# Contents

**Documentation Accessibility** .......................................................... 11

**Chapter 1. Using Essbase Integration Server** ................................. 13

Integration Server ............................................................................ 13
Starting Integration Server ................................................................. 14

Running Integration Server as a Background Process on UNIX Systems .... 15
Integration Server Startup Switches ..................................................... 16

-? ........................................................................................... 17
-A .......................................................................................... 17
-B .......................................................................................... 17
-C .......................................................................................... 18
-D .......................................................................................... 19
-E .......................................................................................... 20
-F .......................................................................................... 20
-I .......................................................................................... 21
-L .......................................................................................... 22
-M .......................................................................................... 23
-N .......................................................................................... 23
-P .......................................................................................... 24
-Q .......................................................................................... 25
-R .......................................................................................... 25
-S .......................................................................................... 26
-T .......................................................................................... 26
-U .......................................................................................... 27

Storing Startup Information in the Configuration File ......................... 28
Storing Startup Information in the Startup File (Windows Only) ............ 29

Additional Configuration Parameters ................................................. 30
Windows Configuration Parameters .................................................... 30
Windows and UNIX Configuration Parameters ................................... 31
Reconnecting Integration Server and OLAP Metadata Catalog ............... 32
Shutting Down Integration Server ...................................................... 32
Integration Server Log File ................................................................. 32
Chapter 6. Working with Users, Locks, and Permissions ............................................. 115

Working with Users .................................................................................................... 115

Working with Locks .................................................................................................... 116

Using Standard Access Mode .................................................................................... 116

Using Exclusive Access Mode .................................................................................... 116

Correcting Problems with Locks ................................................................................ 117

Viewing Integration Services Users with Locks ......................................................... 117

Deleting Locks for Integration Services Users ........................................................... 118

Working with Permissions .......................................................................................... 119
### Viewing Permissions for OLAP Models

119

### Viewing Permissions for Metaoutlines

120

### Chapter 7. Troubleshooting ODBC and Connections

123

- Common Connection Problems
- Database-Specific Problems
- ODBC and Connection Problems
- Isolating Connection Problems
- Testing the Database Client Connection to the Data Source
- Testing the ODBC Connection to the Data Source
  - Testing ODBC on Windows Systems
  - Testing ODBC on UNIX Systems
- Testing the Integration Server Connection to the Data Source
- Testing the Integration Services Console Connection to the Data Source
- Testing the Integration Server Connection to Essbase Server
- Testing the Integration Services Console Connection to Essbase Server
- Correcting Connection Problems
  - User Name and Password Problems
  - Correcting Data Source Problems
  - Correcting Database Client Problems
  - Correcting ODBC Problems
  - Correcting Integration Server Problems
  - Correcting Integration Services Console Problems
- Using ODBC Tracing
  - Using ODBC Tracing on Windows Systems
  - Using ODBC Tracing on UNIX Systems

### Chapter 8. Using Integration Services Shell

147

- Starting Integration Services Shell
- Integration Services Shell
  - SET and LOAD Commands
  - Informational Commands
  - Command Syntax and Execution
  - Recording Member and Data Load Scripts
  - Running Integration Services Shell Script and Batch Files
  - Running OLAP Command Script Files
- Integration Services Shell Commands
  - LOGIN
  - STATUS
  - VERSION
Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.
This chapter describes starting and stopping Essbase Integration Server and how to view its log file.

**Integration Server**

Integration Server is multithreaded server software that is the bridge between the data source, OLAP Metadata Catalog, and Essbase Server. Oracle Essbase Integration Services performs several tasks:

- Retrieves OLAP model and metaoutline information from OLAP Metadata Catalog
- Generates SQL statements
- Retrieves data from external sources
- Loads members and data into the Oracle Essbase database

Integration Server must be running if you want to use Integration Services Console (the graphical user interface) or Integration Services Shell (the command line interface).
Starting Integration Server

Before starting Integration Server, verify all required components are running. These components need not be on your local computer. Also, verify that these components are running:

- The relational database management system (RDBMS) with the relational data source containing the OLAP Metadata Catalog where the metadata is stored
- One or more RDBMSs containing the relational data sources you want to use to create or modify an OLAP model or metaoutline

Essbase Server must be running if you create, change, or load data into an Essbase database or if you want to preview an Essbase database outline.

You can start Integration Server using several methods:

- From a DOS command prompt
- From a UNIX command prompt
- From the Windows Desktop Start menu
- As a Windows service

**Note:** To start Integration Server as a Windows service, you must have configured it as a Windows service during the configuration process. See the Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide.

To start Integration Server from DOS:

1. **At the command line, type the `startup.bat` command.**
   
   The `startup.bat` command ensures the environment is set correctly and that proper PATH and CLASSPATH settings are used when starting Integration Server.

2. **If you want to change a default startup switch, add the switch to the `ais.cfg` file or the `startup.bat` file.**

3. **If you want to change a default configuration parameter, add the parameter to the `ais.cfg` file.**

   See “Storing Startup Information in the Configuration File” on page 28 and “Storing Startup Information in the Startup File (Windows Only)” on page 29 for information on adding startup switches and configuration parameters to files.

To start Integration Server from UNIX:

1. **Type `is.sh`.**

2. **Type the executable name on the command line, for example, type `olapisvr`**

   **Note:** If Integration Server does not start when `olapisvr` is executed from the command line, the operating system path may not be updated correctly. See the Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide.
If you want to change a default startup switch, type the command syntax after the olapisvr command or add the switches to the ais.cfg file.

See “Storing Startup Information in the Configuration File” on page 28 and “Storing Startup Information in the Startup File (Windows Only)” on page 29 for information on adding startup switches to these files.

Values for startup switches entered at the command line override startup switch values stored in the ais.cfg file.

When entering parameters for startup switches at the command line, enclose path names that contain spaces in quotation marks (" "). The following example illustrates the syntax for passing the -E and -M options:

```
olapisvr -E"c:\my logs\eislog" -M"c:\ess\bin\essbase.mdb"
```

To start Integration Server from the Windows Desktop Start menu:

1. Click Start.
2. Select All Programs, then Oracle EPM System, then Essbase, then Integration Services, then Server.

If startup switch settings are stored in startup.bat or ais.cfg, the settings in startup.bat override the settings in ais.cfg when Integration Server is started from the Windows Desktop Start menu.

To start Integration Server as a Windows service:

1. Access the Services dialog box in Windows.
2. Double-click Essbase Integration Server to start the service.

   The Service Control message box is displayed. After a few seconds it is removed, and the Status field entry for Integration Server is changed to Started.
3. If you want to change any default startup switches or configuration parameters, add the switches to the ais.cfg file or the startup.bat file.

### Running Integration Server as a Background Process on UNIX Systems

To run Integration Server in the background, modify ais_start or type the following commands

```
.is.sh
nohup olapisvr &
```

**Note:** If you choose to run Integration Server by using the above commands, ensure you have configured your environment.

The nohup command keeps the Integration Server program running even if you log off. The ampersand (&) suffix makes the program a background process.
Caution! To allow the Integration Services Console client to create OLAP models and metaoutlines, you must run Integration Server and keep it running while any clients are connected to it.

When you start Integration Server, it begins writing to the log, `olapisvr.log`, in the `log` directory. To write to a log on another directory, specify the log name in `ais_start` or type the following command:

```
olapisvr -E mydir/mylog
```

**Note:** Do not type the `.log` file extension. The `.log` extension is automatically appended to the name that you type for the log.

### Integration Server Startup Switches

This topic lists the switches and configuration parameters used to change default settings when starting Integration Server. The switches can be used from the command line, added to the `startup.bat` file, or stored in the `ais.cfg` file.

- `-?`
- `-A`
- `-B`
- `-C`
- `-D`
- `-E`
- `-F`
- `-I`
- `[K]`
- `-L`
- `-M`
- `-N`
- `-P`
- `-R`
- `-S`
- `-T`
- `-U`
- `[V]`

For information on storing startup switches in `startup.bat` or `ais.cfg`, see “Storing Startup Information in the Configuration File” on page 28 and “Storing Startup Information in the Startup File (Windows Only)” on page 29.
The following topics describe the default switches.

-?  
To view a list of available switches for starting Integration Server.

-A  
To ignore the parent of a given member when updating OLAP intersections during a drill-through operation.

To ignore parents when updating OLAP intersections, do one of the following:

- In Windows, add -AY to the startup.bat file.
- In UNIX, type -AY when you start Integration Server.
- In Windows or UNIX, add [A]=Y to the ais.cfg file.

By default, Integration Server sets the value at N so that parents are not ignored when updating OLAP intersections. If a parent is ignored during an update of OLAP intersections, subsequent drill-through operations cannot be performed on any shared members related to that parent.

In Windows:
In the startup.bat file, to ignore the parent of a given member when updating OLAP intersections during a drill-through operation, add -AY after "C:\Oracle\Middleware\EPMSистем11R1\products\Essbase\eis\server\bin\olapisvr.exe". For example:
"C:\Oracle\Middleware\EPMSистем11R1\products\Essbase\eis\server\bin\olapisvr.exe" -AY

In UNIX:
To ignore the parent of a given member when updating OLAP intersections during a drill-through operation, type:
olapisvr -AY

In Windows or UNIX:
In the ais.cfg file, this parameter is specified in the following format:
[A]=Y

-B  
To set the fetch size that Integration Server uses when transferring rows from the data source to the Essbase database, type -BNumberOfRows when you start Integration Server. The default number is 100 rows; the range is from 0 to 32767 rows.

To maximize performance, estimate the size in bytes of the rows being loaded from the relational data source. Divide 32,000 by that estimated size to determine the optimal number of rows to fetch. The goal is to generate blocks of data that are as close to 32 KB as possible, without exceeding that amount. If the size of a block of rows exceeds the 32 KB threshold, the load will fail.
For example, to set the number of rows to 150, type

\texttt{olapisvr -B150}

In the \texttt{ais.cfg} file, this parameter is specified in the following format:

\texttt{[B]=-150}

\textbf{-C}

To set the number of records that Integration Server commits to Essbase:

- In Windows, add \texttt{-Cnumber_of_records} to the startup.bat file.
- In UNIX, type \texttt{-Cnumber of records} when you start Integration Server.
- In Windows or UNIX, add \texttt{[C]=number_of_records} to the \texttt{ais.cfg} file.

The default setting is all records. 

-C can be used with the -N switch.

\textbf{Note:} -C should not be used in operations involving aggregate storage.

-C is often used in testing environments. The recommended setting for testing purposes is 5,000 to 10,000 records. After you have completed testing, shut down Integration Server to revert to the default setting, or reset to the limit allowed by DATAERRORLIMIT.

When -C is used with the DATAERRORLIMIT setting in the \texttt{essbase.cfg} file, you can maximize the number of records written to \texttt{dataload.txt}, the data load error file. For example, if you set DATAERRORLIMIT to 65,000, the maximum number of error records allowed in the Essbase log, and then set -C to 65,000 or less, you will be able to view all error records allowed by Essbase.

\textbf{Note:} If you have included a setting for testing purposes for -C in \texttt{ais.cfg}, be sure to edit the file to clear the \texttt{ais.cfg} file so that it will change the setting back to “all records.”

Setting a smaller number of records during testing enables you to see data load errors incrementally, providing you an opportunity to correct problems before continuing. Resetting the number of records to the default of all records results in faster performance.

There is no limit to the number of records that Integration Server can transfer to Essbase during a data load. If, however, you experience memory problems during a data load, set -C to a smaller value.

\textbf{Note:} The setting of -C does not apply to member loads.

In Windows:
For example, in the startup.bat file, to set the number of records that Integration Server incrementally transfers to Essbase during a data load to 5,000, add -C5000 after "C:\Oracle\Middleware\EPMSYSTEM11R1\products\Essbase\eis\server\bin\olapisvr.exe". For example:
"C:\Oracle\Middleware\EPMSYSTEM11R1\products\Essbase\eis\server\bin\olapisvr.exe" -C5000

In UNIX:
For example, to set the number of records that Integration Server incrementally transfers to Essbase during a data load to 5,000, type:
olapisvr -C5000

In Windows or UNIX:
For example, to set the number of records that Integration Services incrementally transfers to Essbase during a data load to 5,000, in the ais.cfg file, type:
[C]=5000

-D
To specify whether or not to add the DISTINCT clause to the SELECT statement of drill-through SQL.
The default behavior adds the DISTINCT clause. To not add the DISTINCT clause:
● In Windows, add -DN to the startup.bat file.
● In UNIX, type -DN when you start Integration Server.
● In Windows or UNIX, add [D]=N to the ais.cfg file.

In Windows:
In the startup.bat file, to specify that the DISTINCT clause should not be added to drill-through queries, add -DN after "C:\Oracle\Middleware\EPMSYSTEM11R1\products\Essbase\eis\server\bin\olapisvr.exe". For example:
"C:\Oracle\Middleware\EPMSYSTEM11R1\products\Essbase\eis\server\bin\olapisvr.exe" -DN
Any value other than -DN or no value specifies that the DISTINCT clause should always be added to drill-through SQL.

In UNIX:
To specify that the DISTINCT clause should not be added to drill-through queries, type:
olapisvr -DN

In Windows or UNIX:
In the ais.cfg file, this parameter is specified in the following format:
[D]=N
-E

To give the log file a different name:

- In Windows, add `-E log_file_name` to the `startup.bat` file.
- In UNIX, type `-E log_file_name` when you start Integration Server.
- In Windows or UNIX, add `[E]=log_file_name` to the `ais.cfg` file.

The `.log` extension is added automatically. The default name and location is `products\logs\eis\olapisvr.log`.

**In Windows:**

In the `startup.bat` file, to name the log file `myserver.log` and put it in the temp directory on Windows, add `-Ec:\temp\myserver after "C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe". For example:

"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe -E"c:\temp\myserver"

Do not type the `.log` file extension. The `.log` extension is automatically appended to the name you type for the log.

Always enclose path names that contain spaces with quotation marks (" "); for example, type:

"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe" -I -E"c:\program files\eis\myserver"

**In UNIX:**

To name the log file `myserver.log` and put it in a temp directory on UNIX, type

`olapisvr -E/vol1/temp/eislog`

Enclose path names that contain spaces in quotation marks (" "); for example, type:

`olapisvr -E"/vol1/temp/eis log"

**In Windows or UNIX:**

In the `ais.cfg` file, this parameter is specified in the following format:

`[E]=c:\temp\myserver`

or

`[E]=c:\program files\eis\myserver`

**Note:** In the `ais.cfg` file, you do not need to type quotation marks around path names that contain spaces.

-F

To specify whether to add parentheses around user-defined drill-through filters. See Essbase Spreadsheet Add-in help.
The default is not to add parentheses around user-defined drill-through filters. To add parentheses:

- In Windows, add -FY to the startup.bat file.
- In UNIX, type -FY when you start Integration Server.
- In Windows or UNIX, add [F]=Y to the ais.cfg file.

In Windows:

In the startup.bat file, to specify that parentheses should be added around user-defined drill-through filters, add -FY after "C:\Oracle\Middleware\EPMS\products\Essbase\eis\server\bin\olapisvr.exe". For example:

"C:\Oracle\Middleware\EPMS\products\Essbase\eis\server\bin\olapisvr.exe" -FY

Any other value or no value specifies that parentheses should not be added around user-defined drill-through filters.

In UNIX:

To specify that parentheses should be added around user-defined drill-through filters, type:

olapisvr -FY

In Windows or UNIX:

In the ais.cfg file, this parameter is specified in the following format:

[F]=Y

-I

To prevent Integration Server from shutting down as a service on Windows after you log off, add -I to the startup.bat file.

When you run Integration Server as a service on Windows, Oracle recommends you use -I.

**Note:** This startup switch is applicable only in Windows environments.

**Note:** -I cannot be entered in the ais.cfg file. It can only be specified in the startup.bat file.

In Windows:

In the startup.bat file, to prevent Integration Server from shutting down as a service, add -I after "C:\Oracle\Middleware\EPMS\products\Essbase\eis\server\bin\olapisvr.exe". For example:

"C:\Oracle\Middleware\EPMS\products\Essbase\eis\server\bin\olapisvr.exe" -I
To set the level of detail of the messages that Integration Server logs:

- In Windows, add `-L level` to the `startup.bat` file.
- In UNIX, type `-L level` when you start Integration Server.
- In Windows or UNIX, add `[L]=I` to the `ais.cfg` file.

The default level is 2. You can specify the following levels:

- **0**
  
  To log all debug messages and the ODBC SQL generated by Integration Server; generate the `dataloaderrecord.txt` file, listing the first 1,000 records, which may include successfully loaded records and rejected records; and create the `hisdld.rul` file consisting of the names of the dimensions loaded followed by the leaf node number of each dimension or member.

- **2**
  
  To log all informational messages and some of the ODBC SQL generated by Integration Server.

- **3**
  
  To log all warnings.

- **4**
  
  To log all noncritical errors.

- **7**
  
  To log all critical errors where Integration Server terminates the command.

- **8**
  
  To log all critical server errors.

The recommended level setting is 2 or 3.

**In Windows:**

In the `startup.bat` file, to use the highest level of logging, add `-L0` after "C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe". For example:

"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe" -L0

**In UNIX:**

To use the highest level of logging, type:

`olapisvr -L0`

**In Windows or UNIX:**

In the `ais.cfg` file, this switch is specified in the following format:

`[L]=0`
-M
To specify the location of the Essbase message database file if the file is moved to a new location:

- In Windows, add `-M path\essbase.mdb` to the `startup.bat` file.
- In UNIX, type `-M path/essbase.mdb` when you start Integration Server.
- In Windows, add `[M]=path\essbase.mdb` to the `ais.cfg` file.

The default location is `products\products\Essbase\eis\server\bin\essbase.mdb`.

In Windows:
In the `startup.bat` file, to specify the new location of the Essbase message database file as `essbasemsg\bin`, add `-Mc:\essbasemsg\bin\essbase.mdb` after "C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe". For example:
"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe" -Mc:\essbasemsg\bin\essbase.mdb

Always enclose path names that contain spaces with quotation marks (" "); for example, type:
"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe" -M"c:\eissvr\bin\essbase.mdb"

In UNIX:
To specify the new location of `essbase.mdb` as `essbasemsg\bin`, type:
`olapisvr -M/vol1/essbasemsg/bin/essbase.mdb`

Enclose path names that contain spaces with quotation marks (" "); for example:
`olapisvr -M"/vol1/essbase msg/bin/essbase.mdb"

In Windows or UNIX:
In the `ais.cfg` file, this parameter is specified in the following format:

`[M]=c:\eissvr\bin\essbase.mdb`

or

`[M]=c:\ess svr\bin\essbase.mdb`

Note: In the `ais.cfg` file, you do not need to add quotation marks around path names that contain spaces.

-N
To specify the number of threads Integration Server uses when sending data to Essbase during a data load, do one of the following:

- In Windows, add `-N` to the `startup.bat` file.
- In UNIX, type `-N number_of_threads` when you start Integration Server.
In Windows or UNIX, add \[N]=\text{number\_of\_threads}\) to the ais.cfg file.

The default setting is 1 thread.

\(-N\) may be used with \(-C\).

\(-N\) controls the number of threads allocated to data load optimization. Usually, users start with 2 threads and adjust according to the environment.

**In Windows:**

In the startup.bat file, to set the number of threads to 2, add \(-N2\) after "\(C:\)\Oracle\Middleware\EPMS\Server\bin\olapisvr.exe". For example:

"\(C:\)\Oracle\Middleware\EPMS\Server\bin\olapisvr.exe" -N2

**In UNIX:**

To set the number of threads to 2, type:

`olapisvr -N2`

**In Windows or UNIX:**

In the ais.cfg file, this parameter is specified in the following format:

\([N]=2\)

**-P**

To set the TCP port number with which Integration Server communicates with its clients to a port different from the default 3388:

- In Windows, add \(-P\text{portnumber}\) to the startup.bat file
- In UNIX, type \(-P\text{portnumber}\) when you start Integration Server
- In Windows or UNIX, add \([P]=\text{portnumber}\) to the ais.cfg file

**In Windows:**

In the startup.bat file, add \(-P\text{portnumber}\) after "\(C:\)\Oracle\Middleware\EPMS\Server\bin\olapisvr.exe". For example:

"\(C:\)\Oracle\Middleware\EPMS\Server\bin\olapisvr.exe" -P8850

If you change the default port number, in Integration Services Console you must log in to the Integration Server by typing the server name or IP address and the non-standard port number, separated by a colon; for example:

`aspen:8850`

When starting Integration Services Shell, to change the TCP port number, at the command line, type \(-P\text{portnumber}\). For example:

`olapicmd -P8850`

**In UNIX:**
When starting Integration Server or Integration Services Shell, to change the TCP port number, type:

```
olapisvr -P8850
```

or

```
olapicmd -P8850
```

**In Windows or UNIX:**

In the `ais.cfg` file, this parameter is specified in the following format:

```
[P]=8850
```

**-Q**

*Note:* -Q is no longer available.

**-R**

To specify that users are restricted from browsing table records when customizing a drill-through report.

- When -R is not set, users have unrestricted access to all table records.
- When -R is set to Y, users have no access to any table records.
- When -R is set to F, users have no access to fact table records but have unrestricted access to all other table records.

To restrict users from browsing fact table records:

- In Windows, add `-Raccess_parameter` to the `startup.bat` file.
- In UNIX, type `-Raccess_parameter` when you start Integration Server.
- In Windows or UNIX, add `[R]=access_parameter` to the `ais.cfg` file.

By default, -R is not set, giving users unrestricted access to all table records.

**In Windows:**

In the `startup.bat` file, to prevent users from accessing any table records, add -RY after "C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe". For example:

```
"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe" -RY
```

In the `startup.bat` file, to prevent users from accessing fact table records, add -RF after "C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe". For example:

```
"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe" -RF
```

**In UNIX:**
To prevent users from having any access to any table records, type:
\texttt{olapisvr -RY}

To prevent users from having any access to fact table records, type:
\texttt{olapisvr -RF}

\textbf{In Windows or UNIX:}

In the \texttt{ais.cfg} file, this parameter is specified in the following format:

\begin{verbatim}
[R]=Y
\end{verbatim}

or

\begin{verbatim}
[R]=F
\end{verbatim}

\textbf{-S}

To specify the stack size in megabytes of each thread that Integration Server creates:

\begin{itemize}
  \item In Windows, add \texttt{-Stacksize\_in\_megabytes} to the \texttt{startup.bat} file.
  \item In UNIX, type \texttt{-Stacksize\_in\_megabytes} when you start Integration Server.
  \item In Windows or UNIX, add \texttt{[S]=stacksize\_in\_megabytes} to the \texttt{ais.cfg} file.
\end{itemize}

For Windows and Solaris, no stack size setting is required. For AIX, the default stack size setting is 3 MB; for HP-UX, the default stack size setting is 8 MB.

