basis and in a consistent format.
Time dimension	A dimension type that defines how often you collect and update data. You may only tag one dimension as Time, although you do not need to have a Time dimension.
time series reporting	A process of reporting data based on a date calendar (for example, year, quarter, month, or week).
toolbar	A bar of icons that represent Essbase commands. Icons are used as shortcuts to the Essbase menu.
Transmission Control Protocol/Internet Protocol (TCP-IP)	A standard set of communications protocols adapted by many companies and institutions around the world. Links computers with different operating systems and internal architectures. TCP/IP utilities let you exchange files, send mail, and store data to different computers connected to local and wide area networks.
transparent partition	A form of shared partition that provides the ability to transparently access and manipulate remote data as though it is part of the local database. The remote data is retrieved from the data source each time a user requests it. Any updates made to the data are written back to the data source and become immediately accessible to both local data target users and transparent data source users.
two-pass calculation	An Essbase attribute used for re-calculating an Accounts member, a Dynamic Calc member, or a Dynamic Calc And Store member, after a consolidation takes place.
unary operator	A group of mathematical indicators (+, -, *, /, %) that define how roll-ups take place on the database outline.

uncommitted access	A Storage Manager Isolation Level setting that affects how Essbase handles transactions. Under uncommitted access, concurrent transactions hold short-term Write locks and can yield unpredictable results.
upper-level block	A type of data block created for sparse member combinations, when at lease one of the sparse members is a parent level member.
user-defined attribute (UDA)	A method you use to return, within calc scripts and reports, lists of members with particular characteristics. You can report on subsets of data values predefined in the database outline.
validation	A process of checking a rules file against the outline to make sure the rules file is valid.
Version 2.x mode	*See* Free-Form mode.
visual cues	Formatted styles such as fonts and colors that highlight specific types of data values. Data values may be dimension members, parent, child, or shared members, Dynamic Calculations, members containing formulas, read only data cells, read/write data cells, or linked objects.
workbook	An entire spreadsheet file with many sheets.
zoom	The process of retrieving progressively detailed data relative to a selected dimension. Zooming in on a database dimension provides you with greater detail on that dimension, while zooming out moves your perspective to a higher consolidation level. Also referred to as drilling down.

Index

This index spans two volumes. Chapters 1–33 are in Volume I; Chapters 34–47 and an Appendix are in Volume II. Examples of entries: 33-10 means Chapter 33, p. 10; A-2 means Appendix A, p. 2.

Symbols

Numerics

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z