\textbf{In Windows:}

In the \texttt{startup.bat} file, to set the stack size that Integration Server creates to 1 MB, add \texttt{-S1} after \texttt{"C:\Oracle\Middleware\EPMS\System11R1\products\Essbase\eis\server\bin\olapisvr.exe"}. For example:

\begin{verbatim}
"C:\Oracle\Middleware\EPMS\System11R1\products\Essbase\eis\server\bin\olapisvr.exe" -S1
\end{verbatim}

\textbf{In UNIX:}

To set the stack size that Integration Server creates to 2 MB, type:

\texttt{olapisvr -S2}

\textbf{In Windows or UNIX:}

In the \texttt{ais.cfg} file, this parameter is specified in the following format:

\begin{verbatim}
[S]=2
\end{verbatim}

\textbf{-T}

To set the number of network listeners that Integration Server starts:

\begin{itemize}
  \item In Windows, add \texttt{-T\_number} to the \texttt{startup.bat} file
  \item In UNIX, type \texttt{-T\_number} when you start Integration Server
  \item In Windows or UNIX, add \texttt{[T]=\_number} to the \texttt{ais.cfg} file
\end{itemize}
The default (and recommended) number is 10.

Network listeners receive requests from Integration Services Console. Integration Server automatically adds and subtracts listeners as needed, so the number of listeners set at startup does not limit the number of users that can connect to an Integration Server.

**In Windows:**

In the `startup.bat` file, to set the number of network listeners that Integration Server starts to 17, add `-T17` after `"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe"`. For example:

```
"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe" -T17
```

**In UNIX:**

To set the number of network listeners Integration Server starts to 17, type:

```
olapisvr -T17
```

**In Windows or UNIX:**

In the `ais.cfg` file, `-T` is specified in the following format:

```
[T]=17
```

**-U**

To specify whether the Essbase database and application should be unloaded from memory after a load is completed.

Integration Server can successfully unload the Essbase database from memory only if there are no users accessing it.

To specify whether the Essbase database and application should be unloaded from memory after a load is completed, do one of the following:

- In Windows, add `-UY` to the `startup.bat` file.
- In UNIX, type `-UY` when you start Integration Server.
- In Windows or UNIX, add `[U]=Y` to the `ais.cfg` file.

By default, Integration Server does not unload the Essbase database from memory after a load is completed.

**In Windows:**

In the `startup.bat` file, to specify whether the Essbase database and application should be unloaded from memory after a load is completed, add `-UY` after `"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe"`. For example:

```
"C:\Oracle\Middleware\EPMSystem11R1\products\Essbase\eis\server\bin\olapisvr.exe" -UY
```

**In UNIX:**

To specify whether the Essbase database and application should be unloaded from memory after a load is completed, type
In Windows or UNIX:
In the ais.cfg file, -U is specified in the following format:
[U]=Y

**Storing Startup Information in the Configuration File**

The ais.cfg file is used for two purposes:

- To pass startup switches and configuration parameters, other than the defaults, when launching Integration Server
- As an environment for Teradata and Oracle users to configure the required JDBC drivers, as discussed in “Mapping JDBC Data Sources for Windows (Teradata Only)” on page 69 and “Mapping JDBC Data Sources for UNIX (Teradata Users Only)” on page 70.

The ais.cfg file is a text file residing in the \bin directory on Windows, UNIX, and Linux platforms. When you start Integration Server from the command line using the startup.bat command (in DOS or UNIX), the startup routine checks the contents of ais.cfg for any startup switch or parameter information. Then one of the following applies:

- Startup switch or configuration parameter syntax added to the ais.cfg file override default settings (see “Integration Server Startup Switches” on page 16 and “Storing Startup Information in the Startup File (Windows Only)” on page 29).

- If you did not add startup switch or parameter information to ais.cfg, the default settings are used to start Integration Server.

- If ais.cfg contains startup switch syntax and you enter startup switch overrides in the startup.bat file (Windows only) any startup switches in startup.bat override both the startup switch settings contained in the ais.cfg file and the default settings.

To store startup switch and parameter settings in the ais.cfg file:

1. **Open the ais.cfg file in any text editor.**
2. **Add the startup switch you want to store by typing the startup switch letter in brackets; for example:**
   
   [B]

3. **Add the new information for startup switches in the following format:**

   
   [L]=0
   [E]=c:\temp\myserver
   [N]=40
   [M]=c:\ess\bin\essbase.mdb

   To enter the above overrides in the startup.bat file, type:

   "C:\Hyperion\products\Essbase\eis\server\bin\olapisvr.exe" -I -L0 -Ec:\temp\myserver
   -N40 -M"c:\ess\bin\essbase.mdb"

4. **Save and close ais.cfg.**
Storing Startup Information in the Startup File (Windows Only)

The `startup.bat` file has two purposes:

- As an environment to configure the common Oracle Enterprise Performance Management System components, such as Java, ODBC, and JDBC drivers, for Teradata and Oracle users. See the Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide.
- As a means to pass startup switches and configuration parameters, other than the defaults, when launching Integration Server

The `startup.bat` file is a text file residing in the `eis\server\bin` directory. When you start Integration Server as a Windows service, the startup routine checks the contents of `startup.bat` for any startup switch information. Then the startup routine checks `ais.cfg` for any startup switch and configuration parameter information. Then one of the following applies:

- Startup switch syntax added to the `startup.bat` file (Windows only), override default settings (see “Integration Server Startup Switches” on page 16).
- If you added startup switch or configuration parameter syntax only to the `ais.cfg` file, those settings that you added override the default settings for those switches and parameters (see “Integration Server Startup Switches” on page 16 and “Storing Startup Information in the Configuration File” on page 28).
- If you did not add startup switch or parameter information to `startup.bat` or `ais.cfg`, the default settings (see "Integration Server Startup Switches" on page 16 and “Storing Startup Information in the Configuration File” on page 28) are used to start Integration Server.

To store startup switch settings in the `startup.bat` file:

1. Open the `startup.bat` file in any text editor.
2. If you want to change a default startup switch, type the command syntax in the line that calls the Integration Server.

   The values for startup switches that you enter into `startup.bat` override any startup switch values you have stored in the `ais.cfg` file.

   When entering parameters for startup switches in `startup.bat`, enclose path names that contain spaces in quotation marks (" "). The following example illustrates the syntax for passing the -L, -E, -M, -N options:

   "C:\Hyperion\eis serv\bin\olapisvr.exe" -I -L0 -Ec:\temp\myserver -N40 -M"c:\ess svr\bin\essbase.mdb"

   **Note:** In the `startup.bat` file, the above syntax should be entered all on one line.
You can only enter startup switch information into `startup.bat`. Configuration parameters must be entered in `ais.cfg`.

3 **Save and close** `startup.bat`.

### Additional Configuration Parameters

**Subtopics**

- Windows Configuration Parameters
- Windows and UNIX Configuration Parameters

Add the configuration parameters specified in this topic to the `ais.cfg` file to change the default configuration settings described in “Storing Startup Information in the Configuration File” on page 28.

### Windows Configuration Parameters

The configuration parameters described in this section can only be specified in the `ais.cfg` file that is installed on Windows. These parameters cannot be specified in the `startup.bat` file, at the command line, or in the `ais.cfg` file that is installed with the UNIX version of Integration Server.

**[K]**
To specify whether to display the primary keys in an OLAP model. The default behavior is not to display the primary keys in an OLAP model.
If you want to display the primary keys in an OLAP model, add `[K]=Y` to the `ais.cfg` file.

**In Windows:**
In the `ais.cfg` file, this parameter is specified in the following format:

```
[K]=Y
```

**[V]**
To specify whether Integration Server automatically validates an OLAP model or metaoutline when a Save or Save As operation is performed. The default behavior is for Integration Server to validate an OLAP model or metaoutline before saving it.
If you want to disable automatic OLAP model and metaoutline validation, add `[V]=0` to the `ais.cfg` file.

**In Windows:**
In the `ais.cfg` file, this parameter is specified in the following format:

```
[V]=0
```
Windows and UNIX Configuration Parameters

The configuration parameter described in this section can be specified in the ais.cfg file that is installed on Windows and UNIX. This parameter cannot be specified in the startup.bat file or at the command line.

[ADDDUPLICATESFORASO]

To allow addition of duplicate data records when using an aggregate storage database. The default behavior is to set the data load option in aggregate storage to not allow duplicate records.

To allow duplicate data records when using an aggregate storage database, add [ADDDUPLICATESFORASO]=Y to the ais.cfg file.

In Windows and UNIX:

In the ais.cfg file, this parameter is specified in the following format:

[ADDDUPLICATESFORASO]=Y

[UNIFORMVALUEFORMAT]

To obtain correct results when an aggregate storage database contains duplicate records with different values. In the following example, duplicate records at the intersection of SKU, MONTH, and STATE show different results for SALES:

<table>
<thead>
<tr>
<th>SKU</th>
<th>MONTH</th>
<th>STATE</th>
<th>SALES</th>
</tr>
</thead>
<tbody>
<tr>
<td>100-10</td>
<td>JAN</td>
<td>CA</td>
<td>389.99999999999994</td>
</tr>
<tr>
<td>100-10</td>
<td>JAN</td>
<td>CA</td>
<td>390</td>
</tr>
</tbody>
</table>

When a database contains duplicate records with different values, data loads are terminated with an error; for example:

IS Error(EssLoadBufferTerm): Data load failed: input contains different values for the same cell [(Jan, Sales, 100-20, Florida): 389.99999999999994 / 390] 1270089

When you add the [UNIFORMVALUEFORMAT] parameter to the ais.cfg file, data load results for measures are rounded up or down as appropriate to six decimal places (0.000000), ensuring that the duplicate records will have the same value.

Using the same data from the example above, the results would be:

<table>
<thead>
<tr>
<th>SKU</th>
<th>MONTH</th>
<th>STATE</th>
<th>SALES</th>
</tr>
</thead>
<tbody>
<tr>
<td>100-10</td>
<td>JAN</td>
<td>CA</td>
<td>390.000000</td>
</tr>
<tr>
<td>100-10</td>
<td>JAN</td>
<td>CA</td>
<td>390.000000</td>
</tr>
</tbody>
</table>

To enable uniform value formatting, add [UNIFORMVALUEFORMAT]=Y to the ais.cfg file

In Windows and UNIX:

In the ais.cfg file, this parameter is specified in the following format:

[UNIFORMVALUEFORMAT]=Y
Reconnecting Integration Server and OLAP Metadata Catalog

If you lose the connection to Integration Server and OLAP Metadata Catalog, you can reconnect directly from Integration Services Console without losing unsaved work.

➢ To reconnect to Integration Server:

2. In the Integration Services Console, select Connections, then OLAP Metadata Catalog, then Reconnect.

   **Note:** The connection is made implicitly; no messages are displayed.

Shutting Down Integration Server

After using Integration Server, shut it down using any of several methods.

**Note:** Always close Integration Services Console before shutting down Integration Server.

➢ To shut down Integration Server from the command line, either close the Integration Server window, or use Integration Services Shell:

- On Windows platforms, close the Integration Server window.
- Issue the SHUTDOWN command in Integration Services Shell.

See “Integration Services Shell Commands” on page 152.

➢ To shut down Integration Server from the Windows desktop:

1. Right-click the button representing Integration Server in the Windows task bar.
2. Select Close.

➢ To shut down Integration Server as a Windows service:

1. Access the Services dialog box in Windows.
2. Locate Essbase Integration Server in the list of services.
3. Right-click Essbase Integration Server and select Stop.

Integration Server Log File

By default, the Integration Server log file is named olapisvr.log and is located in the products\logs\eis directory. View it from Integration Services Console by selecting the View
Log File option from the Tools menu to launch the Server Log file dialog box, or open olapisvr.log in any text editor. Use the -E startup switch to create the log file with a different location or name. See “-E” on page 20.

**Items Logged in Integration Server File**

Items are logged in the Integration Server log file, depending on the logging level you set when you start Integration Server:

- Processing messages
- Member load error messages
- Settings for switches used to start Integration Server from the command line or as a Windows service
- Two copies of each SQL statement generated to access the data source as follows:
  - The first statement is in the SQL dialect understood by the ODBC driver and is generated by Integration Server.
  - The second statement is generated by the ODBC driver and is translated by the driver into the dialect of SQL understood by the data source being accessed.
- Integration Services Shell commands used to perform an action
- Connection information
- Load status
- Names of any metaoutline running
- Other informational messages

When a user accesses a drill-through report using Essbase Spreadsheet Add-in, items related to the drill-through report are also logged:

- Drill-through report name
- Metaoutline with which this drill-through report is associated
- Essbase Server computer being accessed
- Essbase application and database
- User accessing drill-through report

See “Integration Server Startup Switches” on page 16.

Integration Server error message text is located in `products\Essbase\eis\server\bin\error.txt` on Windows systems; `products/Essbase/eis/server/bin/error.txt` on UNIX.

Essbase error message text is located in `products\Essbase\eis\server\bin\message.txt` on Windows systems; `products/Essbase/eis/server/bin/message.txt` on UNIX.
Integration Server Log File Size

Your server log file will grow to a maximum of 50 megabytes in size. Then it will be truncated to 10 megabytes with the most recent 10 megabytes of entries being retained. The log file will again grow to 50 megabytes before it is truncated.

Change the default settings by specifying LOGFILEMIN and LOGFILEMAX in the ais.cfg file, for example:

\[
\begin{align*}
[\text{LOGFILEMIN}] &= 20 \\
[\text{LOGFILEMAX}] &= 60
\end{align*}
\]

Data Load Error File

If data loads generate errors, the file dataload.txt is placed in a folder Integration Server creates for the load under the products/Essbase/eis/server/loadinfo directory. The dataload.txt file lists rejected dimensions and members and error message codes to identify data load problems.

The folder Integration Server creates in the loadinfo directory is in the following format:

<application_database_timestamp_sessionnumber>

If you performed a data load for the MyTBC application and MyTBC_DB database at 10 P.M. on May 1, 2003, the folder is named:

MyTBC_MyTBC_DB_2003_May_1_10_00_pm_<sessionnumber>

In the case of data load failure, open the dataload.txt file located in the folder described previously and review the error codes.

These are the most common error codes:

<table>
<thead>
<tr>
<th>Message Number</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>3303</td>
<td>Member not found in database.</td>
</tr>
<tr>
<td>3304</td>
<td>Insufficient access to store data.</td>
</tr>
<tr>
<td>3333</td>
<td>Bad data value supplied.</td>
</tr>
<tr>
<td>3335</td>
<td>Record rejected because of duplicate member names.</td>
</tr>
<tr>
<td>3336</td>
<td>Member/Data Unknown.</td>
</tr>
<tr>
<td>3337</td>
<td>Record rejected because of dimension conflicts with Header Name.</td>
</tr>
</tbody>
</table>

Figure 1 shows three records from the dataload.txt file with the error code 3303.
Sharing data across national and language boundaries is a challenge for multi-national businesses. Traditionally, each computer stores and renders text based on its locale specification. A locale identifies the local language and cultural conventions such as the formatting of currency and dates, sort order of the data, and the character set encoding to be used on the computer. The encoding of a character set refers to the specific set of bit combinations used to store the character text as data, as defined by a code page or an encoding format. In Essbase, code pages map characters to bit combinations for non-Unicode encodings.

Because different encodings can map the same bit combination to different characters, a file created on one computer can be misinterpreted by another computer that has a different locale.

The Unicode Standard was developed to enable computers with different locales to share character data. Unicode provides encoding forms with thousands of bit combinations, enough to support the character sets of multiple languages simultaneously. By combining all character mappings into a single encoding form, Unicode enables users to correctly view character data created on computers with different locale settings.

Users whose computers are set up in different languages can work with the same database. For example, using alias tables in their respective languages, users in Taiwan can view database reports displaying Chinese characters while users in France can view the same reports in French characters.

User-defined character sets (UDC) are not supported and the Chinese National Standard GB 18030-2000 is not supported.

**Note:** For information on using Unicode in Essbase, see the Oracle Essbase Database Administrator’s Guide.

### Unicode and Non-Unicode Application Modes

Applications are designated as Unicode-mode applications or non-Unicode-mode applications. **Unicode-mode applications** support multiple character sets. Integration Services uses the UTF-8 encoding form to interpret and store character text. Character-based artifacts in Unicode-mode applications, such as member and alias names, can include characters from different languages.

Clients working with Unicode-mode applications can have different locales than Essbase Server. For example, client computers with Japanese locales and client computers with German locales can work with the same Unicode-mode application on an Essbase Server that has a Spanish locale.
Non-Unicode-mode applications support one character set that is defined by a locale value, which must be the same for Essbase Server and all non-Unicode clients that work with the non-Unicode-mode applications. Both Unicode-mode and non-Unicode-mode applications can reside on the same Essbase Server.

**Note:** You cannot convert a Unicode-mode application to non-Unicode mode.

### When to Use Unicode-Mode Applications

Consider working with Unicode-mode applications only if you have any of the following situations:

- You need to enable users with different languages to view, in their own languages and character sets, information from a common database. For example, using alias tables in Japanese and German, users in Japan and Germany could view, in their own languages, information about a common product set.

- You need to handle artifact names longer than non-Unicode-mode applications support. For example, application and database names need to be larger than eight characters or, if you are working with a multi-byte character set, you need to handle more characters in artifact names.

- You have experienced what is called the “round-trip” problem. The round-trip problem can occur in communications between multi-byte operating systems and application programs where two different bit values can map to the same character. As Java applications, Oracle Essbase Administration Services and Oracle Hyperion Provider Services always work in Unicode. No encoding conversions occur when these clients work with Unicode-mode applications and UTF-8-encoded text files; hence no round-trip conversion errors.

When deciding on using Unicode-mode applications, you should also consider the following points:

- Using non-Unicode text files with Unicode-mode applications requires an understanding of locales and care in managing to them. To prevent errors that could cause database corruption, using UTF-8-encoded files is recommended. For details, see the *Oracle Essbase Database Administrator’s Guide*.

- To work with Unicode-mode applications, custom client applications that were written to support non-Unicode-mode applications must be built to use the longer string lengths used by Unicode-mode applications. This may be a simple re-build or may involve re-programming, depending on the design of the applications. Also, depending on how they are coded, the new client applications may require more memory.

### Unicode-Enabled Administration Tools

Oracle provides Administration Services and MaxL to administer Unicode-mode applications. The main administration activities include, in addition to the normal Essbase administration activities, changing the Unicode-related mode of the Essbase Server to enable or disable creation of Unicode-mode applications, creation of Unicode-mode applications, migration of non-
Unicode-mode applications to Unicode mode, and viewing the Unicode-related status of servers and applications.

### Enabling Regional Options for Multiple Languages

#### Subtopics
- Enabling Regional Options in Windows XP
- Enabling Regional Options in Windows 2003

When you enable the appropriate languages in Windows, Integration Services Console can display OLAP model and metasubtree elements in the languages of the source database. Integration Services Console uses the settings you configure in Windows to display languages.

### Enabling Regional Options in Windows XP

To enable the Regional and Language Options in Windows XP:

1. In the Windows taskbar, select **Start**, then **Settings**, then **Control Panel** and double-click **Regional and Language Options**.
   
   The Regional Options tab of the Regional and Language Options dialog box is displayed.

2. In the **Standards and formats** frame, from the drop-down list, select the appropriate language to use for formatting items such as numbers, currencies, time, and dates.
   
   In most cases, you set this language to match your location.

3. Click the **Languages** tab and click the **Details** button.
   
   The Text Services and Input Languages dialog box is displayed.

4. In the **Installed services** frame, click **Add**.

5. In the **Input Languages** list, select the language that you want to add and select the check box for the type of text service you want to install.
   
   For example, you may select Cyrillic to display Russian characters or Simplified Chinese and Traditional Chinese to display Chinese characters.

6. If the languages that you need do not appear in the **Input Languages** frame, or if you want to add a specific code page to a language, click the **Advanced** tab and perform the following tasks:
   
   a. In the **Code page conversion tables** frame, select the appropriate code page conversion tables.
      
      Select the code page tables that correspond to the languages that you want to display on the Integration Services Console computer.

   b. Click **Apply**.

7. Click **OK** to close the Regional Language and Options dialog box.
Enabling Regional Options in Windows 2003

To enable the Regional and Language Options in Windows 2003:

1. In the Windows taskbar, select Start, then Control Panel and double-click Regional and Language Options.

   The Regional Options tab of the Regional and Language Options dialog box is displayed.

2. In the Standards and formats frame, from the drop-down list, select the appropriate language to use for formatting items such as numbers, currencies, time, and dates.

   In most cases, you set this language to match your location.

3. Click the Languages tab and click the Details button.

   The Text Services and Input Languages dialog box is displayed.

4. In the Default input language drop-down list, select the installed input language to use.

5. If the language you want does not appear in the Default input language drop-down list, click Add in the Installed services frame.

   The Add Input Language dialog box is displayed.

6. In the Input languages list, select the language that you want to add.

7. In the Keyboard layout/IME list, select the check box for the type of text service you want to install and click OK.

   For example, you may select Cyrillic to display Russian characters or Simplified Chinese and Traditional Chinese to display Chinese characters.

8. Optional: If the languages that you need do not appear in the Add input languages dialog box, or if you want to add a specific code page to a language, perform the following steps:

   a. Click Cancel to close both the Add input languages dialog box.

   b. Click Cancel to close the Text Services and Languages dialog box.

   c. In the Regional and Language Options dialog box, click the Advanced tab.

   d. In the Code page conversion tables frame, select the appropriate code page conversion tables.

      Select the code page tables that correspond to the languages that you want to display on the Integration Services Console computer.

   e. Click Apply.

9. Click OK to close the Regional Language and Options dialog box.

   For more information on Windows Regional Language and Options, refer to the Windows 2003 documentation.
Creating Database User Aliases and Synonyms

When using the sample application, you must connect to TBC_MD as user TBC unless you set up your user name as an alias or synonym for TBC.

- On Microsoft SQL Server, make your login user name an alias of TBC.
- On IBM DB2 and Oracle, log in with your user name and password, and create synonyms that map to the tables in TBC_MD.

This user alias or table synonym enables you to access a table that is qualified by TBC because it was created by the TBC user; for example, TBC.MO_INFO in the TBC_MD OLAP Metadata Catalog.

For more information, see the documentation for the RDBMS you are using.

Scheduling Jobs in UNIX

➢ To schedule jobs, perform the following steps:

1. Set up the $ENV variable to a file that initializes the environment for Integration Services, including running is.sh.

2. Give authorization to the Integration Services user to schedule jobs using the cron scheduling daemon.

After the configurations are complete, you can access the relational data source to create an OLAP Metadata Catalog and—if you choose—to set up the sample application. For more information about creating the OLAP Metadata Catalog, see Chapter 3, “Creating, Upgrading, and Deleting OLAP Metadata Catalogs” For more information about setting up the sample application, see Chapter 5, “Setting Up the Sample Applications.”

Starting the Windows Task Scheduler Service

Integration Services uses the Task Scheduler service for the Windows platforms to perform scheduled member and data loads. To allow loads to be scheduled, you must ensure that both of these conditions are met:

- The Task Scheduler service must be started on the computer that runs Essbase Integration Server.
- The Task Scheduler service is enabled to start automatically each time the computer that runs Essbase Integration Server is started.

➢ To start the Windows Task Scheduler service:

1. On the Windows desktop, click Start.

2. Select Control Panel.

3. In the Control Panel window, click Administrative Tools.
In the Administrative Tools window, double-click the Services icon.

In the list of services, locate Task Scheduler and verify that the entry in the Status column is Started and the entry in the Startup type column is Automatic.

If you must change one or both entries, this is the process:

a. Double-click Task Scheduler.

b. In the Task Scheduler Properties dialog box, select the General tab.

c. Under Startup type, select Automatic.

d. To start the service, click Start in the Task Scheduler Properties dialog box.
To create OLAP models or build metaoutlines, you must connect Integration Services to a database that contains metadata (the OLAP Metadata Catalog database) and to a database in which needed users' data is stored (the Data Source database). To make these connections, you must configure a data source name for an OLAP Metadata Catalog database and a Data Source database. You do this by mapping a supported Open Database Connectivity (ODBC) driver to each database.

Note: You are not required to use two different databases for your Data Source database and OLAP Metadata Catalog database. Additionally, you are not required to use two different data source names to connect to the Data Source database and OLAP Metadata Catalog database if the two databases reside in the same database.

About Data Source Configuration

In Integration Services, a data source name must be configured for the source database or flat files that contains users' data (the Data Source database). A data source must also be configured for the relational database that contains the OLAP Metadata Catalog. Databases, including an OLAP Metadata Catalog, can run on any supported platform, provided that you have the ODBC driver and, in some cases, the database client software needed to access them.

The Integration Services installation program installs DataDirect ODBC driver files that you use to configure data source names for both the Data Source database and OLAP Metadata Catalog database.
If database client software is required, the relational database management system (RDBMS) should include one of the database clients listed in the Oracle Hyperion Enterprise Performance Management System Certification Matrix (http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html). If your configuration requires database client software, ensure that the versions of the database client and the RDBMS are compatible.

This chapter provides information on how to configure ODBC drivers. For information on how to install and configure database client software, see the RDBMS documentation.

**Note:** Integration Services does not support data source table names and column names that contain spaces or special characters, such as a period (.). See Appendix B, “Integration Services Limits and Guidelines” for a complete listing of the unsupported characters.

On Windows, use the ODBC Administrator to configure ODBC data sources. All ODBC data source names are configured only on the computer that runs Essbase Integration Server.

**Note:** If a supported ODBC driver is already mapped to a Data Source database, you do not need to map it again. In this case, you need to map a supported ODBC driver only to the OLAP Metadata Catalog database.

### Server and Client Software and ODBC Connections

Integration Server runs on Windows, AIX, HP-UX, Solaris, or Red Hat Enterprise Linux. You must install Integration Server software, including ODBC drivers, and if required, database client software on the Integration Server computer.

Integration Services client software runs on Windows. You must install the Integration Services Console software on Integration Services client computers.

Configure the ODBC connections to the Data Source database and the OLAP Metadata Catalog database only on the computer running Integration Server. You do not need to configure ODBC connections on Integration Services client computers.

### Supported ODBC Drivers

For a complete list of supported ODBC drivers by platform, see the Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide, available on the Oracle Technical Network.

For detailed information on the DataDirect Wire Protocol drivers provided with Integration Services, see the DataDirect Connect ODBC Reference in the products\common\ODBC\Merant\6.1\help (for Windows) directory or common/ODBC/Merant/6.1/help (for UNIX) directory. Use Adobe Acrobat Reader, which is available for free download at www.adobe.com to read the PDF files.
Configuring Data Source Names on Windows Systems

To configure data source names on Windows systems, you must start ODBC Administrator and then map an ODBC driver to one or more relational databases or flat file data sources that you will use for creating and storing OLAP models and metaoutlines. Run the ODBC Administrator utility from the Windows Control Panel.

Note: Integration Services does not support data source table names and column names that contain spaces or special characters, such as a period (.). See Appendix B, “Integration Services Limits and Guidelines” for a complete list of the unsupported characters.

The procedures in the following sections show you how to configure data source names to create connections to databases. The examples in the procedures use a database named TBC as the sample Data Source database and a database named TBC_MD as the sample OLAP Metadata Catalog database.

For each site-specific data source name that you configure, obtain the following information from your database administrator:

- The name of the relational database or flat file data source for which you want to configure a data source name
  For example, in the sample application, TBC is the name of the Data Source database; TBC_MD is the name of the OLAP Metadata Catalog database.

- IP address or name of the computer on which the database tables are stored
  You can enter either an alphabetic computer name (for example, sequoia), or an IP address (for example, 172.0.0.125).
  In the sample application, if you are configuring a data source name for the TBC database, use the IP address of the computer where the TBC database tables are stored. If you are configuring a data source name for the TBC_MD database (the sample OLAP Metadata Catalog database), use the IP address of the computer where the TBC_MD tables are stored.

- Port number on which your database listens
  For clarity in presenting the sample application in Integration Services Console, the sample Data Source database and sample OLAP Metadata Catalog database are two separate databases that use two different data source names.

If your data is stored in a relational database, you are not required to use two different databases for your Data Source database and OLAP Metadata Catalog database. Additionally, you are not required to use two different data source names to connect to your Data Source database and OLAP Metadata Catalog database if all the tables reside in the same database.

If your data is stored in one or more flat files, you must still use a relational database when setting up your OLAP Metadata Catalog database.
**Note:** If you plan to use flat files as data sources, you must configure their Data Source Name (DSN). See “Configuring the DataDirect Text Driver on Windows” on page 47 for detailed instructions.

**Accessing ODBC Data Source Administrator in Windows**

For any RDBMS, you must access the Windows ODBC Data Source Administrator to begin the data source name configuration process.

➢ To access the Administrator:

1. On the Windows desktop, select **Start**, then **Settings**, then **Control Panel** to open the **Control Panel** window.
2. In the **Control Panel** window, double-click the **Administrative Tools** icon, and then double-click the **Data Sources (ODBC)** icon to open the **ODBC Data Source Administrator** dialog box.
3. In the **ODBC Data Source Administrator** dialog box, click the **System DSN** tab.

**Configuring DataDirect Wire Protocol Drivers in Windows**

Use the following procedures to configure a data source name for your Data Source database using a DataDirect Wire Protocol driver for Oracle, IBM DB2. Then repeat the procedure to configure a data source name for the relational database that you want to use as your OLAP Metadata Catalog database.

Before beginning the configuration procedure, perform the procedure in “Accessing ODBC Data Source Administrator in Windows” on page 44.

**Note:** You may repeat the configuration procedure to create any number of data source names to relational Data Source databases or OLAP Metadata Catalog databases.

➢ To configure a data source name:

1. In the **ODBC Data Source Administrator**, click **Add** to open the **Create New Data Source** dialog box.
2. In the driver list box of the **Create New Data Source** dialog box, select the appropriate Wire Protocol driver for your RDBMS, and click **Finish**.
   
   The Wire Protocol Driver Setup dialog box for your RDBMS is displayed.
3. In the **Data Source Name** text box, type the name that you want to use for this data source connection.
   
   In the sample application, TBC is the data source name used to connect to the Data Source database called TBC; TBC_MD is the data source name used to connect to the sample OLAP Metadata Catalog database called TBC_MD.
4. Optional: If you want to record a description, in the **Description** text box, type a description that indicates how you use this data source name.
   
   For example, you might type the following to describe the My Business database:
Customers, products, markets

You might type the following to describe the sample application database:

Sample relational data source

You might type the following to describe the sample OLAP Metadata Catalog database:

Sample OLAP Metadata Catalog

5 Make the necessary entries to configure data source names by completing the set of steps appropriate to your RDBMS.

Refer to the following sections for the specific configuration instructions for your RDBMS:

- “IBM DB2 UDB or DB2 OS/390” on page 45
- “Oracle” on page 46

6 Test the connection by following the procedures in “Testing DataDirect Wire Protocol Drivers on Windows” on page 46.

When you finish testing, repeat Step 2 through Step 5 to configure a data source name for an OLAP Metadata Catalog database.

IBM DB2 UDB or DB2 OS/390

Perform the following steps to configure the Wire Protocol driver for IBM DB2 UDB or OS/390:

1 In the IP Address text box, type the address (either the computer name or the numeric IP address) of the computer where the database tables are stored.

2 In the Tcp Port text box, type the port number on which the IBM DB2 database server listens.
   - If you are using IBM DB2 UDB, the default port number is 50000.
   - If you are using IBM DB2 OS/390, the default port number is 446.

3 If you are using IBM DB2 UDB, in the Database Name text box, type the name of the database for which you want to create this data source name.

4 If you are using IBM DB2 OS/390, perform the following actions:
   a. In the Location text box, type the IBM DB2 location name as defined during the IBM DB2 OS/390 installation.
   b. In the Collection text box, type the name that identifies a logical group of database objects (the default is DATADIRECT00).

Note: For IBM DB2 OS/390, you configure a data source name for the Data Source database only. You cannot configure a data source name for the OLAP Metadata Catalog database. OLAP Metadata Catalog functionality is not supported in IBM DB2 OS/390.
To use Integration Services with IBM DB2 OS/390, you must add an entry to the ais.cfg file after you configure the data source name for the source database. See “Adding a Configuration File Entry for IBM DB2 OS/390 in Windows” on page 47 for more information.

5 Click the Bind tab.

6 Accept the default in the Bind tab and click the Create Package button.

For more information about the bind process, see DataDirect Technologies Connect ODBC Reference. The DataDirect documentation is located in common/ODBC/Merant/6.1/help. You can also access the Product Documentation on the DataDirect Web site at http://www.datadirect-technologies.com.

7 In the Login dialog box, type the user name and password associated with the database for which this ODBC data source name is being created, and click OK.

A message is displayed informing you that the package was created successfully.

8 Return to Step 6 in “Configuring DataDirect Wire Protocol Drivers in Windows” on page 44 to complete the configuration process.

Oracle

➢ Perform the following steps to configure the Wire Protocol driver for Oracle:

1 In the Host text box, type the address (either the computer name or the numeric IP address) of the computer where the database tables are stored.

2 In the Port Number text box, type the port number on which the Oracle database server listens.

   The default port number for Oracle is 1521.

3 In the SID text box, type the name of the database for which you want to create this data source name.

4 Return to Step 6 in “Configuring DataDirect Wire Protocol Drivers in Windows” on page 44 to complete the configuration process.

Testing DataDirect Wire Protocol Drivers on Windows

➢ To test a data source name connection:

1 Click Test Connect.

   The Login to Wire Protocol dialog box for your RDBMS is displayed.

2 In the User Name and Password text boxes, enter a valid user name and password for this data source connection, and click OK.

   If an error message is displayed, make the necessary corrections, and click Test Connect again.

   When no errors occur, a dialog box displays the message, Connection established!

3 In the message dialog box, click OK.
You are returned to the ODBC Wire Protocol Driver Setup dialog box.

4 In the ODBC Wire Protocol Driver Setup dialog box, click OK.

You are returned to the System DSN tab of the ODBC Data Source Administrator dialog box. The data source name that you entered and the driver that you mapped to the data source name are displayed in the System Data Sources list box.

5 Repeat Step 1 through Step 5 in “Configuring DataDirect Wire Protocol Drivers in Windows” on page 44 to configure a data source name for an OLAP Metadata Catalog database.

6 When you have completed configuration of all needed data source names, click OK to close the ODBC Data Source Administrator dialog box.

Adding a Configuration File Entry for IBM DB2 OS/390 in Windows

If you are using IBM DB2 OS/390, you must add an entry to the ais.cfg file to support Integration Services functionality. You must add entries for each data source name for IBM DB2 OS/390 that you have configured.

Note: For IBM DB2 OS/390, you configure a data source name for the Data Source database only. You cannot configure a data source name for the OLAP Metadata Catalog database. OLAP Metadata Catalog functionality is not supported in IBM DB2 OS/390.

To add an entry for IBM DB2 OS/390 to the ais.cfg file:

1 On the computer running Integration Server, use a text editor to open ais.cfg.

2 Add a new line for IBM DB2 OS/390 using the data source name or names you have configured.

   Add separate lines for each data source name configured for IBM DB2 OS/390. The syntax is:

   [DS:dsn:390]

   For example, using the sample application database, TBC, the following is the entry to the ais.cfg file:

   [DS:TBC:390]

3 Save and close ais.cfg.

Configuring the DataDirect Text Driver on Windows

Use the following procedure to configure a data source name for your Data Source database that consists of one or more ASCII text files using a DataDirect Text driver.

Note: Integration Services does not support data source table names and column names that contain spaces or special characters, such as a period (.). See Appendix B, “Integration Services Limits and Guidelines” for a complete listing of the unsupported characters.
For more information on configuring the ODBC driver settings for text files, see the *DataDirect Connect ODBC Reference* in the `common/ODBC/Merant/6.1/help` directory.

**Note:** You must complete the steps in “Configuring DataDirect Wire Protocol Drivers in Windows” on page 44 to configure a data source name for the relational database that you want to use as your OLAP Metadata Catalog database.

Before beginning the configuration procedure, perform the procedure in “Accessing ODBC Data Source Administrator in Windows” on page 44.

To configure a data source name for a flat file data source:

1. **Click** Add **to open the Create New Data Source dialog box.**
2. **In the driver list box of the Create New Data Source dialog box,** select the Merant OEM 6.1 TextFile driver, and click Finish.
   
   The General tab of the ODBC Text Driver Setup dialog box for flat file configuration is displayed.
3. **In the Data Source Name text box,** type a name for the flat file data source.
4. **Optional:** If you want to record a description, **in the Description text box,** type a description that indicates how you use this data source name.
   
   For example, you might type the following to describe the My Business database:
   
   Customers, products, markets
5. **In the Database Directory text box,** type the path to the folder where the flat files are located.
6. **In the Default Table Type drop-down list,** select whether the text file is comma-separated, tab-separated, character-separated, fixed length, or stream.

   **Note:** If your text files are character-separated, type the character that is used as a delimiter in the Delimiter Character text box.
7. If the first line of your text files contains the column names of the database table, **select the Column Names in First Line check box;** otherwise, proceed to Step 8.
8. **Select the Advanced tab.**
9. **In the Rows to Scan text box,** type the number of rows that should be scanned for the driver to determine the data types in the file.
   
   The default is 25 rows. If 0 is entered, all rows in the file are scanned.
10. **In the Action for Undefined Tables group,** choose one of the following actions that the driver should take when it encounters a file that has not been defined.
    
    - **Prompt for Definition.** The text driver prompts you when it encounters a file that has not been defined.
    - **Guess Definition.** The text driver analyzes the file and guesses the file’s format.
Complete the following steps to define the structure of your data source text files:

11. Click the Define button to display the Define File dialog box.
12. Navigate to the folder that contains your data source text files.
13. Select a text file to define and click Open to display the Define Table dialog box.
14. In the Table text box, type a table name to associate with this text file.
   The table name you define is the table name that will be displayed in the left frame of the OLAP Model main window.
15. If the first line of text in the file contains the column names, select the Column Names in the First Line check box.
   If the first line of the text file does not contain the column names, then do not select this check box.
16. From the Table Type drop-down list, select the delimiter that is used in the text file.
17. If you select Character in the Table Type drop-down list in Step 16, specify the character in the Delimiter Character text box.
   You can specify any printable character except single and double quotation marks.
18. In the Decimal Symbol text box, specify either a comma or period as a decimal separator to use when data is stored.

Perform one of the following actions to define the column names:

- If you specified in Step 16 that your text files were comma-separated, tab-separated, or character-separated, click the Guess button to have the text driver guess at the column names and display them in the list box under Column Information.
- If you specified in Step 16 that your text files were fixed-length or stream type, click the Parse button to display the Parse Table dialog box, where you define the table column names.
  For more information on using the Parse Table dialog box, see the DataDirect Connect ODBC Reference. You can also view the DataDirect online help by clicking the Help button in the Parse Table dialog box.
- If you want to define the values and fields for each column manually, click Add to add the column name to the list box under Column Information.
  For each table that you add manually, you must specify the data type, data mask (where appropriate), column precision, column scale, length, and offset. For more information on these settings, see the DataDirect Connect ODBC Reference. You can also view the DataDirect online help by clicking the Help button in the Define Table dialog box.

Optional: Modify or delete any of the existing column definitions by selecting a column name in the list box under Column Definition and clicking the Modify or Remove button.

Click OK to define the table.

Complete Step 11 through Step 14 for each data source table that you want to define.
When you have finished defining each data source table, click **Cancel** in the **Define File** dialog box to return to the **Advanced** tab of the **ODBC Text Driver Setup** dialog box.

Specify the remaining settings on the **Advanced** tab.

For information on the remaining settings, refer to the *DataDirect Connect ODBC Reference* in the common/ODBC/Merant/6.1/help directory. You can also view the DataDirect online help by clicking the Help button in the Advanced tab of the ODBC Text Driver Setup dialog box.

## Configuring the SQL Server ODBC Driver for Microsoft SQL Server

Use the following procedure to configure a data source name for your Data Source database using the SQL Server driver. Then repeat the procedure to configure a data source name for your OLAP Metadata Catalog database.

**Note:** You may repeat the configuration procedure to create any number of data source names to relational Data Source databases or OLAP Metadata Catalog databases.

The procedures that follow provide information on basic configuration. If you are using more advanced options, refer to the online help for each wizard screen for assistance in completing the wizards.

To use the SQL Server driver to configure a data source name:

1. **Access ODBC Data Source Administrator** by following the procedure in “Accessing ODBC Data Source Administrator in Windows” on page 44.
2. Click **Add** to open the **Create New Data Source** dialog box.
3. In the driver list box of the **Create New Data Source** dialog box, select the **SQL Server** driver, and click **Finish**.
   The **Create a New Data Source to SQL Server** dialog box is displayed.
4. In the **Name** text box, type the data source name that you want to use for this data source connection.
   In the sample application, TBC is the data source name used to connect to the Data Source database called TBC; TBC_MD is the data source name used to connect to the sample OLAP Metadata Catalog database called TBC_MD.
5. **Optional:** In the **Description** text box, type a description that indicates how you use this data source name.
   For example, you might type the following names to describe the My Business database:
   **Customers, products, markets**
   You might type the following statement to describe the sample application database:
   **Sample relational data source**
You might type the following statement to describe the sample OLAP Metadata Catalog database:

Sample OLAP Metadata Catalog

6 In the Server text box, type the address (either the computer name or the numeric IP address) of the computer where the database tables are stored.

7 Click Next.

The second wizard screen is displayed.

8 Select the appropriate option button for the method to use for verifying login IDs:
   - With Window NT authentication using the network login ID.
   - With SQL Server authentication using a login ID and password entered by the user.

9 Optional: If you are using a network library other than TCP/IP (the default) to connect to the database server, perform the following steps:
   a. Click the Client Configuration button.
   b. In the Network libraries list, select the appropriate option button for the library that you are using to connect to the database server computer, and click OK.
      You are returned to the second wizard screen.

10 Optional: Select the Connect to SQL Server to obtain default settings for the additional configuration check box.
   When this check box is selected, the driver obtains default settings from Microsoft SQL Server that it uses to complete additional setup screens in the wizard.
   When this check box is clear, the driver uses standard defaults to complete the additional setup screens in the wizard.

11 Click Next.

The third wizard screen is displayed.

12 Select the Change the default database to check box and then type or select the name of the database for which you want to create this data source name.
   For example, in the sample application, TBC is the name of the Data Source database; TBC_MD is the name of the OLAP Metadata Catalog database.

13 Click Next.

The fourth wizard screen is displayed.

14 Make any entries appropriate to your system configuration and click Finish.

15 Repeat Step 2 through Step 14 to configure a data source name for an OLAP Metadata Catalog.

16 When you have completed configuring all needed data sources, click OK to close the ODBC Data Source Administrator dialog box.
Configuring the Teradata ODBC Driver for Teradata

Use the following procedure to configure a data source name for your Data Source database using the Teradata driver. Then repeat the procedure to configure a data source name for the relational database that you want to use as your OLAP Metadata Catalog database.

**Note:** You may repeat the configuration procedure to create any number of data source names to Data Source databases or OLAP Metadata Catalog databases.

To configure a data source name:

1. Access the ODBC Data Source Administrator by following the procedure in "Accessing ODBC Data Source Administrator in Windows" on page 44.
2. Click Add to open the Create New Data Source dialog box.
3. In the driver list box of the Create New Data Source dialog box, select Teradata, and click Finish.

   The ODBC Driver Setup for Teradata RDBMS dialog box is displayed.
4. In the Name text box, type the data source name that you want to use for this data source connection.
   In the sample application, TBC is the data source name used to connect to the Data Source database called TBC; TBC_MD is the data source name used to connect to the sample OLAP Metadata Catalog database called TBC_MD.
5. **Optional:** In the Description text box, type an description that indicates how you use this data source name.
   For example, you might type the following to describe the My Business database:
   Customers, products, markets
   You might type the following to describe the sample application database:
   Sample relational data source
   You might type the following to describe the sample OLAP Metadata Catalog database:
   Sample OLAP Metadata Catalog
6. In the Teradata Server Info text box, type the address (either the computer name or the numeric IP address) of the computer where the database tables are stored.
7. In the Default Database text box, type the name of the database for which you want to create this data source name.
   For example, in the sample application, TBC is the name of the Data Source database; TBC_MD is the name of the OLAP Metadata Catalog database.
8. **Optional:** Enable the X Views option for Teradata.

   You can limit the number of data sources displayed in the left frame of an OLAP Model main window based on the security of the Teradata user ID that was used to connect. Follow steps a through c to enable the X Views option:
   a. Click the Options button.
b. Check the Use X Views check box.
c. Click OK to return to the ODBC Driver Setup for Teradata RDBMS dialog box.

9 Click OK to return to the System DSN tab of the ODBC Data Source Administrator dialog box.

The data source name that you entered and the driver that you mapped to it are displayed in the System Data Sources list box.

10 Repeat Step 2 through Step 9 above to configure a data source name for an OLAP Metadata Catalog.

Note: You may repeat the above procedure to create any number of data source connections to relational data sources or OLAP Metadata Catalogs.

11 When you have completed configuring all needed data sources, click OK to close the ODBC Data Source Administrator dialog box.

Editing a Data Source Name

To edit configuration information for a data source name:

1 Access the ODBC Data Source Administrator by following the procedures in "Accessing ODBC Data Source Administrator in Windows" on page 44.

2 Select the data source name and click Configure to open the driver setup dialog box or wizard specific to your RDBMS.

3 Correct any information that you want to change.

Configuring Data Source Names on UNIX Systems

ODBC Administrator is not available on UNIX systems. On AIX, HP-UX, Solaris, and Linux, you must manually set environment variables and edit the odbc.ini file. You can edit the odbc.ini file (using a text editor such as vi) to configure, add, or change data sources names, and to add or change drivers.

Note: Integration Services does not support data source table names and column names that contain spaces or special characters, such as a period (.) See Appendix B, "Integration Services Limits and Guidelines" for a complete listing of the unsupported characters.

Flat File Data Sources: If you are using a flat file data source, you must define the structure of the text files in your data source in a QETXT.INI file. The process for creating the QETXT.INI file for UNIX operating systems is described in the DataDirect Connect ODBC Reference in common/ODBC/Merant/6.1/help.
Configuring Environment Variables

On UNIX systems, you must set environment variables to enable access to ODBC core components. The `is.sh` and `is.csh` shell scripts that set the required variables are provided in the Integration Services home directory.

You must run one of the scripts before starting Integration Server and using ODBC to connect to Data Source databases and OLAP Metadata Catalog databases. You can include the scripts in the login script for the user name that you use to run Integration Server.

**Teradata:** If you are using Teradata drivers, you must set environment variables to point to the location of the installed drivers.

**Note:** Use the `env` command to verify environment settings.

Editing the odbc.ini File

You configure data source names for a Data Source database or OLAP Metadata Catalog database in an `odbc.ini` file. In one section of the file, add a name and description for the ODBC data source. In a separate, newly created section of the file, provide the ODBC driver path, file name, and all other required driver settings.

The Integration Services installation program installs a sample `odbc.ini` file in the `$/ORACLE_EPM_HOME/common/ODBC/Merant/6.1` directory. The file contains generic ODBC connection and configuration information for supported ODBC drivers. Use the file as a starting point to map the ODBC drivers that you use to the Data Source database and OLAP Metadata Catalog database.

**IBM DB2:** For each server computer to which you are connecting using the DataDirect Wire Protocol driver, you must create a bind package. DataDirect supplies a program that creates the bind package for you. See “Creating IBM DB2 Bind Packages for DataDirect Wire Protocol Driver Connections” on page 61 for instructions on using the DataDirect-supplied program.

**Teradata:** When configuring a data source name, the data source description must be set to `tdata.sl` (HP-UX) or `tdata.so` (AIX and Solaris). Specific examples for AIX, HP-UX, and Solaris operating systems are shown in “Examples of ODBC Settings for Teradata” on page 60.

**Tip:** The procedure in this section shows you how to configure a data source name by manually editing the `odbc.ini` file.

**Note:** If you use a file other than the `odbc.ini` file (located by default in `common/ODBC/Merant/6.1`), be sure to set the ODBCINI environment variable to the name of the file that you use.

To add an ODBC data source to an `odbc.ini` file:

1. On the computer running Integration Server, open the `odbc.ini` file by using a text editor such as vi.
2 Locate the section starting with [ODBC Data Sources] and add a new line with the data source name and description; for example:

mydata=data source for analysis

To minimize confusion, the data source name can match the name of the database in the RDBMS.

3 Add a new section to the file by creating a new line with the new data source name enclosed in brackets; for example:

[mydata]

4 On the lines following the data source name, add the full path and file name for the ODBC driver required for this data source and any other required ODBC driver information.

Use the examples shown in the following sections as guidelines for specific RDBMSs.

Note: Ensure that the ODBC driver file actually exists in the location that you specify for the Driver= setting.

5 Add a new section to the file by creating a new line with ODBC enclosed in brackets; for example:

[ODBC]

6 Under the [ODBC] heading, add the full path to the location where the /lib and /messages directories are contained.

Use the examples shown in the following sections as guidelines for specific RDBMSs.

7 When you finish editing odbc.ini, save the file and exit the text editor.

For information about the odbc.ini file and the ODBC driver settings for each RDBMS or flat file data source, see the DataDirect Connect ODBC Reference in the common/ODBC/Merant/6.1/help directory. For information about vendor-supplied ODBC driver settings, refer to the installation documentation for the vendor-supplied ODBC drivers.

Flat File Data Sources: If you are using a flat file data source, you must define the structure of the text files in your data source in a QETXT.INI file. The process for creating the QETXT.INI file for UNIX operating systems is described in the DataDirect Connect ODBC Reference in common/ODBC/Merant/6.1/help.

Examples of ODBC Settings for IBM DB2 UDB

The following example illustrates how you might edit odbc.ini to connect to a Data Source database, such as db2data on IBM DB2 (on AIX), using a DataDirect Wire Protocol driver. You can make a separate entry to connect to the OLAP Metadata Catalog database.

Use the vi $ODBCINI command to edit the odbc.ini file and insert the following statements:

[ODBC Data Sources]
db2data=DB2 Source Data on AIX
...
[db2data]
Driver=/EPM_ORACLE_HOME/common/ODBC/Merant/6.1/Drivers/ardb225.so
IpAddress=isaix7
Example of ODBC Settings for IBM DB2 OS/390

The following example illustrates how you might edit odbc.ini to connect to a Data Source database, such as db2data on IBM DB2 version 8.2 (on AIX), using a DataDirect Wire Protocol driver. You can make a separate entry to connect to the OLAP Metadata Catalog database.

Use the vi $ODBCINI command to edit the odbc.ini file and insert the following statements:

```
[ODBC Data Sources]
os390=DB2 OS/390 Source Data
...[os390]
Driver=/EPM_ORACLE_HOME/common/ODBC/Merant/6.1/Drivers/ARdb222.so
Collection=DATADIRECT00
Database=
DynamicSections=100
GrantAuthid=PUBLIC
GrantExecute=1
IpAddress=isaix21
IsolationLevel=CURSOR_STABILITY
Location=DALLAS
Package=PACK42
PackageOwner=
TcpPort=446
WithHold=446
[ODBC]
IANAAppCodePage=4
```

56 Configuring Data Sources
InstallDir=common/ODBC/Merant/6.0
Trace=0
TraceDll=common/ODBC/Merant/6.0/lib/odbctrac.so
TraceFile=odbctrace.out
UseCursorLib=0

**Note:** To use Integration Services with IBM DB2 OS/390, you must add an entry to the ais.cfg file after you configure the data source name for the source database.

**Example of ODBC Settings for Oracle**

The following example illustrates how you might edit odbc.ini to connect to a Data Source database, oradata, on Oracle (on Solaris), using a DataDirect ODBC driver. You can make a separate entry to connect to the OLAP Metadata Catalog database.

```
[ODBC Data Sources]
oradata=Oracle Source Data on Solaris
...
[oradata]
Driver=/EPM_ORACLE_HOME/common/ODBC/Merant/6.1/Drivers/arora25.so
Description=my oracle source
HostName=oraclehost
SID=tbc1
PortNumber=1521
[ODBC]
IANAAppCodePage=4
InstallDir=common/ODBC/Merant/6.1
Trace=0
TraceDll=common/ODBC/Merant/6.1/lib/odbctrac.so
TraceFile=odbctrace.out
UseCursorLib=0
```

**Tip:** Run `ivtestlib` to verify that the environment is set to run the correct ODBC driver file. For example, run `ivtestlib` and paste the path and file name that follow `Driver=` in the `odbc.ini` file that you edited.

**Example of ODBC Settings for Flat File Data Source**

The following example illustrates how you might edit `odbc.ini` to connect to a Data Source database that consists of one or more flat files. The example also includes descriptions for each line item in parentheses. You can make a separate entry to connect to the OLAP Metadata Catalog database, which must be stored in a relational database.

Also included is an example of the `QETXT.INI` file. The example uses shows how you would configure three tables from the TBC database: Product, Region, Market, and an abbreviated version of Sales.

**Note:** Integration Services does not support data source table names and column names that contain spaces or special characters, such as a period (.). See Appendix B, “Integration Services Limits and Guidelines” for a complete listing of the unsupported characters.
Note: You must use a relational database as your OLAP Metadata Catalog database and configure it as described in “Editing the odbc.ini File” on page 54.

[ODBC Data Sources]
tbc_ff=TBC flat file data source
...
[tbc_ff]
Driver=/EPM_ORACLE_HOME/common/ODBC/Merant/6.1/Drivers/ARtxt25.so
   (location of a driver)
Description=DataDirect 6.1 TextFile(*.*)
AllowUpdateAndDelete=0 (a variable edited by user)
ApplicationUsingThreads=1 (a variable edited by user)
CacheSize=64 (a variable edited by user)
CenturyBoundary=20 (a variable edited by user)
Database=/EPM_ORACLE_HOME/flat_files/stream (a data file location)
DataFileExtension=TXT (a variable edited by user, default is TXT)
DecimalSymbol=. (a variable edited by user, default is dot(.))
Delimiter=~ (a variable edited by user)
FileOpenCache=0 (a variable edited by user)
FirstLineNames=1 (a variable edited by user)
IntlSort=1 (a variable edited by user)
ScanRows=5 (a variable edited by user)
TableType=Character (a variable edited by user)
UndefinedTable=GUESS (a variable edited by user)

[ODBC]
IANAAppCodePage=4
InstallDir=common/ODBC/Merant/6.1
Trace=0
TraceDll=common/ODBC/Merant/6.1/Drivers/odbctrac.so
TraceFile=odbctrace.out
UseCursorLib=0

For more information on editing the ODBC.INI file in UNIX to configure a flat file data source see the DataDirect Connect ODBC Reference in /common/ODBC/Merant/6.1/help.

Example of QETXT.INI for Flat File Data Sources

The following is an example of the QETXT.INI file. The example shows how you would configure three tables from the TBC database: Product, Region, Market, and an abbreviated version of Sales.

[Defined Tables]
PRODUCT.txt=Product
REGION.txt=REGION
SALESFACTSHORT.txt=SalesFactShort
MARKET.txt=Market

[Product]
FILE=PRODUCT.txt
FLN=1
TT=Comma
Charset=ANSI
DS=.
FIELD1=PRODUCTID,NUMERIC,9,0,9,0,
FIELD2=FAMILYID,NUMERIC,8,0,8,0,
For more information on creating the QETXT.INI file in UNIX to configure tables for a flat file data source see the DataDirect Connect ODBC Reference in common/ODBC/Merant/6.1/help.

**Example of ODBC Settings for Microsoft SQL Server**

The following example illustrates how you might edit odbc.ini to connect to a Data Source database, such as sqldata, on Microsoft SQL Server (on HP-UX), using a vendor-supplied ODBC driver. You can make a separate entry to connect to the OLAP Metadata Catalog database.

```
[ODBC Data Sources]
sqldata=SQL Server 2005 Source Data on HP-UX
...
sqldata
Driver=/EPM_ORACLE_HOME/common/ODBC/Merant/6.1/Drivers/ARsqls25.so
```
Examples of ODBC Settings for Teradata

The following example illustrates how you might edit odbc.ini to connect to a Data Source database, tbc, on Teradata (on AIX), using the Teradata ODBC driver. You can make a separate entry to connect to the OLAP Metadata Catalog database. The data source description in [ODBC Data Sources] must be set to tdata.so.

[ODBC Data Sources]
tbc=tdata.so
...
[tbc]
Driver=/usr/odbc/drivers/tdata.so
Description=NCR 3600 running Teradata V2R5
DBCName=139.64.140.45
DefaultDatabase=tbc
UseXViews=Yes
[ODBC]
IANAAppCodePage=4
InstallDir=common/ODBC/Merant/6.1
Trace=0
TraceDll=common/ODBC/Merant/6.1/lib/odbctrac.so
TraceFile=odbctrace.out
UseCursorLib=0

HP-UX

The following example illustrates how you might edit odbc.ini to connect to the same Data Source database, tbc, on Teradata (on HP-UX), using the Teradata ODBC driver. You can make a separate entry to connect to the OLAP Metadata Catalog database. The data source description in [ODBC Data Sources] must be set to tdata.sl.

[ODBC Data Sources]
tbc=tdata.sl
...
[tbc]
Driver=/usr/odbc/drivers/tdata.sl
Description=NCR 3600 running Teradata V2R5
DBCName=139.64.140.45
DefaultDatabase=tbc
UseXViews=Yes
[ODBC]
IANAAppCodePage=4
InstallDir=common/ODBC/Merant/6.1
Trace=0
TraceDll=common/ODBC/Merant/6.1/lib/odbctrac.so
TraceFile=odbctrace.out
UseCursorLib=0
Solaris:

The following example illustrates how you might edit `odbc.ini` to connect to the same Data Source database, `tbc`, on Teradata (on Solaris), using the Teradata ODBC driver. You can make a separate entry to connect to the OLAP Metadata Catalog database. The data source description in [ODBC Data Sources] must be set to `tdata.so`.

```
[ODBC Data Sources]
tbc=tdata.so
...
[tbc]
Driver=/usr/odbc/drivers/tdata.so
Description=NCR 3600 running Teradata V2R5
DBCName=139.64.140.45
DefaultDatabase=tbc
UseXViews=Yes
[ODBC]
IANAAppCodePage=4
InstallDir=common/ODBC/Merant/6.1
Trace=0
TraceDll=common/ODBC/Merant/6.1/lib/odbctrac.so
TraceFile=odbctrace.out
UseCursorLib=0
```

**Note:** The `UseXViews` parameter is optional on all UNIX platforms. The `UseXViews` parameter limits the number of data sources displayed in the left frame of an OLAP Model main window based on the security of the Teradata user ID that was used to connect.

---

**Creating IBM DB2 Bind Packages for DataDirect Wire Protocol Driver Connections**

In the UNIX environment, you must create a bind package for each connection between Integration Server and an IBM DB2 database that you make using the DataDirect (formerly MERANT) Wire Protocol driver. Without a bind package for each connection, the Wire Protocol driver will not work properly. DataDirect supplies a command line program to create the bind packages.

1. On the computer running Integration Server, open a command shell window.
2. In the command shell window type:
   ```
   bind19 dsn
   ```
Replace dsn with the data source name for which you are creating this bind package. For example, to create a bind package for the data source name TBC, type:

bind19 TBC

Note: You are prompted for a user name and password if this information is not stored in the system information file.

If successful, a message is displayed stating that the package was created and bound.

For troubleshooting information, access the DataDirect Web site at http://www.datadirect-technologies.com and select the Product Documentation link.

3 Repeat Step 2 for all bind packages you want to create.

4 When all bind packages are created, close the command shell window.

Adding a Configuration File Entry for IBM DB2 OS/390

If you are using IBM DB2 OS/390, you must add an entry to the ais.cfg file to support Integration Services functionality. You must add entries for each data source name for IBM DB2 OS/390 that you have configured.

Note: For IBM DB2 OS/390, you configure data source name for the Data Source database only. You cannot configure a data source name for the OLAP Metadata Catalog database. OLAP Metadata Catalog functionality is not supported in IBM DB2 OS/390.

To add an entry for IBM DB2 OS/390 to the ais.cfg file:

1 On the computer running Integration Server, use a text editor to open ais.cfg.

2 Add a new line for IBM DB2 OS/390 using the data source name or names you have configured.

Add separate lines for each data source name configured for IBM DB2 OS/390. The syntax is:

[DS:dsn:390]

For example, using the sample application database, TBC, the following is the entry to the ais.cfg file:

[DS:TBC:390]

3 Save and close ais.cfg.

After Configuration of Data Source Names

After you configure data source names for the Data Source database and OLAP Metadata Catalog database, you can connect to databases from Integration Services. You can then create, modify, and save OLAP models and metaoutlines. For information on viewing TBC tables, columns,
OLAP models, and metaoutlines in Integration Services Console, see “Viewing TBC Tables and Columns” on page 113.

If you plan to use flat (text) files as data source, you must set up data source names for those files. See “Configuring Data Source Names on Windows Systems” on page 43 “Configuring Data Source Names on UNIX Systems” on page 53.

**Note:** The SQL Server ODBC driver may time out during a call to an SQL Server database, particularly during a data load. If a timeout occurs, try again when the database is not busy. Increasing the driver time-out period may avoid this problem. For more information, see the ODBC documentation for the driver that you are using.

For more information on ODBC connection problems and solutions, see Chapter 7, “Troubleshooting ODBC and Connections”.

### Connecting to Server Components and Data Sources

Whether you are installing Integration Services for the first time or upgrading from a previous release, after you start the console, the OLAP Metadata Catalog Setup dialog box is displayed. Using this dialog box, you can either create a catalog or upgrade an existing catalog.

**Note:** An OLAP Metadata Catalog must be created and configured before you can connect to it. For information on configuring a relational data source for an OLAP Metadata Catalog, see Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide. For information on creating OLAP Metadata Catalogs, see Chapter 3, “Creating, Upgrading, and Deleting OLAP Metadata Catalogs.”

After the process of creating or upgrading an OLAP Metadata Catalog is complete, you can begin to create OLAP models and metaoutlines. You must connect to Integration Server, to an external data source, and to the OLAP Metadata Catalog where you want to store the OLAP models and metaoutlines that you create.

To use Integration Services Console to perform data loads, member loads, or member and data loads, you must connect the client software to an additional server component, Essbase Server.

You are prompted by the Login dialog box to connect to an Integration Server and an OLAP Metadata Catalog. Optionally, in the Login dialog box, you can also connect to an instance of Essbase Server and create default connection settings for both server components.

For information on troubleshooting connections to server components, see Chapter 7, “Troubleshooting ODBC and Connections”.

### Connecting to Integration Server

Integration Server manages access to OLAP Metadata Catalogs and to data sources. A catalog stores information that defines the external data source and the structure of OLAP model and
metaoutline elements. To create an OLAP model or a metaoutline, you must connect to Integration Server.

**Note:** The Integration Server system administrator must provide a user name and password to an OLAP Metadata Catalog before you can connect to the server.

**Note:** It is highly recommended that you do not connect the Integration Server and Integration Services Console via a wide area network (WAN). Doing so may cause severe performance degradation.

> To connect to Integration Server:

1. To display the Login dialog box from the OLAP Model main window of the Integration Services Console, select Connections, then OLAP Metadata Catalog, then Connect.

   **Note:** The Login dialog box is displayed automatically when you start Integration Services Console. You can redisplay the Login dialog box by using the procedures described in this topic.

2. In the Server drop-down list under Integration Services, enter or select the name or IP address of an Integration Server computer; for example, cypress.

   If the port number on which Essbase Integration Server communicates with the console has been set to a value other than the default, you must type the server name or IP address and type the nonstandard port number, separated by a colon; for example:

   cypress:3390

3. In the OLAP Metadata Catalog drop-down list, select the data source name of an OLAP Metadata Catalog; for example, TBC_MD for the sample TBC (The Beverage Company) catalog database.

   Oracle:TBC_MD

4. In the Code Page drop-down list box, select the code page of the language you want to use during the current Integration Services Console session.

   The code page is a portion of the locale which identifies the local language and cultural conventions such as the formatting of currency and dates and the sort order of data.

5. In the User Name drop-down list box under Integration Services, type or select your user name.

   It is necessary to type a user name the first time it is used. After a user is successfully connected to Integration Services, the user name is then displayed in the User Name drop-down list.

6. In the Password text box under Essbase Integration Server, type your password.

   **Note:** Use the same user name and password that you use to connect directly to the database server and to access the database that contains the OLAP Metadata Catalog.

7. Click OK or enter information for connecting to an instance of Essbase Server.
Connecting to Essbase Server

Essbase Server creates and manages Essbase databases. You do not have to connect Integration Services Console to Essbase Server to create an OLAP model or metaoutline, but you must connect to Essbase Server to view a sample Essbase outline or to load members or data into an Essbase database.

In the Login dialog box, you can specify the information for connecting to Essbase Server, but Integration Services Console does not connect to Essbase Server until you attempt to view a sample Essbase outline or to load members or data.

If you plan to use Essbase Integration Services Console on a client computer that is outside the firewall for your network, and the console requires that Integration Server and Essbase Server be located inside the firewall for your network, you must log on to Essbase Server with a name or an IP address that connects from both sides of the firewall. The system administrator provides you with this name or IP address.

Note: The system administrator for Essbase Server must provide you with a user name and password before you can connect.

➢ To set connection information for an instance of Essbase Server:

1. In the OLAP Metaoutline main window, select Connections, then OLAP Metadata Catalog, then Connect to display the Login dialog box.

   Note: The Login dialog box is displayed automatically when you start Integration Services Console. You can redisplay the Login dialog box at any time while working in the OLAP Metaoutline main window by using the procedures described in this topic.

2. In the Server text box under Essbase Server, type or select the name of a computer that is running Essbase Server; for example, sequoia.

   Note: If you are using Integration Services Console on a client computer that is outside the firewall for your network, and you require access to instances of Integration Server and Essbase Server located inside the firewall for your network, be sure to use a name or an IP address for the instance of Essbase Server that is accessible from both sides of the firewall.

3. In the User Name drop-down list box under Essbase Server, type or select your Essbase user name.

   It is necessary to type a user name the first time it is used. After a user is successfully connected to Essbase Server, the user name is then displayed in the User Name drop-down list.

4. In the Password text box under Essbase Server, type your Essbase password.

5. Click OK.
Setting Connection Defaults

If you typically use a specific instance of Integration Server, OLAP Metadata Catalog, and Essbase Server, you can save the settings as the default. After you set default connection information, you can reuse these settings without reentering the information.

If you use multiple instances of Integration Server and OLAP Metadata Catalogs, you can also set multiple relationships and select the appropriate one to use so that you do not need to reenter the information.

To set connection defaults:

1. To display the Login dialog box, in the OLAP Model main window or OLAP Metaoutline main window, select Connections, then OLAP Metadata Catalog, then Connect.
2. In the Login dialog box, click Set Login Defaults.
   - The Set Login Defaults dialog box is displayed.
3. In the Server text box, type or select the name of an Integration Server computer; for example, cypress.
4. In the OLAP Metadata Catalog text box, type or select the name of an OLAP Metadata Catalog; for example, TBC_MD for the sample TBC database.
5. In the Code Page drop-down list box, select the code page of the language you want to use for this Integration Services Console session.
   - The code page is a portion of the locale which identifies the local language and cultural conventions such as the formatting of currency and dates and the sort order of data.
6. In the User Name text box, type or select a user name for the OLAP Metadata Catalog.
7. Take one of the following actions:
   - To set the connection defaults, click Set Default.
   - To save the connection information without setting the information as the default, click Add to List.
8. In the Default Server text box, type or select the name of an Essbase Server computer; for example, cypress.
9. In the Default User Name text box, type or select a user name for the instance of Essbase Server; for example, TBC.
10. Click OK to return to the Login dialog box.

Connecting to Data Sources

A data source is an external data repository—typically a large database—whose data you want to analyze by using Essbase. You must connect Integration Services to all data sources that you plan to use in creating OLAP models and metaoutlines and in loading data into an Essbase database.

Note: You can connect to more than one relational data source to create OLAP models.
To connect to a data source:

1. In the Integration Services Console **Welcome** dialog box, click the appropriate icon to create a new OLAP model or metaoutline. Alternatively, select the **Existing** or **Recent** tab and double-click an OLAP model or metaoutline to open it for editing.

   The Data Source dialog box is displayed.

2. In the **Data Source** drop-down list, select the data source to be used; for example, TBC in the sample application.

   An ODBC data source must be created on the computer that is running Integration Server for any external data sources that you want to use. If the data source that you need is not visible in the scroll list, contact the Integration Services system administrator.

   For more information about troubleshooting server and data source connections, see Chapter 7, “Troubleshooting ODBC and Connections”.

   The following example is a Net Service Name stanza that defines TBC in the tnsnames.ora file:

   ```
   TBC =
   (DESCRIPTION =
   (ADDRESS_LIST =
   (ADDRESS = (PROTOCOL = TCP)(HOST = labmachine2)(PORT = 1521))
   )
   (CONNECT_DATA =
   (SERVICE_NAME = orasid)
   )
   )
   ```

   For Oracle using onames, in the example above, TBC is the Schema Name used to connect to an Oracle database. This is the database identifier that you use when you are using SQL *Plus to connect to a database.

   **DataDirect Drivers:** If you are using DataDirect drivers with Oracle, pick a data source name from the Data Source drop-down list.

3. In the **Code Page** drop-down list box, select the code page of the language you want to use during the current Integration Services Console session, and click **OK**.

   The code page is a portion of the locale which identifies the local language and cultural conventions such as the formatting of currency and dates and the sort order of data.

4. In the **User Name** drop-down list box, type or select your user name.

   **Note:** If you are using a flat file data source, no user name is required.

5. In the **Password** text box, type your password.

   **Note:** If you are using a flat file data source, no password is required.

6. Click **OK**.

   The left frame of the OLAP Model main window initially displays information about the first data source to which you connected. Data source information is displayed hierarchically by data source name and owner name.
For each owner name, data is further sorted and organized by tables, views, and synonyms. You can use the listed source tables to create an OLAP model or use additional source tables by connecting to other data sources.

Information about subsequent data sources to which you connect is displayed in the same manner as information about the first connected data source was displayed; that is, by data source name and owner name, then tables, views, and synonyms.

Expand the plus sign, +, to display tables, views, and synonyms contained in the data source.

If you want to connect to additional data sources, complete the steps in the following procedure:

1. **Select Connections, then Add Data Sources.**
   - The Data Source dialog box is displayed. You can connect to any number of available data sources without closing this dialog box.

2. **In the Data Source drop-down list, select the additional data source to which you want to connect.**
   - If you are using DataDirect drivers with Oracle, pick a data source name from the Data Source drop-down list.

3. **In the Code Page drop-down list box, select the code page of the language you want to use during the current Integration Services Console session, and click OK.**

4. **In the User Name drop-down list box, type or select your user name.**
   - It is necessary to type a user name the first time it is used. After a user is successfully connected to a data source, the user name is then displayed in the User Name drop-down list.

   **Note:** If you are using a flat file data source, no user name is required.

5. **In the Password text box, type your password and click Connect.**

   **Note:** If you are using a flat file data source, no password is required.

The left frame of the OLAP Model main window displays information about the data source to which you just connected along with information about the first data source to which you connected.

6. **Repeat Step 2 through Step 3 for each data source to which you want to connect.**
   - As you connect to additional data sources, the left frame of the OLAP Model main window displays information about all data sources to which you are connected.

7. **When you finish connecting to all appropriate data sources, click Close.**
Mapping JDBC Data Sources for Windows (Teradata Only)

Teradata users must map a JDBC data source to a relational database in the ais.cfg file in order to use the XML Import/Export feature of Integration Services. Teradata requires JDBC Type 3.

To map a data source, edit the ais.cfg file (located in EPM_ORACLE_HOME\products\Essbase\eis\server\bin) using any text editor. If the ais.cfg file does not exist, use a text editor to create and edit it, then save it in EPM_ORACLE_HOME\products\Essbase\eis\server\bin.

Multiple OLAP Metadata Catalogs can be mapped under the [JDBC DSN] heading. If you want to comment out certain data sources in a list of data sources, insert the pound sign (#) as the first character in a line. (See the data source mapping examples that follow.)

The following syntax provides usage guidelines for editing ais.cfg to connect to a Teradata or an Oracle relational data source.

Syntax

[ JDBC DSN ]<Teradata ODBC DSN>[:<Gateway Host>]:<Port # on Gateway>/
<Server Host>oracle:<tnsname | oname>[:<host name>]:<port #>:<SID>

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[JDBC DSN]</td>
<td>(Required) Available by default</td>
</tr>
</tbody>
</table>

**For Teradata:**

<table>
<thead>
<tr>
<th>Teradata ODBC DSN</th>
<th>The name of the Teradata data source used as the OLAP Metadata Catalog in Integration Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gateway Host</td>
<td>The host on which the Teradata Gateway program is running</td>
</tr>
<tr>
<td>Port # on Gateway</td>
<td>Port number of the Gateway host</td>
</tr>
<tr>
<td>Server Host</td>
<td>The host on which the Teradata server resides</td>
</tr>
</tbody>
</table>

**For Oracle:**

<table>
<thead>
<tr>
<th>tnsname</th>
<th>The name of the Oracle Net Service Name (found in $ORACLE_HOME/network/admin/tnsnames.ora) used as the OLAP Metadata Catalog in Integration Services</th>
</tr>
</thead>
</table>
| oname   | The Schema Name used to connect to an Oracle database if you are using onames instead of tnsnames
          | This is the database identifier that you use when you connect to the database using SQL*Plus.                                     |
| host name | The host on which the Oracle server is installed or running                                                                    |
| port #  | Port number of the host on which the Oracle server is configured                                                              |
| SID     | The server identifier for Oracle on host <host name>                                                                            |

In the following examples for mapping a JDBC data source for Teradata and Oracle, note that the pound sign (#) at the beginning of a line indicates that the line has been commented out.
Example of JDBC Type 3 Settings for Teradata

[JDBC DSN]td_catalog_dsn:labmachine1:6063/teradataServer
tbc_md:labmachine1:6063/teradataServer
6063:teradata

Example of JDBC Type 4 Settings for Oracle

[JDBC DSN]oracle:ora_tbc_md:labmachine2:1521:orasid
oracle:tbc_md:labmachine2:1521:orasid

When using Oracle with tnsnames, if the definitions for tnsname, host name, port #, and SID elements do not match the corresponding elements of a Net Service Name stanza in the tnsnames.ora file, a Java environment error will be the result. The preceding JDBC settings example is built using the elements in the following tnsnames.ora stanza:

ora_tbc_md =
 (DESCRIPTION =
 (ADDRESS_LIST =
 (ADDRESS = (PROTOCOL = TCP)(HOST = labmachine2)(PORT = 1521))
 )
 (CONNECT_DATA =
 (SERVICE_NAME = orasid)
 )
 )

When using Oracle with onames, in the example above, ora_tbc_md is the Schema Name used to connect to an Oracle database. This is the database identifier you use when you are connecting to the database using SQL*Plus.

Mapping JDBC Data Sources for UNIX (Teradata Users Only)

Teradata users must map a JDBC data source to a relational database in the ais.cfg file in order to use the XML Import/Export feature of Integration Services. Teradata requires JDBC Type 3.

To map a data source, edit the ais.cfg file (located in products/Essbase/eis/server/bin) using any text editor. If the ais.cfg file does not exist, use a text editor to create and edit it, and then save it in products/Essbase/eis/server/bin.

Multiple OLAP Metadata Catalogs can be mapped under the [JDBC DSN] heading. If you want to comment out certain data sources in a list of data sources, insert the pound sign (#) as the first character in a line. (See the data source mapping examples that follow.)

The following syntax provides usage examples for editing ais.cfg to connect to a Teradata or an Oracle relational data source.

Syntax

[JDBC DSN]
<Teradata ODBC DSN>:<Gateway Host>/<Port # on Gateway>/<Server Host>
oracle:<tnsname | oname>:<host name>:<port #>:<SID>
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[JDBC DSN]</td>
<td>(Required) Available by default</td>
</tr>
</tbody>
</table>

**For Teradata:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teradata ODBC DSN</td>
<td>The name of the Teradata data source used as the OLAP Metadata Catalog in Integration Services</td>
</tr>
<tr>
<td>Gateway Host</td>
<td>The host on which the Teradata JDBC Type 3 driver is configured and the Jserver is running</td>
</tr>
<tr>
<td>Port # on Gateway</td>
<td>Port number of the Gateway host</td>
</tr>
<tr>
<td>Server Host</td>
<td>The host on which the Teradata server resides</td>
</tr>
</tbody>
</table>

**For Oracle:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tnsname</td>
<td>The name of the Oracle Net Service Name (found in $ORACLE_HOME/network/admin/tnsnames.ora) used as the OLAP Metadata Catalog in Integration Services</td>
</tr>
<tr>
<td>oname</td>
<td>The Schema Name used to connect to an Oracle database if you are using onames instead of tnsnames</td>
</tr>
<tr>
<td>This is the database identifier that you use when you are using SQL *Plus to connect to the database.</td>
<td></td>
</tr>
<tr>
<td>host name</td>
<td>The host on which the Oracle server is installed and running</td>
</tr>
<tr>
<td>port #</td>
<td>Port number of the host on which the Oracle server is configured</td>
</tr>
<tr>
<td>SID</td>
<td>The server identifier for Oracle on host &lt;host name&gt;</td>
</tr>
</tbody>
</table>

In the following examples for mapping a JDBC data source for Teradata and Oracle, note that the pound sign (#) at the beginning of a line indicates that the line has been commented out.

**Example of JDBC Type 3 Settings for Teradata**

```
[JDBC DSN]
td_catalog_dsn:labmachine1:6063/teradataServer
#tbc_md:labmachine1:6063/tera1
td_tbc_md:labmachine1:6063:tera1/
```

**Example of JDBC Type 4 Settings for Oracle**

```
[JDBC DSN]
oracle:ora_tbc_md:labmachine2:1521:orasid
#oracle:tbc_md:labmachine2:1521:orasid
```

When using Oracle with tnsnames, if the definitions for tnsname, host name, port #, and SID elements do not match the corresponding elements of a Net Service Name stanza in the tnsnames.ora file, a Java environment error results. The preceding JDBC settings example is built using the elements in the following tnsnames.ora stanza:

```
ora_tbc_md =
  (DESCRIPTION =
   (ADDRESS_LIST =
     (ADDRESS = (PROTOCOL = TCP)(HOST = labmachine2)(PORT = 1521))
   (CONNECT_DATA =
     (SERVICE_NAME = orasid)
   )
  )
```
When using Oracle with onames, in the example above, `ora_tbc_md` is the Schema Name used to connect to an Oracle database. This is the database identifier that you use when you are using SQL*Plus to connect to a database.

**Manually Configuring the Environment for UNIX Systems**

After you install server products on any UNIX platform, you can manually configure the environment for the user to allow Essbase Integration Server to run and access the relational data source.

*Note:* Running `ais_start` automatically configures your environment.

The steps to configure your environment include configuring the user login scripts to run the `is.sh` (or) `is.csh` file and update the environment variables for Essbase Integration Server.
Creating, Upgrading, and Deleting OLAP Metadata Catalogs

Before you run Integration Services, you must create an OLAP Metadata Catalog to store the OLAP models that you create and the metaoutlines that you build from the OLAP models.

Note: If you intend to use the Sample application to acquaint yourself with Integration Services, skip to Chapter 5, “Setting Up the Sample Applications.” The Sample application has its own relational data source, OLAP Metadata Catalog, OLAP model, and metaoutline. Return to this chapter when you are ready to use a relational data source to build OLAP models and metaoutlines. It is recommended that you use the automatic installation process to install the Sample applications. See “Setting Up the Standard Sample Application Automatically” on page 91.

This chapter tells you what to consider before you create an OLAP Metadata Catalog in a relational database. You can create the catalog either automatically, through the use of Integration Services Console, or manually. You can also manually create a Unicode-enabled OLAP Metadata Catalog. For manual creation of an OLAP Metadata Catalog, this chapter lists the SQL scripts that you must run to create tables for the catalog. The procedures in this chapter assume that you know how to create tables by running SQL scripts.

About OLAP Metadata Catalogs

An OLAP Metadata Catalog is a database that consists of 33 tables in which Integration Services stores OLAP models and metaoutlines.

For a complete list of supported RDBMSs, see Oracle Hyperion Enterprise Performance Management System Certification Matrix (http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html)

You can run the RDBMS for an OLAP Metadata Catalog on any supported platform as long as you have the ODBC driver and database client software required to connect to the catalog from
the computer that runs Essbase Integration Server. You can also run the RDBMS for the relational data source that you use to create OLAP models and build metaoutlines on any supported platform, if you have the ODBC driver and database client software required to access the data source. For a detailed matrix of specifically supported databases and ODBC drivers, see Oracle Hyperion Enterprise Performance Management System Certification Matrix (http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html).

The RDBMS for the OLAP Metadata Catalog can be different from the RDBMS for the relational data source, and the platforms for the two RDBMSs do not need to be the same.

You can have more than one OLAP Metadata Catalog and, by using the XML Import/Export utility, you can move OLAP models and metaoutlines from one OLAP Metadata Catalog to another OLAP Metadata Catalog.

### Creating or Upgrading OLAP Metadata Catalogs

To create an OLAP Metadata Catalog, you must have permission to create tables in the database or have whatever similar access privileges are required by the RDBMS that you are using.

**Note:** Microsoft SQL Server—The tables in a catalog are not accessible if they are created by a user that does not have db_owner privileges.

To create an OLAP Metadata Catalog:

1. **Create a database for OLAP Metadata Catalog tables using the applicable RDBMS utility.**
   
   See “Creating Databases for OLAP Metadata Catalog Tables” on page 74.

2. **Take one of the following actions:**
   
   - Create an OLAP Metadata Catalog automatically by using Integration Services Console (see “Creating or Upgrading OLAP Metadata Catalogs Automatically” on page 75).
   - Create tables for the OLAP Metadata Catalog database by running SQL scripts (see “Creating OLAP Metadata Catalogs Manually” on page 76).

### Creating Databases for OLAP Metadata Catalog Tables

Create a database for the OLAP Metadata Catalog tables by using the applicable RDBMS utility, in the same way that you create any database.

- Create a database device or tablespace
- Allot 30 MB for storage
- Create user names and passwords, if needed
- Grant user privileges or permissions
One OLAP Metadata Catalog can store all OLAP models and metaoutlines from numerous relational data sources. You can also create separate OLAP Metadata Catalogs to store additional OLAP models and metaoutlines for different projects.

**Creating or Upgrading OLAP Metadata Catalogs Automatically**

When you complete installation of Integration Services and start Integration Services Console, the program automatically displays the OLAP Metadata Catalog Setup dialog box, as shown in Figure 2.

This dialog box enables you to create a standard OLAP Metadata Catalog automatically, using Integration Services Console. If you have an older version of the OLAP Metadata Catalog, you also use this dialog box to upgrade the older version to the current version.

![OLAP Metadata Catalog Setup Dialog Box](image)

If you prefer to create the OLAP Metadata Catalog manually, see “Upgrading OLAP Metadata Catalogs Manually” on page 77.

To create an OLAP Metadata Catalog automatically:

1. **From the Server Name drop-down list in the OLAP Metadata Catalog Setup dialog box, type or select the server computer on which you have installed Essbase Integration Server.**

   It is necessary to type a server name the first time it is used. After the OLAP Metadata Catalog is successfully created, the server name is then displayed in the Server Name drop-down list box.
2 From the Catalog ODBC DSN drop-down list, select the Data Source Name for the OLAP Metadata Catalog that you are creating.

Note: You must create the database in which you will store the OLAP Metadata Catalog, assign the appropriate user permissions, and configure the ODBC connection before you can create the catalog.

3 In the Code Page drop-down list box, select the code page that corresponds to the code page of the relational database that will be used as the OLAP Metadata Catalog.

The code page is a portion of the locale which identifies the local language and cultural conventions such as the formatting of currency and dates and the sort order of data.

4 In the User Name drop-down list, select or type the user name to which you have assigned permission to access the database in which the OLAP Metadata Catalog will be stored.

5 In the Password text box, type the password for the user name to which you have assigned permission to access the database in which the OLAP Metadata Catalog will be stored.

6 If you want the OLAP Metadata Catalog Setup dialog box to be displayed automatically each time you start Integration Services Console, leave the Show this dialog at Startup check box selected.

If you do not select the “Show this dialog at Startup” check box, you can access the “OLAP Metadata Catalog Setup” dialog box from the console by selecting Tools, then Create Catalog.

If an OLAP model or metaoutline is open when you select Tools, then Create Catalog, you are prompted to disconnect from the current catalog (Connections, then OLAP Metadata Catalog, then Disconnect). Note that disconnecting from an OLAP Metadata Catalog causes the open OLAP model or metaoutline to close.

7 Click Create.

Integration Services Console creates the OLAP Metadata Catalog automatically.

Note: If you are migrating from a previous release of Integration Services, the OLAP Metadata Catalog auto-creation process upgrades the existing OLAP Metadata Catalog.

Creating OLAP Metadata Catalogs Manually

If you choose to create the OLAP Metadata Catalog manually, you must run SQL scripts by using the same utility program that you normally use to create tables.

The scripts that you use to create the tables for a non-Unicode OLAP Metadata Catalog are named:

`oc_create_database_name.sql`

The scripts that you use to create the tables for a Unicode OLAP Metadata Catalog are named:

`oc_create_database_name_unicode.sql`
These scripts are listed in Table 1 on page 79, along with the utility programs with which they have been tested.

**Note:** strongly recommends that you use the automatic installation process to create the OLAP Metadata Catalog (see “Creating or Upgrading OLAP Metadata Catalogs Automatically” on page 75).

To create tables for the OLAP Metadata Catalog database manually:

1. Start the utility program.
2. Connect to the database that you created for the OLAP Metadata Catalog.
3. Open the appropriate SQL script file in the `ocscript` directory.
4. Run the SQL script to build the tables.
   
   On Microsoft SQL Server, you receive a message that you did not create data or rows. This message is normal because you created only tables and columns.
5. Verify that you have created the OLAP Metadata Catalog tables.
   
   For example, type a command such as:
   
   ```sql
   SELECT * FROM JOIN_HINTS
   ```
   
   or, start the applicable RDBMS utility program and verify that the OLAP Metadata Catalog has the new tables.
6. Close the utility program.

### Upgrading OLAP Metadata Catalogs Manually

If you have OLAP Metadata Catalogs from an earlier release of Integration Services and you choose not to use the automatic creation-upgrade process, you must upgrade the existing OLAP Metadata Catalogs manually. You cannot use the existing catalogs with the new release of Integration Services until the catalogs are upgraded.

**Note:** If you use Integration Services Console to create an OLAP Metadata Catalog automatically, the system upgrades existing OLAP Metadata Catalogs and you do not need to upgrade them manually (see “Creating or Upgrading OLAP Metadata Catalogs Automatically” on page 75). recommends that you use the automatic process to create the OLAP Metadata Catalog.

You manually upgrade the tables of an OLAP Metadata Catalog by running the SQL scripts that use the same database utility program that you typically use to create tables. The SQL scripts to upgrade tables for the OLAP Metadata Catalog are in the `ocscript` directory where you installed Integration Services. The upgrade scripts are named `oc_upgrade*_database_name.sql` and are listed in Table 1 on page 79, along with the utility programs with which they have been tested.
Note: If you manually rebuild an OLAP Metadata Catalog, you must drop (delete) the catalog tables by using `oc_drop*database_name.sql`. Then you create the OLAP Metadata Catalog tables by using `oc_create*database_name.sql`. If you choose to rebuild a catalog manually, it is not necessary to run any upgrade scripts.

Caution! If you drop (delete) an OLAP Metadata Catalog, you also delete the OLAP models and metaoutlines that it contains.

The following topics discuss various aspects of upgrading OLAP Metadata Catalogs manually:

- “SQL Scripts Used to Create and Upgrade Tables” on page 78
- “Proper Order of SQL Scripts” on page 79
- “Upgrading Tables in the OLAP Metadata Catalog” on page 80

SQL Scripts Used to Create and Upgrade Tables

The SQL scripts to create tables for the OLAP Metadata Catalog (listed in Table 1) are located in the `ocscript` directory where you installed Integration Services.

Integration Services provides SQL scripts for each supported RDBMS:

- `oc_create_database_name.sql` to build tables
- `oc_drop_database_name.sql` to drop tables
- `oc_upgrade20_database_name.sql` to upgrade tables from Integration Services Release 1.x to 2.0 (does not apply to Teradata users)
- `oc_upgrade61_database_name.sql` to upgrade tables from Integration Services Release 2.0 to 6.1 (does not apply to Teradata users)
- `oc_upgrade65_database_name.sql` to upgrade tables from Integration Services Release 6.1 to 6.5
- `oc_upgrade651_database_name.sql` to upgrade tables from Integration Services Release 6.5 to 6.5.1

Note: If you need to rebuild an OLAP Metadata Catalog, you must drop (delete) the catalog tables before you build them. Remember that if you drop an OLAP Metadata Catalog, you also delete the OLAP models and metaoutlines that it contains.

The utilities listed in Table 1 have been tested to work with the SQL scripts.
<table>
<thead>
<tr>
<th>Database</th>
<th>SQL Script</th>
<th>Utility Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM DB2</td>
<td>oc_create_db2.sql</td>
<td>IBM DB2 Command Center or &gt;DB2 -tvf</td>
</tr>
<tr>
<td></td>
<td>oc_drop_db2.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade20_db2.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade61_db2.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade65_db2.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade651_db2.sql</td>
<td></td>
</tr>
<tr>
<td>Oracle</td>
<td>oc_create_oracle.sql</td>
<td>SQL*Plus</td>
</tr>
<tr>
<td></td>
<td>oc_drop_oracle.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade20_oracle.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade61_oracle.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade65_oracle.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade651_oracle.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_create_oracle_unicode.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_create_oracle9i_unicode.sql</td>
<td></td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>oc_create_sqlsrv.sql</td>
<td>Query Analyzer (Microsoft SQL Server 7.0 and 2005)</td>
</tr>
<tr>
<td></td>
<td>oc_drop_sqlsrv.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade20_sqlsrv.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade61_sqlsrv.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade65_sqlsrv.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade651_sqlsrv.sql</td>
<td></td>
</tr>
<tr>
<td>Teradata</td>
<td>oc_create_teradata.sql</td>
<td>BTEQ</td>
</tr>
<tr>
<td></td>
<td>oc_drop_teradata.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade65_teradata.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade651_teradata.sql</td>
<td></td>
</tr>
</tbody>
</table>

### Proper Order of SQL Scripts

If you choose to upgrade the OLAP Metadata Catalog manually, you must run the SQL scripts in successive release order beginning with the upgrade script applicable to your current release. This is necessary because most releases make different changes to the catalog, and changes of one release are predicated on the changes of the preceding release. However, some releases of Integration Services (Release 6.2 and Releases 6.5.2, 6.5.3, 6.5.4, 7.0, and 7.1) do not require any
changes to the OLAP Metadata Catalog from the immediately preceding release. No upgrade scripts exist for these releases.

Upgrading from Integration Services Release 1.x

If you are upgrading from Release 1.x to the current release, you run the following upgrade scripts in the order listed:

- `oc_upgrade20_*.sql`
- `oc_upgrade61_*.sql`
- `oc_upgrade65_*.sql`
- `oc_upgrade651_*.sql`

Upgrading from Integration Services Release 2.0

If you are upgrading from Release 2.0 to the current release, you run the following upgrade scripts in the order listed:

- `oc_upgrade61_*.sql`
- `oc_upgrade65_*.sql`
- `oc_upgrade651_*.sql`

Upgrading from Integration Services Release 6.1

If you are upgrading from Release 6.1 to the current release, you run the following scripts in succession:

- `oc_upgrade65_*.sql`
- `oc_upgrade651_*.sql`

Upgrading from Integration Services Release 6.5

If you are upgrading from Release 6.5 to the current release, you run the following script:

- `oc_upgrade651_*.sql`

---

**Caution!** After you update an OLAP Metadata Catalog, you cannot roll back to a previous version. The new version of OLAP Metadata Catalog is not compatible with previous releases of Integration Services. In addition, do not attempt to use the new catalog with previous releases of Integration Services software. Such an attempt can result in corrupted OLAP Metadata Catalog data.

---

**Upgrading Tables in the OLAP Metadata Catalog**

To upgrade an OLAP Metadata Catalog manually, you upgrade the tables of the OLAP Metadata Catalog database.

To upgrade tables for the OLAP Metadata Catalog database:

1. **Start the utility program.**
2. **Connect to the database that you created for the OLAP Metadata Catalog as the user who created the original OLAP Metadata Catalog tables.**
3. **Open the appropriate SQL script file in the ocs script directory.**
See “SQL Scripts Used to Create and Upgrade Tables” on page 78.

4 Run the SQL script (or scripts, if applicable) to upgrade the tables.

See “Proper Order of SQL Scripts” on page 79.

On Microsoft SQL Server, you receive a message that you did not create data or rows. This message is normal because you created only tables and columns.

5 Verify that the new tables have been added to the existing OLAP Metadata Catalog.

You can verify this by entering a command, for example:

```
SELECT * FROM OM_DESCRIPTION
```

or, you can start the applicable RDBMS utility program and verify that the OLAP Metadata Catalog has the new tables.

6 Close the utility program.

**Mapping OLAP Metadata Catalogs**

After you create an OLAP Metadata Catalog manually, you must map the catalog to a supported ODBC driver.

For information about supported ODBC drivers, see [Oracle Hyperion Enterprise Performance Management System Certification Matrix](http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html). For information about configuring OLAP Metadata Catalog as an ODBC data source, see Chapter 2, “Configuring Data Sources”.

**Note:** If you attempt to access a Microsoft SQL Server database with the Microsoft native ODBC driver and you do not have access permission, SQL Server connects you to the default database without notifying you.

**Deleting OLAP Metadata Catalogs**

You can delete an existing OLAP Metadata Catalog at any time.

---

**Caution!** Remember that if you delete an OLAP Metadata Catalog, you also delete the OLAP models and metaoutlines that it contains.

- To delete an OLAP Metadata Catalog:
  1. From either the OLAP Model or OLAP Metaoutline main window, select Tools, then Create Catalog to display the OLAP Metadata Catalog Setup dialog box, as shown in Figure 2 on page 75.
  2. From the Server Name drop-down list, select the appropriate Essbase Integration Server computer.
  3. From the Catalog ODBC DSN drop-down list, select the Data Source Name for the OLAP Metadata Catalog that you want to delete.
4 In the **User Name** drop-down list, select or type your user name.

5 In the **Password** text box, type your password.

6 **Click Yes to the confirmation prompt to delete the catalog.**

   If the OLAP Metadata Catalog that you are attempting to delete is currently in use, you are prompted to disconnect from the current catalog (**Connections**, then **OLAP Metadata Catalog**, then **Disconnect**). You must disconnect before you can delete the catalog.

**Note:** Disconnecting from an OLAP Metadata Catalog causes the open OLAP model or metaoutline to close.
OLAP Metadata Catalog is a relational database that contains OLAP models, metaoutlines, and the information necessary to retrieve the required data from the data source. You can install OLAP Metadata Catalog in any supported RDBMS. See the Oracle Hyperion Enterprise Performance Management System Certification Matrix (http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html) for a complete list of supported databases.

This chapter describes the tables in OLAP Metadata Catalog.

**OLAP Metadata Catalog**

OLAP Metadata Catalog contains a collection of related tables. The tables provide a place for you to store OLAP models and metaoutlines. When you load members or data into an Essbase database, Integration Services retrieves the information stored in OLAP Metadata Catalog to determine:

- What data to retrieve from the data source
- What operations to perform on the data
- How to load the data into the Essbase database

Regularly back up the database that contains OLAP Metadata Catalog so that you can restore OLAP Metadata Catalog from backups if it becomes corrupted. Use one of the procedures in Table 2, depending on what OLAP Metadata Catalog component you want to restore.

**Caution!** It is strongly recommended that you do not alter the table names or data in OLAP Metadata Catalog with tools other than Integration Services.
### Tables Relating to the OLAP Model

Table 3 describes each table in OLAP Metadata Catalog that stores information about OLAP models. For information about the other tables in OLAP Metadata Catalog, see:

- “Tables Relating to the Metaoutline” on page 85
- “Tables Relating to Drill-Through” on page 86
- “Tables Relating to Hybrid Analysis” on page 87
- “Miscellaneous Tables” on page 87

#### Table 3  OLAP Model Tables

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB_FILTER</td>
<td>Contains information about filters in the OLAP model. This information includes the hierarchy filter and system-generated ID numbers for parts of the OLAP model.</td>
</tr>
<tr>
<td>MB_TX_RULE</td>
<td>Contains information about hierarchies in the OLAP model. This information includes transformations to perform on the hierarchy and system-generated ID numbers for parts of the OLAP model.</td>
</tr>
<tr>
<td>OA_INFO</td>
<td>Contains information about columns in a dimension table or a fact table. This information includes the column name, the column data type, the OLAP model it belongs to, the dimension or fact table it belongs to, the source table and column on which it is based, whether it is hidden in OLAP models and metaoutlines, whether it is used for Drill-Through, whether it is enabled for attributes, and how to create it if it is a user-defined column.</td>
</tr>
<tr>
<td>OD_DETAILS</td>
<td>Contains information about dimensions in the OLAP model. This information includes system-generated ID numbers for parts of the OLAP model.</td>
</tr>
<tr>
<td>OD_INFO</td>
<td>Contains information about dimensions in the OLAP model. This information includes the dimension name, the OLAP model the dimension belongs to, and whether the dimension is a measures dimension.</td>
</tr>
<tr>
<td>OM_INFO</td>
<td>Contains high-level information about an OLAP model: the name of the OLAP model, the description, the name of the data source on which the OLAP model is based, the owner, and access privileges.</td>
</tr>
<tr>
<td>OM_PROPERTIES</td>
<td>Contains property information for the OLAP model.</td>
</tr>
<tr>
<td>OMB_DETAILS</td>
<td>Contains information about hierarchies in the OLAP model. This information includes the position of each column in the hierarchy, the order in which to build the hierarchy, and system-generated ID numbers for parts of the OLAP model.</td>
</tr>
<tr>
<td>OMB_INFO</td>
<td>Contains information about hierarchies in the OLAP model. This information includes the hierarchy name, the OLAP model that it belongs to, and the dimension to which it belongs.</td>
</tr>
<tr>
<td>Table Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>OV_INFO</td>
<td>Contains information about the dimension tables and the fact table in an OLAP model. This information includes the dimension or fact table name, the OLAP model it belongs to, its position in the right frame of the OLAP Model main window, and whether it is a dimension table or a fact table.</td>
</tr>
<tr>
<td>OVL_REL_DETAILS</td>
<td>Contains information about logical joins between dimension tables and fact tables. This information includes the name of the first column and the name of the column to which the first column is joined.</td>
</tr>
<tr>
<td>OVL_REL_INFO</td>
<td>Contains information about logical joins between dimension tables and fact tables. This information includes the name of the first dimension or fact table and the name of the dimension or fact table to which the first dimension is joined.</td>
</tr>
<tr>
<td>OVP_REL_DETAILS</td>
<td>Contains information about physical joins in the source tables. This information includes the name of the first source column and the name of the source column to which the first source column is joined.</td>
</tr>
<tr>
<td>OVP_REL_INFO</td>
<td>Contains information about physical joins in the source tables. This information includes the name of the first source table and the name of the source table to which the first source table is joined.</td>
</tr>
<tr>
<td>OM_DESCRIPTIONS</td>
<td>Contains descriptions for a model, its dimensions, its members, and its hierarchies. This information includes the system-generated ID number for each model, dimension, member, or hierarchy that is described.</td>
</tr>
<tr>
<td>JOIN_HINTS</td>
<td>Contains information about the database hints defined for a join. This information includes the type of join, the physical table to join, and the text of the join hint.</td>
</tr>
</tbody>
</table>

**Tables Relating to the Metaoutline**

Table 4 describes each table in OLAP Metadata Catalog that stores information about metaoutlines. For information about the other tables in OLAP Metadata Catalog, see:

- “Tables Relating to the OLAP Model” on page 84
- “Tables Relating to Drill-Through” on page 86
- “Tables Relating to Hybrid Analysis” on page 87
- “Miscellaneous Tables” on page 87

Table 4  Metaoutline Tables

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MO_FILTERS</td>
<td>Contains information about metaoutline filters. This information includes the filter name and the actions the filter will perform.</td>
</tr>
<tr>
<td>MO_INFO</td>
<td>Contains high-level information about a metaoutline: the name of the metaoutline, its description, the date it was last changed, the OLAP model on which the metaoutline is based, its owner, its access privileges, and whether the spreadsheet user of Drill-Through reports must enter the Integration Server name and a user name and password for the external data source.</td>
</tr>
</tbody>
</table>
### Table Name | Description
--- | ---
**MO_PROPERTIES** | Contains information about metaoutline properties. This information includes the name and value of metaoutline properties, the name of the source table and column that Integration Services uses to assign the properties, the numeric range that represents a range specified by the user, Global Settings properties, and sort order. Sort order is based on a column in the data source, including columns not present in the OLAP model or metaoutline. Sort order can be ascending or descending. Also contains information on association of attributes (by level or by name) to base dimensions.

**MOD_ATTRIBUTES** | Contains information about attribute dimensions, such as attribute dimension type, alias names, and transformation rules.

**MOD_INFO** | Contains information about dimensions in a metaoutline. This information includes the dimension name, the OLAP dimension with which it is associated (unless it is a user-defined dimension), the type of dimension (a user-defined dimension, a measures dimension, a standard dimension, or an attribute dimension), and how members are positioned within the dimension.

**MS_INFO** | Contains information about members in a metaoutline. This information includes the name of the member, the dimension to which it belongs, whether it is a user-defined member or a standard member, the OLAP table with which it is associated, the OLAP model column with which it is associated, the name of its parent, how it is positioned with its siblings, and the database measure with which it is associated (if it is a user-defined member).

**MS_PREDICATES** | Contains information about members in a metaoutline. This information includes member filters.

**MS_TX_RULE** | Contains information about members in a metaoutline. This information includes member transformation rules.

**MSR_INFO** | Contains information about database measures in a metaoutline. This information includes the name of the measure, the OLAP model table with which it is associated, the OLAP model column with which it is associated, and how it is positioned with its siblings.

**MSR_PREDICATES** | Contains information about database measures in a metaoutline. This information includes measure filters.

**MSR_TX_RULE** | Contains information about database measures in a metaoutline. This information includes the name and definition of measure transformation rules.

**MO_DESCRIPTIONS** | Contains descriptions for a metaoutline, its filters, its dimensions, and its members. This information includes the system-generated ID number for each metaoutline, filter, dimension, or member that is described.

---

### Tables Relating to Drill-Through

Drill-through reports are created in the OLAP Metaoutline main window of the Integration Services Console (refer to online help for information on creating drill-through reports). These reports enable Essbase Spreadsheet Add-in users to view data in the data source database that is not stored in the Essbase database.

Table 5 describes each table in OLAP Metadata Catalog that stores information about drill-through reports. For information about the other tables in OLAP Metadata Catalog, see:

- “Tables Relating to the OLAP Model” on page 84
- “Tables Relating to the Metaoutline” on page 85
- “Tables Relating to Hybrid Analysis” on page 87
“Miscellaneous Tables” on page 87

Table 5  Drill-Through Tables

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTR_ATTRIBUTES</td>
<td>Contains information about drill-through columns. This information includes the order in which Essbase Spreadsheet Add-in sorts and displays drill-through information.</td>
</tr>
<tr>
<td>DTR_INFO</td>
<td>Contains information about drill-through columns. This information includes the name of the drill-through column, whether the spreadsheet wizard is displayed, and the time or row query governor specified.</td>
</tr>
<tr>
<td>MO_INTERSECTION</td>
<td>Contains information about drill-through columns. This information includes system-generated ID numbers for the OLAP model, the drill-through column, and the intersection level on which the column is defined.</td>
</tr>
<tr>
<td>MO_PROPERTIES</td>
<td>Contains information about drill-through filters. This information includes the Essbase user name and password, drill-through report name, and metaoutline name.</td>
</tr>
</tbody>
</table>

Tables Relating to Hybrid Analysis

You enable Hybrid Analysis in the OLAP Metaoutline main window of the Integration Services Console (refer to online help for information on enabling Hybrid Analysis). Hybrid Analysis enables you to access lower-level data stored in a data source database. This stored data, although not physically loaded into Essbase, is mapped to the appropriate Essbase hierarchies and is available to Oracle Essbase Spreadsheet Add-in users. Unlike drill-through data, which is displayed on a separate worksheet, Hybrid Analysis data is displayed seamlessly on the current worksheet in Oracle Essbase Spreadsheet Add-in reports.

Table 6 describes the table in OLAP Metadata Catalog that stores information about Hybrid Analysis. For information about the other tables in OLAP Metadata Catalog, see:

- “Tables Relating to the OLAP Model” on page 84
- “Tables Relating to the Metaoutline” on page 85
- “Tables Relating to Drill-Through” on page 86
- “Miscellaneous Tables” on page 87

Table 6  Hybrid Analysis Tables

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MO_PROPERTIES</td>
<td>Contains information about Hybrid Analysis columns. This information includes system-generated ID numbers for the OLAP model; the Hybrid Analysis column; and column precision, scale, and nullability.</td>
</tr>
</tbody>
</table>

Miscellaneous Tables

Table 7 describes tables in OLAP Metadata Catalog that store miscellaneous information about OLAP models and metaoutlines and about Integration Services. For information about the other tables in OLAP Metadata Catalog, see:

- “Tables Relating to the OLAP Model” on page 84
<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CB_CONCURRENCY</td>
<td>Contains information about users accessing an OLAP model or metaoutline. This information includes the user name, the user session ID, the OLAP model or metaoutline that the user is accessing, whether the user is accessing an OLAP model or a metaoutline, whether the user has a read or read/write lock, and the date and time the user acquired the lock.</td>
</tr>
<tr>
<td>CB_VERSION</td>
<td>Contains information about the Integration Services software you have installed. This information includes the release number of Integration Services.</td>
</tr>
</tbody>
</table>
This chapter tells you what to consider before and after you set up the standard sample application, whether automatically or manually. For manual setup, the topics in this chapter detail which scripts and batch files you must run to create and load the TBC database and the TBC_MD OLAP Metadata Catalog.

The procedures in this chapter assume that you know how to create a database using a relational database management system (RDBMS). For manual setup, the procedures assume that you know how to create tables and load them with data running SQL scripts, batch files, and shell scripts. For information on these topics, see the documentation for the RDBMS that you are using.

**Note:** Sample application creation, either automatic or manual, is not supported on 64-bit UNIX platforms or on the Windows 2008 64-bit platform.

### Overview of Sample Applications

Integration Services provides sample applications that consist of the following features:

- A database for a fictitious company called The Beverage Company (TBC)
- A sample OLAP model and metaoutline created from the TBC database
- A Hybrid Analysis-enabled OLAP model and metaoutline
- A special OLAP Metadata Catalog to store the sample OLAP models and metaoutlines and any additional OLAP models and metaoutlines that you create using the sample data
Preparing the sample application consists of three tasks:

- Setting up the standard sample application
- **Optional:** Setting up the Hybrid Analysis sample application
- **Optional:** Setting up the Unicode sample application

You can set up the standard sample application automatically using Integration Services Console (see “Setting Up the Standard Sample Application Automatically” on page 91), or you can set up the standard sample application manually (“Setting Up the Standard Sample Application Manually” on page 93. Using either method, setting up the standard sample application involves two main tasks:

- You create two relational databases: one for the data in the TBC sample application and one for the OLAP Metadata Catalog database (TBC_MD).

  **Note:** Creating the relational databases is a manual task that you must perform prior to setting up the standard sample application, regardless of whether you choose the automatic or manual method.

- You then create tables in each of these databases and load data into them, using scripts provided with Integration Services. You can create tables either automatically, through the use of Integration Services Console, or manually.

When you finish setting up the standard sample application, the OLAP Metadata Catalog database (TBC_MD) will contain an OLAP model (TBC Model) and a metaoutline (TBC Metaoutline) based on data in the sample TBC relational data source.

You create the Hybrid Analysis sample application manually by completing the tasks in “Setting Up the Hybrid Analysis Sample Application” on page 101. Upon completion of these tasks, the sample TBC relational data source will contain Hybrid Analysis data. The OLAP Metadata Catalog will contain a Hybrid Analysis-enabled OLAP model (HA TBC Model) and metaoutline (HA TBC Metaoutline).

You create the Unicode sample application manually by completing the tasks in “Setting Up the Unicode Sample Application” on page 110. Upon completion of these tasks, the sample TBC_U relational data source will contain Unicode enabled data. The Unicode TBC_U OLAP Metadata Catalog will contain Unicode-enabled OLAP model (TBC_Model_Unicode) and metaoutline (TBC_Metaoutline_Unicode).

**Caution!** If you have installed the sample application from a previous release of Integration Services, you should back up and preserve your existing sample database, OLAP Metadata Catalog, and the OLAP models and metaoutlines that OLAP Metadata Catalog contains. You can then upgrade your existing catalog to be compatible with the current release of the software (see “Upgrading OLAP Metadata Catalogs Manually” on page 77). You cannot, however, store newly created OLAP models and metaoutlines in your previous catalog.
Preparing to Set Up the Standard Sample Application

Before you set up the standard sample application, you must install the database client software for a supported RDBMS. When setting up the standard sample application, you must use a computer on which the server component of Integration Services (Essbase Integration Server) and the necessary database client software are both installed. For more information, see “Setting Up the Standard Sample Application Automatically” on page 91.

For manual setup of the standard sample application, the standard sample application scripts do not have to be installed on the same computer that you are using, but you must be able to access the script files.

Note: strongly recommends that you use the automatic installation process to install the standard sample applications (see “Setting Up the Standard Sample Application Automatically” on page 91).

If you choose to set up the standard sample application manually, refer to Table 8 and also verify that the appropriate client utility program is installed.

Table 8 Required Utilities for Setting Up the Standard Sample Application

<table>
<thead>
<tr>
<th>Database</th>
<th>Utility Programs</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM DB2 UDB</td>
<td>IBM DB2 Command Window or &gt;DB2 -tvf</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>Query Analyzer</td>
</tr>
<tr>
<td>Oracle</td>
<td>SQL*Plus</td>
</tr>
<tr>
<td>Teradata</td>
<td>BTEQ</td>
</tr>
</tbody>
</table>

The remainder of this chapter assumes that you have installed the required database client software.

Setting Up the Standard Sample Application Automatically

After you have completed the Integration Services installation process, you can choose to install either or both the OLAP Metadata Catalog and the standard sample application. The system first displays the OLAP Metadata Catalog Setup dialog box to enable automatic setup of the catalog (see “Creating or Upgrading OLAP Metadata Catalogs Automatically” on page 75).

Whether or not you choose to create the OLAP Metadata Catalog automatically, you can choose to create the standard sample application automatically, after you finish creating the OLAP Metadata Catalog. You can create the standard sample application quickly and easily using the automatic feature. You are encouraged to use this convenient feature, but if you choose manual setup, see “Setting Up the Standard Sample Application Manually” on page 93 for detailed procedures.
Hybrid Analysis: If you plan to set up the sample application for Hybrid Analysis, you must follow the procedures in “Setting Up the Hybrid Analysis Sample Application” on page 101.

Note: ODBC connections must be configured before you begin to set up the standard sample application. If you have not yet configured ODBC connections, see Chapter 2, “Configuring Data Sources” for information on setting up ODBC data sources, including an OLAP Metadata Catalog and relational data sources.

To set up the standard sample application automatically:

1. Select Tools, then Create Sample to access the Sample Application Setup dialog box shown in Figure 3:

   ![Sample Application Setup Dialog Box](image)

   **Figure 3** Sample Application Setup Dialog Box

   **Sample Application Setup**
   
   Use this dialog box to automatically create the sample TBC (The Beverage Company) application, including the sample data, sample OLAP model, and sample metaoutline.

   **Sample ODBC DSN**
   
   TBC

   **User Name**
   
   TBC

   **Password**
   
   

2. In the **Sample ODBC DSN** text box, select or type the data source name for creating or dropping (deleting) tables in the source database.

   The User Name text box displays the name, TBC. You must use this name to create the TBC standard sample application database.

3. In the **Password** text box, type the password for the user to which you have assigned privileges to access the TBC database for the standard sample application.

4. Click **Create** to create the standard sample application.

   Integration Services creates the necessary tables, loads the TBC sample data into the tables, and loads the sample OLAP model and metaoutline data into the OLAP Metadata Catalog.

   If Integration Services detects that you are not currently connected to the OLAP Metadata Catalog in which to load the sample OLAP model and metaoutline data, the Login dialog box is displayed so that you can enter the information necessary to connect.

   After you close the Login dialog box and are returned to the Sample Application Setup dialog box, click **Create** again to initiate setup of the standard sample application.
If, after clicking Create, Integration Services detects either an existing sample data source or an existing sample OLAP model and metaoutline in the OLAP Metadata Catalog, you are prompted with the message that tables already exist in the database. Click OK to return to the Sample Application Setup dialog box to reenter information, or click Delete to delete the existing standard sample application data.

If you click Delete, your existing OLAP Metadata Catalog sample OLAP models and metaoutlines will be deleted and replaced with the sample OLAP model and metaoutline for the current release.

After you have completed creation of the Integration Services standard sample application, you are ready to begin working with Integration Services Console to create your own sample OLAP models and metaoutlines.

**Setting Up the Standard Sample Application Manually**

To set up the standard sample application manually, you begin by creating two databases in a supported RDBMS: TBC (the sample database) and TBC_MD (an OLAP Metadata Catalog). You then create tables in each of these databases and load data into them, using scripts provided with Integration Services. The names and directory locations of these scripts are detailed in the topics that follow.

*Note:* strongly recommends that you use the automatic installation process to install the standard sample applications (see “Setting Up the Standard Sample Application Automatically” on page 91).

After you create the TBC and TBC_MD databases in a supported RDBMS, you use several scripts and files to create the standard sample application:

- A script for creating the TBC sample database tables
  
  This script is located in the `samples\tbcdbase` directory.

- A script for loading the TBC sample database data
  
  This script is located in the `samples\tbcdbase` directory.

- A script for creating the TBC_MD OLAP Metadata Catalog database tables
  
  This script is located in the `ocscript` directory.

- XML files for importing a sample OLAP model and metaoutline in the TBC_MD OLAP Metadata Catalog database
  
  These XML files are located in the `samples\tbcmodel` directory.

The scripts for creating the standard sample application and OLAP Metadata Catalog are installed with the server software during Integration Services installation. Different versions of the scripts are provided for each of the supported RDBMSs.
Note: Some of the standard sample application scripts require slightly different procedures, depending on the RDBMS that you are using.

**Setting Up the TBC Relational Data Source**

The relational data source for the standard sample application is TBC.

Note: To create a database, you must have database administrator or similar access privileges required by the RDBMS that you are using.

To set up TBC relational data source:

1. **Create the TBC database using an RDBMS.**
   For more information, see “Creating the TBC Database” on page 94.

2. **Create tables for the TBC database by running SQL scripts.**
   For instructions, see “Creating Tables for the TBC Database” on page 94.

3. **Load data into the TBC database by running SQL scripts.**
   For instructions, see “Loading Data into the TBC Tables” on page 95.

---

**Creating the TBC Database**

Create the TBC database in the same way that you create any database using an RDBMS:

- Create a database device or tablespace named TBC.
- Allot 20 MB for storage.
- Create a user TBC who can drop and create tables.
- Grant user privileges or permissions to create and drop tables.

---

**Caution!** The user TBC must create the tables for the TBC database, or portions of the standard sample application will not work.

---

**Creating Tables for the TBC Database**

Create tables for the TBC database with the same utility program that you typically use to create tables by running SQL scripts.

The standard sample application SQL scripts needed to create tables for the TBC database are in the `samples\tbcdbase` directory where you installed Integration Server.

The utilities listed in Table 9 have been tested to work with the SQL scripts:
Table 9  Tested Utilities for Creating TBC Tables

<table>
<thead>
<tr>
<th>Database</th>
<th>SQL Script</th>
<th>Utility Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM DB2</td>
<td>tbc_create_db2.sql</td>
<td>IBM DB2 Command Window or &gt;DB2 -tvf</td>
</tr>
<tr>
<td></td>
<td>tbc_drop_db2.sql</td>
<td></td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>tbc_create_sqlsrv.sql</td>
<td>Query Analyzer</td>
</tr>
<tr>
<td></td>
<td>tbc_drop_sqlsrv.sql</td>
<td></td>
</tr>
<tr>
<td>Oracle</td>
<td>tbc_create_oracle.sql</td>
<td>SQL*Plus</td>
</tr>
<tr>
<td></td>
<td>tbc_drop_oracle.sql</td>
<td>SQL*Loader command line utility</td>
</tr>
<tr>
<td>Teradata</td>
<td>tbc_create_teradata.sql</td>
<td>BTEQ</td>
</tr>
<tr>
<td></td>
<td>tbc_drop_teradata.sql</td>
<td></td>
</tr>
</tbody>
</table>

Each RDBMS has two scripts—one to build tables and another to drop tables.

Caution! You must create the tables for the TBC database as user TBC, or portions of the standard sample application will not work.

To create tables for the TBC database:

1. Start the utility program.
2. Connect to the TBC database as user TBC.

   Note: Use uppercase TBC for Microsoft SQL Server.
3. In the samples\tbcdbase directory, open the appropriate SQL script file.
4. Run the SQL script.
   Microsoft SQL Server: You should receive a message that you did not create data or rows. This message is normal because you created only tables and columns.
5. Verify that you have created the TBC tables; for example, type `SELECT * FROM PRODUCTDIM`
   or start the RDBMS and verify that the TBC database has the new tables.
6. Close the utility program.

Loading Data into the TBC Tables

Load data into the TBC tables by running an SQL script using the same utility program that you typically use to load tables by running SQL scripts.

The utilities listed in Table 10 have been tested to work with SQL scripts.
To load data into the TBC tables:

1. From the command line, move to the `samples\tbcdbase` directory where you installed Integration Server.
2. Start the utility program.
3. Connect to the TBC database as user TBC.
4. In the `sample\tbcdbase` directory, open the `sampledata.sql` script file.
5. Run the SQL script using your RDBMS tool.
6. Verify that you have loaded data into the TBC tables; for example, type `SELECT * FROM TBC.PRODUCT` or start the RDBMS and execute a query.
7. Close the utility program.

### Setting Up the TBC_MD OLAP Metadata Catalog

The OLAP Metadata Catalog for the standard sample application is TBC_MD. For more information on OLAP Metadata Catalogs, see Chapter 3, “Creating, Upgrading, and Deleting OLAP Metadata Catalogs.”

To set up the TBC_MD OLAP Metadata Catalog:

1. Create a TBC_MD database using an RDBMS.
   For more information, see “Creating the TBC_MD Database” on page 97.
2. Create tables for the TBC_MD database by running SQL scripts.
   For instructions, see “Creating Tables for the TBC_MD Database” on page 97.
3. Load data into the TBC_MD database using the XML import utility.
   For instructions, see “Using the XML Import Utility to Load Data into the TBC_MD Tables” on page 99.
Note: If you have a previous release of Integration Services and are upgrading your OLAP Metadata Catalog, you cannot roll back to the previous version. The new version of the OLAP Metadata Catalog is not compatible with earlier releases of Integration Services.

Creating the TBC_MD Database

Create the TBC_MD database in the same way that you create any database using an RDBMS:

- Create a database device or tablespace named TBC_MD.
- Allot 20 MB for storage.
- Create a user TBC who can drop and create tables.
- Grant user privileges or permissions.

Caution! You must create the tables for the TBC_MD database as user TBC or portions of the standard sample application will not work.

Creating Tables for the TBC_MD Database

Create tables for the TBC_MD database with the same utility program that you typically use. The standard sample application SQL scripts used to create tables for the TBC_MD database are in the ocscript directory where you installed Integration Server. The SQL scripts in the ocscript directory are the same scripts that you use to create any OLAP Metadata Catalog. For information on OLAP Metadata Catalogs, see Chapter 3, “Creating, Upgrading, and Deleting OLAP Metadata Catalogs.”

The utilities listed in Table 11 have been tested to work with the SQL scripts:

<table>
<thead>
<tr>
<th>Database</th>
<th>SQL Script</th>
<th>Utility Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM DB2</td>
<td>oc_create_db2.sql</td>
<td>IBM DB2 Command Center or &gt;DB2 -tvf</td>
</tr>
<tr>
<td></td>
<td>oc_drop_db2.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade20_db2.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade61_db2.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade65_db2.sql</td>
<td></td>
</tr>
<tr>
<td></td>
<td>oc_upgrade651_db2.sql</td>
<td></td>
</tr>
</tbody>
</table>

Setting Up the TBC_MD OLAP Metadata Catalog 97
<table>
<thead>
<tr>
<th>Database</th>
<th>SQL Script</th>
<th>Utility Program</th>
</tr>
</thead>
</table>
| Microsoft SQL Server | oc_create_sqlsrv.sql  
|                  | oc_drop_sqlsrv.sql  
|                  | oc_upgrade20_sqlsrv.sql  
|                  | oc_upgrade61_sqlsrv.sql  
|                  | oc_upgrade65_sqlsrv.sql  
|                  | oc_upgrade651_sqlsrv.sql  | Query Analyzer |
| Oracle         | oc_create_oracle.sql  
|                  | oc_drop_oracle.sql  
|                  | oc_upgrade20_oracle.sql  
|                  | oc_upgrade61_oracle.sql  
|                  | oc_upgrade65_oracle.sql  
|                  | oc_upgrade651_oracle.sql  
|                  | oc_create_oracle_unicode.sql  
|                  | oc_create_oracle9i_unicode.sql  | SQL*Plus |
| Teradata       | oc_create_teradata.sql  
|                  | oc_drop_teradata.sql  
|                  | oc_upgrade65_teradata.sql  
|                  | oc_upgrade651_teradata.sql  | BTEQ |

**Note:** The scripts used to upgrade OLAP Metadata Catalog manually from Release 2.0 to Release 6.2 are identical to those used to upgrade from Release 2.0 to Release 6.1. Therefore, the names have not been changed from the names used in Integration Services Release 6.1. If you are upgrading OLAP Metadata Catalog from Release 2.0 to Release 6.2, use the scripts named *61*.sql that are appropriate for your RDBMS.

Integration Services provides five SQL scripts for each RDBMS (with the exception of Teradata):

- **oc_create_database_name.sql** to build tables
- **oc_drop_database_name.sql** to drop tables
- **oc_upgrade20_database_name.sql** to upgrade tables from Integration Services Release 1.x to 2.0 (does not apply to Teradata users)
- **oc_upgrade61_database_name.sql** to upgrade tables from Integration Services Release 2.0 to 6.2 (does not apply to Teradata users)
- **oc_upgrade65_database_name.sql** to upgrade tables from Integration Services Release 6.1 or 6.2 to 6.5
• `oc_upgrade651_database_name.sql` to upgrade tables from Integration Services Release 6.5 to 6.5.1

If you need to rebuild tables, you must first drop the tables before you build them again.

**Caution!** You must create the tables for the TBC_MD database as user TBC or portions of the standard sample application will not work.

To create tables for the TBC_MD database:

1. Start the utility program.
2. Connect to the TBC_MD database as user TBC.
3. In the `ocscript` directory, open the appropriate SQL script file.
4. Run the SQL script.
   - **Microsoft SQL Server:** You receive a message that you did not create data or rows. This message is normal because you created only tables and columns.
5. Verify that you have created the TBC_MD tables; for example, type the following command: `SELECT * FROM TBC.MS_INFO`
   - or start the RDBMS and verify that the TBC_MD database has the new tables.
6. Close the utility program.

**Using the XML Import Utility to Load Data into the TBC_MD Tables**

Use the XML Import utility to load OLAP model and metaoutline data into the TBC_MD tables. The XML files listed in Table 12 have been tested to work with their respective RDBMS. These files are located in the `samples\tbcmodel` directory.

<table>
<thead>
<tr>
<th>Database</th>
<th>XML File in sample\tbcmodel</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM DB2</td>
<td>sample_model.xml</td>
</tr>
<tr>
<td>Oracle</td>
<td>sample_metaoutline.xml</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td></td>
</tr>
<tr>
<td>Teradata</td>
<td></td>
</tr>
</tbody>
</table>

**Using XML Import to Load Metadata into TBC_MD**

To complete the procedure below, you must start Integration Services Console and connect to Integration Server.
To start Integration Services Console and connect to Integration Server:

1 On the Windows desktop, click Start, and select All Programs, then Oracle EPM System, then Essbase, then Integration Services, then Server.

2 On the Windows desktop, click Start, and select All Programs, then Oracle EPM System, then Essbase, then Integration Services, then Console.

3 If necessary, click Close to clear the OLAP Metadata Catalog Setup dialog box.

Integration Services Console automatically displays the Login dialog box.

a. In the Server text box, under Integration Services, select or type the name of an Integration Server computer; for example, cypress.

   If the port number on which Essbase Integration Server communicates with the console has been set to a value other than the default, you must type the server name or IP address and type the non-standard port number, separated by a colon; for example:

   cypress:3390

b. In the OLAP Metadata Catalog text box, select or type the name of the OLAP Metadata Catalog, TBC_MD, for the sample TBC (The Beverage Company) database.

c. In the User Name and Password text boxes, type the user name and password for the standard sample application user, TBC, and click OK.

To load sample OLAP model metadata into the TBC_MD tables using XML Import:

1 In Integration Services Console, select File, then XML Import/Export.

2 Select the Import tab.

3 In the XML File Path text box, enter the file path for the sample OLAP model or click the Open XML File button and navigate to the following folder:

   EPM_ORACLE_HOME\products\Essbase\eis\server\samples\tbcmodel

4 Select the XML file to import for the sample OLAP model, sample_model.xml.

5 Click OK.

Integration Services Console displays the XML Import/Export dialog box with the XML file path and name in the XML File Path text box. The name of the sample OLAP model is displayed in the OLAP Model Name text box.

6 Click the Import to Catalog button to load the selected XML file into the sample OLAP Metadata Catalog.

To load sample OLAP metaoutline metadata into the TBC_MD tables using XML Import:

1 In Integration Services Console, select File > XML Import/Export.

2 Select the Import tab.

3 In the XML File Path text box, enter the file path for the sample OLAP metaoutline or click the Open XML File button and navigate to the following file:

   EPM_ORACLE_HOME\products\Essbase\eis\server\samples\tbcmodel

4 Select the XML file to import for the sample OLAP metaoutline, sample_metaoutline.xml.
5 Click OK.

Integration Services Console displays the XML Import/Export dialog box with the XML file path and name in the XML File Path text box, the name of the sample OLAP metaoutline in the OLAP Metaoutline Name text box, and the name of the OLAP model upon which this metaoutline is based in the Based on OLAP Model text box.

6 Click the **Import to Catalog** button to load the selected XML file into the sample OLAP Metadata Catalog.

**Setting Up the Hybrid Analysis Sample Application**

Before setting up the Hybrid Analysis sample application, you must set up the standard sample application (see “Setting Up the Standard Sample Application Automatically” on page 91 or “Setting Up the Standard Sample Application Manually” on page 93).

Setting up the Hybrid Analysis sample application is a manual process. Follow the instructions given in this section for your particular platform.

*Note:* The Hybrid Analysis application scripts require slightly different procedures, depending on the RDBMS that you are using.

The utilities listed in Table 9 are required for running the various batch files and SQL scripts used in creating the Hybrid Analysis sample application.

<table>
<thead>
<tr>
<th>Database</th>
<th>Required Utilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM DB2</td>
<td>DB2CMD command line interface</td>
</tr>
<tr>
<td>Oracle</td>
<td>• SQL*Plus</td>
</tr>
<tr>
<td></td>
<td>• SQL*Loader command line utility</td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>• ISQL</td>
</tr>
<tr>
<td></td>
<td>• BCP command line utility</td>
</tr>
<tr>
<td>Teradata</td>
<td>• BTEQ</td>
</tr>
<tr>
<td></td>
<td>• FastLoad</td>
</tr>
</tbody>
</table>

**Setting Up the Hybrid Analysis Sample Application on IBM DB2**

You use several files to create the Hybrid Analysis sample application on IBM DB2:

- A batch file to send commands to the DB2CMD command line interface:
  
  `install_db2.bat`

- A data load script for the TBC sample database:
An XML file that you use to import a sample OLAP model into the TBC_MD OLAP Metadata Catalog database:
HA_TBC_Model.xml

An XML file that you use to import a sample metaoutline into the TBC_MD OLAP Metadata Catalog database:
HA_TBC_Metaoutline.xml

A text file containing sample data:
ha_sampledata.txt

Setting up the Hybrid Analysis sample application on IBM DB2 consists of performing the following tasks:

- Running the batch file install_db2.bat
- Importing the sample OLAP model and metaoutline

**Note:** Before setting up the Hybrid Analysis Sample Application on IBM DB2, verify that the DB2CMD command line interface is installed.

**Running the Batch File on IBM DB2**

The first task in setting up the Hybrid Analysis sample application on IBM DB2 consists of running the install_db2.bat file. This batch file creates tables in the sample database, loads metadata into the dimension tables, and loads sample data into the fact table.

Run the install_db2.bat file located in:
EPM_ORACLE_HOME\products\Essbase\eis\server\Samples\tbcdbase

This batch file sends SQL commands to the DB2CMD command line interface.

Pass the following three parameters to install_db2.bat:

- The user ID that you use to connect with IBM DB2
- Your password
- The IBM DB2 database name

The install_db2.bat file runs the DB2CMD command line interface, which executes the ha_create_db2.sql load script file. This script file performs the following tasks:

- Deletes any existing tables in the sample TBC database
- Creates a new set of tables in the TBC database
- Inserts records into the dimension tables
- Creates the ha_results_createtables.txt file
- Loads the sample data into the fact table
The `ha_results_createtables.txt` file contains the results of the table-creating and loading operations.

**Note:** After the member and data load is finished, close the IBM DB2 command window.

### Importing the Sample OLAP Model and Metaoutline

The final task in setting up the Hybrid Analysis sample application on IBM DB2 consists of importing the two XML files for the sample OLAP model and metaoutline:

➤ To import the sample OLAP model:

1. Start Integration Services Console.
2. Select **File**, then **XML Import/Export**.
   - The XML Import/Export window is displayed.
3. Select the **Import** tab.
4. In the **XML File Path** text box, enter the file path for the sample OLAP model or click the **Open XML File** button and navigate to the following file:
   ```plaintext
   EPM_ORACLE_HOME\products\Essbase\eis\server\samples\tbcmodel\HA_TBC_Model.xml
   ```
5. Click **Open XML File**.

➤ To import the sample metaoutline:

1. Select **File**, then **XML Import/Export**.
   - The XML Import/Export window is displayed.
2. Select the **Import** tab.
3. In the **XML File Path** text box, enter the file path for the sample OLAP metaoutline or click the **Open XML File** button and navigate to the following file:
   ```plaintext
   EPM_ORACLE_HOME\products\Essbase\eis\server\samples\tbcmodel\HA_TBC_Metaoutline.xml
   ```
4. Click **Open XML File**.

### Setting Up Hybrid Analysis Sample Application on Oracle

You use several files to create the Hybrid Analysis sample application on Oracle:

- A batch file to send commands to the SQLPLUS command line interface:
  ```plaintext
  install_oracle.bat
  ```
- A data load script for the TBC sample database:
  ```plaintext
  ha_create_oracle.sql
  ```
- An Oracle control file to load data into the fact table:
  ```plaintext
  hasales.ctl
  ```
• An XML file that you use to import a sample OLAP model into the TBC_MD OLAP Metadata Catalog database:
  HA_TBC_Model.xml
• An XML file you use to import a sample metaoutline into the TBC_MD OLAP Metadata Catalog database:
  HA_TBC_Metaoutline.xml
• A text file containing sample data:
  ha_sampledata.txt

Setting up the Hybrid Analysis sample application on Oracle consists of performing the following tasks:

• Running the batch file install_oracle.bat
• Importing the sample OLAP model and metaoutline

**Note:** Before setting up the Hybrid Analysis Sample Application on Oracle, verify that SQL*PLUS and SQLLDR command line interfaces are installed.

### Running the Batch File

The first task in setting up the Hybrid Analysis sample application on Oracle consists of running the install_oracle.bat file. This batch file creates tables in the sample database, loads metadata into the dimension tables, and loads sample data into the fact table.

Run the install_oracle.bat file located in:
EPM_ORACLE_HOME\products\Essbase\eis\server\samples\tbcdbase

This batch file sends SQL commands to the SQL*PLUS command line interface.

Pass the following three parameters to install_oracle.bat:

• The user ID that you use to connect with Oracle
• Your password
• The Oracle service name as defined in the TNSNAMES.ORA file

### Metadata Load Script

The install_oracle.bat file runs the SQLPLUS command line interface, which executes the ha_create_oracle.sql metadata load script file. This script file performs the following tasks:

• Deletes any existing tables in the sample TBC database
• Creates a new set of tables in the TBC database
• Inserts records into the dimension tables
• Creates the ha_results_createtables.txt file

The ha_results_createtables.txt file contains the results of the table-creating operation.
**Sample Data Load Script**

After loading the metadata into the dimension tables, the `install_oracle.bat` file runs the SQLLDR command line interface, which performs the following tasks:

- Loads the sample data into the fact table
- Creates the `ha_results_loadfact.txt` file

The `ha_results_loadfact.txt` file contains the results of the data-loading operation.

**Importing the Sample OLAP Model and Metaoutline**

The final task in setting up the Hybrid Analysis sample application on Oracle consists of importing the two XML files for the sample OLAP model and metaoutline:

1. To import the sample OLAP model:
   1. Start Integration Services Console.
   2. Select File > XML Import/Export.
      The XML Import/Export dialog box is displayed.
   3. Select the Import tab.
   4. In the XML File Path text box, enter the file path for the sample OLAP model or click the Open XML File button and navigate to the following file:
      
      `EPM_ORACLE_HOME\products\Essbase\eis\server\samples\tbcmodel\HA_TBC_Model.xml`
   5. Click Open XML File.

2. To import the sample metaoutline:
   1. Select File > XML Import/Export.
      The XML Import/Export dialog box is displayed.
   2. Select the Import tab.
   3. In the XML File Path text box, enter the file path for the sample OLAP metaoutline or click the Open XML File button and navigate to the following file:
      
      `EPM_ORACLE_HOME\products\Essbase\eis\server\Samples\tbcmodel\HA_TBC_Metaoutline.xml`
   4. Click Open XML File.

**Setting Up Hybrid Analysis Sample Application on Microsoft SQL Server**

Several files are used to create the Hybrid Analysis sample application on Microsoft SQL Server:

- A batch file to send commands to MS Interface SQL (ISQL):
  
  `install_sqlsrv.bat`
- A data load script for the TBC sample database:
Setting up the Hybrid Analysis sample application on Microsoft SQL Server consists of performing the following tasks:

- Running the `install_sqlsrv.bat` batch file
- Importing the sample OLAP model and metaoutline

**Note:** Before setting up the Hybrid Analysis Sample Application on Microsoft SQL Server, verify that you have ISQL and BCP (Bulk Copy) installed. These utilities were most likely included when you installed SQL Server Client.

### Running the Batch File

To set up the Hybrid Analysis sample application on Microsoft SQL Server, you first run the `install_sqlsrv.bat` file. This batch file creates tables in the sample database, loads metadata into the dimension tables, and loads sample data into the fact table. Run the `install_sqlsrv.bat` file located in:

```
EPM_ORACLE_HOME\products\Essbase\eis\server\Samples\tbcdbase
```

This file sends commands to the Interface SQL (ISQL) command line interface. These commands install the sample application to the sample database TBC.

Pass the following three parameters to `install_sqlsrv.bat`:

- The user ID that you use to connect with the Microsoft SQL Server (TBC)
- Your password
- The Microsoft SQL Server name (name of machine where the SQL Server is installed)

### Metadata Load Script

The `install_sqlsrv.bat` file runs the ISQL utility, which executes the `ha_create_sqlsrv.sql` metadata load script file. This file performs the following tasks:

- Deletes any existing tables in the sample TBC database
- Creates a new set of tables in the TBC database
- Inserts records into the dimension tables
• Creates the ha_results_createtables.txt file

The ha_results_createtables.txt file contains the results of the table-creating operation.

Sample Data Load Script

After loading the metadata into the dimension tables, the install_sqlsrv.bat file runs the BCP (Bulk Copy) utility which performs the following tasks:

• Loads the sample data into the fact table
• Creates the ha_results_loadfact.txt file

The ha_results_loadfact.txt file contains the results of the data-loading operation.

Importing the Sample OLAP Model and Metaoutline

The final task in setting up the Hybrid Analysis sample application on Microsoft SQL Server consists of importing the two XML files for the sample OLAP model and metaoutline:

➤ To import the sample OLAP model:

1  Start Integration Services Console.
2  Select File > XML Import/Export.
    The XML Import/Export dialog box is displayed.
3  Select the Import tab.
4  In the XML File Path text box, enter the file path for the sample OLAP model or click the Open XML File button and navigate to the following file:
    EPM_ORACLE_HOME\products\Essbase\eis\server\samples\tbcmodel\HA_TBC_Model.xml
5  Click Open XML File.

➤ To import the sample metaoutline:

1  Select File > XML Import/Export.
    The XML Import/Export dialog box is displayed.
2  Select the Import tab.
3  In the XML File Path text box, enter the file path for the sample OLAP metaoutline or click the Open XML File button and navigate to the following file:
    EPM_ORACLE_HOME\products\Essbase\eis\server\Samples\tbcmodel\HA_TBC_Metaoutline.xml
4  Click Open XML File.

Setting Up Hybrid Analysis Sample Application on Teradata

Several files create the Hybrid Analysis sample application on Teradata:
• A batch file to send commands to MS Interface SQL (ISQL):
  install_teradata.bat
• A data load script for the TBC sample database:
  ha_create_teradata.sql
• An SQL file to add constraints to the fact table after the data load has been completed:
  ha_altertable_teradata.sql
• A FastLoad script file, which loads data to the fact table:
  ha_fastload_teradata.sql
• An XML file that you use to import a sample OLAP model into the TBC_MD OLAP Metadata Catalog database:
  HA_TBC_Model.xml
• An XML file that you use to import a sample metaoutline into the TBC_MD OLAP Metadata Catalog database:
  HA_TBC_Metaoutline.xml
• A text file containing sample data:
  ha_sampledata.txt

Setting up the Hybrid Analysis sample application on Microsoft SQL Server consists of performing the following tasks:

• Running the batch file install_teradata.bat
• Importing the sample OLAP model and metaoutline

**Note:** Before setting up the Hybrid Analysis Sample Application on Teradata, verify that the BTEQ and FastLoad utilities are installed.

**Running the Batch File**

The first task in setting up the Hybrid Analysis sample application on Teradata consists of running the install_teradata.bat file. This batch file creates tables in the sample database, loads metadata into the dimension tables, loads sample data into the fact table, and adds constraints to the fact table.

Run the install_teradata.bat file located in:

EPM_ORACLE_HOME\products\Essbase\eis\server\Samples\tbcdbase

This batch file sends SQL commands to the BTEQ command line interface.

Pass the following three required parameters to install_teradata.bat:

• The user ID that you use to connect with Teradata
• Your password
• The Teradata database name (DBC)
You can also pass, as an optional parameter, the database name where you want to install the sample application. (The default database is TBC.)

**Metadata Load Script**

The `install_teradata.bat` file runs the BTEQ utility, which executes the `ha_create_teradata.sql` metadata load script file. This script file performs the following tasks:

- Deletes any existing tables in the sample TBC database
- Creates a new set of tables in the TBC database
- Inserts records into the dimension tables
- Creates the `ha_results_createtables.txt` file

The `ha_results_createtables.txt` file contains the results of the table-creating operation.

**Sample Data Load Script**

After loading the metadata into the dimension tables, the `install_teradata.bat` file runs the FastLoad utility, which performs the following tasks:

- Loads the sample data into the fact table using the `ha_fastload_teradata.txt` file
- Creates the `ha_results_loadfact.txt` file

The `ha_results_loadfact.txt` file contains the results of the data-loading operation.

**Alter Table Script**

After loading the sample data into the fact table, the `install_teradata.bat` file runs the BTEQ utility, which executes the `ha_altertable_teradata.sql` alter table script file. This script file performs the following tasks:

- Deletes a few tables no longer needed in the sample TBC database
- Creates indexes and places constraints on the fact table
- Creates the `ha_results_altertables.txt` file

The `ha_results_altertables.txt` file contains the results of the table-altering operation.

**Importing the Sample OLAP Model and Metaoutline**

The final task in setting up the Hybrid Analysis sample application on Teradata consists of importing the two XML files for the sample OLAP model and metaoutline:

1. To import the sample OLAP model:
   - Start Integration Services Console.
2. Select File > XML Import/Export.
   
   The XML Import/Export window is displayed.

3. Select the Import tab.

4. In the XML File Path text box, enter the file path for the sample OLAP model or click the Open XML File button and navigate to the following file:

   EPM_ORACLE_HOME\products\Essbase\eis\server\Samples\tbcmodel\HA_TBC_Model.xml

5. Click Open XML File.

---

To import the sample metaoutline:

1. Select File > XML Import/Export.

   The XML Import/Export window is displayed.

2. Select the Import tab.

3. In the XML File Path text box, enter the file path for the sample OLAP metaoutline or click the Open XML File button and navigate to the following file:

   EPM_ORACLE_HOME\products\Essbase\eis\server\Samples\tbcmodel\HA_TBC_Metaoutline.xml

4. Click Open XML File.

---

### Setting Up the Unicode Sample Application

Setting up the Unicode sample application is a manual process.

**Note:** The Unicode sample application in Integration Services is supported only on Oracle relational database management systems (RDBMSs).

The utilities listed in Table 14 are required for running the various batch files and SQL scripts used in creating the Unicode sample application.

<table>
<thead>
<tr>
<th>Database</th>
<th>Required Utilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle</td>
<td>• SQL*Plus</td>
</tr>
<tr>
<td></td>
<td>• SQL*Loader command line utility</td>
</tr>
</tbody>
</table>

### Creating the Unicode TBC_U Database

Create the Unicode TBC_U database in the same way that you create any database using an RDBMS:

- Create a database device or tablespace named TBC_U.
- Allot 20 MB for storage.
Create a user TBC_U who can drop and create tables.

Grant user privileges or permissions to create and drop tables.

**Caution!** You should create the tables for the Unicode-enabled database as user TBC_U, or portions of the Unicode sample application may not work.

---

**Creating the Unicode TBC_MD_U OLAP Metadata Catalog**

Create the Unicode TBC_MD_U OLAP Metadata Catalog by following the steps in “Setting Up the TBC_MD OLAP Metadata Catalog” on page 96, using the file `oc_create_database_name_unicode.sql`.

---

**Setting Up the Unicode Sample Application**

**Note:** The Unicode sample application in Integration Services is supported only on Oracle relational database management systems (RDBMSs).

You use several files to create the Unicode sample application:

- A table creation and data load script for the Unicode sample database:
  
  `tbc_create_database_name_unicode.sql`

- An XML file that you use to import a sample OLAP model into the TBC_MD_Unicode OLAP Metadata Catalog database:
  
  `TBC_Model_Unicode.xml`

- An XML file that you use to import a sample metaoutline into the TBC_MD_Unicode OLAP Metadata Catalog database:
  
  `TBC_Metaoutline_Unicode.xml`

Setting up the Unicode sample application consists of performing the following tasks:

- Running the SQL file `tbc_create_database_name_unicode.sql`
- Importing the Unicode sample model
- Importing the Unicode sample metaoutline

---

**Running the SQL File**

The first task in setting up the Unicode sample application consists of running the `tbc_create_database_name_unicode.sql` file using one of the utilities listed in Table 14 on page 110.

**Note:** Ensure you have created a Unicode-enabled database for your sample application as described in “Creating the Unicode TBC_U Database” on page 110.
Run the `tbc_create_database_name_unicode.sql` file located in:

EPM_ORACLE_HOME/products/Essbase/eis/server/samples/tbcdbase/

The `tbc_create_database_name_unicode.sql` file performs the following tasks:

- Deletes any existing tables in the Unicode sample database
- Creates a new set of tables in the Unicode sample database
- Loads Unicode sample metadata into the dimension tables
- Loads Unicode sample data into the fact table

**Importing the Unicode Sample Model and Metaoutline**

The final task in setting up the Unicode sample application consists of importing the XML file for the Unicode sample model:

- To import the Unicode sample OLAP model:
  1. Start Integration Services Console and log in to the Unicode catalog.
  2. Select File>XML Import/Export.
  
     The XML Import/Export dialog box is displayed.
  3. Select the Import tab.
  4. In the XML File Path text box, enter the file path for the Unicode sample OLAP model or click the Open XML File button and navigate to the following file:

     `eis\server\samples\tbcmodel\TBC_Model_Unicode.xml`
  5. Select the file and click Import to Catalog.
  
     A message is displayed when the import is complete.
  6. Click OK.

- To import the Unicode sample metaoutline:
  1. If necessary, select File>XML Import/Export.
  
     The XML Import/Export dialog box is displayed.
  2. Select the Import tab.
  3. In the XML File Path text box, enter the file path for the Unicode sample metaoutline or click the Open XML File button and navigate to the following file:

     `EPM_ORACLE_HOME/products/Essbase/eis\server\Samples\tbcmodel\TBC_Metaoutline_Unicode.xml`
  4. Select the file and click Import to Catalog.
  
     A message is displayed when the import is complete.
  5. Click OK.
  6. Click Close to close the XML Import/Export dialog box.
After Setting Up the Sample Application

You must connect to both TBC and TBC_MD from Integration Services Console to create, modify, and store TBC OLAP models and TBC metaoutlines. To make these connections, each database (TBC and TBC_MD) must be mapped to a supported ODBC driver, as described in Chapter 2, “Configuring Data Sources”.

When you connect to Integration Services Console, you can view TBC columns, tables, OLAP models, and metaoutlines in Integration Services Console. For more information, see “Viewing TBC Tables and Columns” on page 113.

Note: You must connect to TBC and TBC_MD as user TBC unless you create user name aliases or synonyms in the RDBMS.

Viewing TBC Tables and Columns

After you set up the sample application and configure TBC and TBC_MD by mapping them to supported ODBC drivers, you can view TBC tables and columns in Integration Services Console.

➢ To see TBC tables and columns in the left frame of Integration Services Console:
  1. Start Integration Server.
  2. Start Integration Services Console.
  3. In a blank Integration Services Console window, select Connections, then OLAP Metadata Catalog, then Connect, and connect to the OLAP Metadata Catalog, TBC_MD.
  4. In the New tab, double-click the OLAP Model icon.
  5. In the Data Source dialog box, connect to the TBC sample database.

Viewing OLAP Models

After you set up the sample application and configure TBC and TBC_MD by mapping them to supported ODBC drivers, you can view OLAP models in Integration Services Console.

➢ To see the OLAP model (TBC Model) in the right frame of Integration Services Console:
  1. Start Integration Server.
  2. Start Integration Services Console.
  3. If the Login dialog box is not already displayed, in a blank Integration Services Console window, select Connections, then OLAP Metadata Catalog, then Connect.
  4. Connect to the OLAP Metadata Catalog TBC_MD.
  5. Click the Existing tab, select TBC Model, and click Open.

  Hybrid Analysis: To view the Hybrid Analysis-enabled OLAP model, select HA TBC Model.
Duplicate Member (Oracle only): To view the Duplicate Member-enabled OLAP model, select Duplicate Member TBC Model.

6. In the Data Source dialog box, connect to the TBC sample database.

Viewing Metaoutlines

After you set up the sample application and configure TBC and TBC_MD by mapping them to supported ODBC drivers, you can view metaoutlines in Integration Services Console.

To see the metaoutline (TBC Metaoutline) in the right frame of Integration Services Console:
1. Start Integration Server.
2. Start Integration Services Console.
3. If the Login dialog box is not already displayed, in a blank Integration Services Console window, select Connections, then OLAP Metadata Catalog, then Connect.
4. Connect to the OLAP Metadata Catalog TBC_MD.
5. Click the Existing tab, then click the plus symbol (+) to the left of TBC Model to expand the view, and select TBC Metaoutline.

   Hybrid Analysis: To view the Hybrid Analysis-enabled metaoutline, click the plus symbol (+) to the left of HA TBC Model to expand the view, and select HA TBC Metaoutline.

   Duplicate Member (Oracle only): To view the Duplicate Member-enabled metaoutline, click the plus symbol (+) to the left of Sample_duplicate member to expand the view, and select Duplicate Member TBC Metaoutline.

6. Click Open.

The Data Source dialog box is displayed.

7. In the Data Source dialog box, connect to the TBC sample database.
This chapter describes how to view information about users, OLAP models, and metaoutlines in OLAP Metadata Catalog. It also describes how to clear locks and change permissions on OLAP models and metaoutlines.

See Chapter 4, “Tables in OLAP Metadata Catalog.”

**Working with Users**

When working with Integration Services, you must manage access for three types of users:

- Integration Services users who can access OLAP models and metaoutlines stored in OLAP Metadata Catalogs. (Integration Services users are the database users defined in the data source that contains the OLAP Metadata Catalog that you are using.)
- Essbase users who can access the Essbase database that you create from a metaoutline
- Source database and data warehouse users who can access the tables and views in the specified source database or data warehouse

Use the tools provided with each data source to manage each set of users. Consult the data source documentation if you are not sure how to perform any of the following tasks:

- Create new users
- View a list of available users
- Change the permissions for users
- Delete current users
- Disconnect users
Working with Locks

You can use either of two methods for opening an OLAP model or metaoutline in Integration Services Console: standard access mode (the default mode) and exclusive access mode. The locks that are established by these different modes are specific to Integration Services and are completely separate from the locking performed by the data source database.

Tip: If you plan only to view an OLAP model or metaoutline, use standard access mode. If you plan to edit an OLAP model or metaoutline, use exclusive access mode.

Using Standard Access Mode

When you open an OLAP model in standard access mode, Integration Services gives you a read lock on that OLAP model. When you open a metaoutline in standard access mode, Integration Services gives you a read lock on that metaoutline and a read lock on the OLAP model on which the metaoutline is based. Multiple users can have read locks on the same OLAP model or metaoutline at the same time.

When you save an OLAP model or metaoutline in standard access mode, Integration Services upgrades the read lock to a write lock for the duration of the save. After Integration Services saves the changes to the OLAP model or metaoutline, the lock reverts back to a read lock. However, if other users have the OLAP model or metaoutline open, your changes are not updated for the other users until they reopen the document.

In addition, if other users have the OLAP model or metaoutline open at the same time as you do, they can edit and save it, thereby overwriting your changes. Using exclusive access mode prevents users from overwriting your changes. See “Using Exclusive Access Mode” on page 116.

If you opened and edited an OLAP model or metaoutline in standard access mode, and are concerned that other users may overwrite your changes, perform one of the following actions:

- Save the OLAP model or metaoutline using a different name. See Integration Services Console Help.
- If you have not yet started editing the document or have entered only a few changes, close the OLAP model or metaoutline and then reopen it using exclusive access mode and reapply the changes.
- Contact the users with read locks and ask them to close the OLAP model or metaoutline. To view a list of users with read locks, see “Viewing Integration Services Users with Locks” on page 117.

Using Exclusive Access Mode

Opening an OLAP model or metaoutline in exclusive access mode eliminates the risk of other users overwriting your changes. When you open an OLAP model in exclusive access mode,
Integration Services gives you a write lock on that OLAP model that remains in effect until you close the model. When you open a metaoutline in exclusive access mode, Integration Services gives you a write lock on that metaoutline and a read lock on the OLAP model on which the metaoutline is based that remains in effect until you close the metaoutline. While you have an OLAP model or metaoutline open in exclusive access mode, other users can open and view the documents, but they cannot save them.

**Correcting Problems with Locks**

Due to the complex nature of Integration Services and the other software components with which it interacts, you may experience problems that result in locks not being released from OLAP models or metaoutlines. Unreleased locks can result in an inability to save, even if no other users are on the system. Unreleased locks can also prevent you from opening an OLAP model or metaoutline.

If you suspect that unreleased locks remain from previous sessions, delete the locks by selecting *Servers*, then *OLAP Metadata Catalog*, then *Delete Locks*. Deleting locks from the menu removes only your locks; it does not delete locks held by other users.

If you suspect that unreleased locks remain from other user sessions, check to see which users have locks. See “Viewing Integration Services Users with Locks” on page 117. If you are convinced that some or all of the locks are from terminated user sessions, delete them. See “Deleting Locks for Integration Services Users” on page 118.

**Viewing Integration Services Users with Locks**

The CB_CONCURRENCY table in OLAP Metadata Catalog contains information about the users who access OLAP models and metaoutlines. This table contains the following columns:

- **CB_USER_NAME**: the name of the user accessing the OLAP model or metaoutline; for example, *sa*.
- **CB_USER_SESSION_ID**: a system-generated reference number that uniquely identifies an editing session
- **CB_OBJECT_ID**: an OLAP model or metaoutline number
- **CB_OBJECT_TYPE**: a type reference that indicates whether the user is accessing an OLAP model or a metaoutline
  - The number 1 represents an OLAP model.
  - The number 2 represents a metaoutline.
- **CB_OBJECT_LOCK**: a lock reference number that indicates whether the user has a read or write lock
  - The number 1 represents a read lock.
  - The number 2 represents a write lock.
- **CB_LOCK_TIMESTAMP**: the date and time that the user acquired the lock
To determine which users are accessing a specific OLAP model or metaoutline:

1. **Enter the following SELECT statement in the OLAP Metadata Catalog database, using the tools for the data source you are using:**

   ```sql
   SELECT CB_USER_NAME, CB_OBJECT_ID, CB_OBJECT_TYPE, CB_OBJECT_LOCK, CB_LOCK_TIMESTAMP
   FROM CB_CONCURRENCY
   ```

2. **View the results.**

   In the following example, the `sa` user is the only user currently accessing an OLAP model or metaoutline. The `sa` user has two read locks on one OLAP model and one read lock each on two different metaoutlines.

<table>
<thead>
<tr>
<th>CB_USER_NAME</th>
<th>CB_OBJECT_ID</th>
<th>CB_OBJECT_TYPE</th>
<th>CB_OBJECT_LOCK</th>
<th>CB_LOCK_TIMESTAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sa</code></td>
<td>889844639</td>
<td>1</td>
<td>1</td>
<td>Apr 9 2004 4:43PM</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>889845263</td>
<td>2</td>
<td>1</td>
<td>Apr 9 2004 4:43PM</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>889844639</td>
<td>1</td>
<td>1</td>
<td>Apr 9 2004 5:20PM</td>
</tr>
<tr>
<td><code>sa</code></td>
<td>892167813</td>
<td>2</td>
<td>1</td>
<td>Apr 9 2004 5:20PM</td>
</tr>
</tbody>
</table>

   (4 row(s) affected)

   You can determine the following information from the sample CB_CONCURRENCY table shown in the preceding list:

   - The first row of the results tells you that the `sa` user (CB_USER_NAME = `sa`) has a read lock (CB_OBJECT_LOCK = 1) on an OLAP model (CB_OBJECT_TYPE = 1) with an ID of 889844639.
   - The second row of the results tells you that the `sa` user (CB_USER_NAME = `sa`) has a read lock (CB_OBJECT_LOCK = 1) on a metaoutline (CB_OBJECT_TYPE = 2) with an ID of 889845263.
   - The third row of the results tells you that the `sa` user (CB_USER_NAME = `sa`) has a read lock (CB_OBJECT_LOCK = 1) on an OLAP model (CB_OBJECT_TYPE = 1) with an ID of 889844639.
   - The fourth row of the results tells you that the `sa` user (CB_USER_NAME = `sa`) has a read lock (CB_OBJECT_LOCK = 1) on a metaoutline (CB_OBJECT_TYPE = 2) with an ID of 892167813.

   When you open a metaoutline, you receive a read lock on the metaoutline and on the OLAP model on which it is based; therefore, you can assume that the `sa` user is working on two different metaoutlines based on the same OLAP model.

### Deleting Locks for Integration Services Users

If you are certain that the other users who have read or write locks on an OLAP model or a metaoutline that you want to save do not need their locks, delete their locks from the CB_CONCURRENCY table.

**Caution!** Make sure the users do not need their locks before you delete the locks.
To delete read and write locks on OLAP models or metaoutlines:

1. **Determine which users have locks.**
   
   See “Viewing Integration Services Users with Locks” on page 117.

2. **Delete the rows containing the unwanted locks.**
   
   For example, to delete all locks held by the `sa` user, issue the following DELETE statement in the OLAP Metadata Catalog database, using the tools for the data source:
   
   ```sql
   DELETE FROM CB_CONCURRENCY WHERE CB_USER_NAME = 'sa'
   ```

### Working with Permissions

When you save an OLAP model or metaoutline for the first time, you determine what read or write permissions other users have. Integration Services supports the following kinds of permissions:

- **Allow read/write access for other users.** This setting permits all other users to read and write to the OLAP model or metaoutline. This setting is the default.

- **Allow read access for other users.** This setting permits all other users to read but not write to the OLAP model or metaoutline; that is, other users cannot save changes to the OLAP model or metaoutline.

- **Disallow all access for other users.** This setting denies all other users read or write permission to the OLAP model or metaoutline. You are the only user who can read or write to it.

To change the permissions of an OLAP model or metaoutline, take one of the following actions:

- Change the OLAP model properties in Integration Services Console. See the Integration Services Console Help.

- Save the metaoutline with a different name by selecting **File**, then **Save As**. See the Integration Services Console Help.

- To edit tables containing the permissions for the OLAP model or metaoutline, use the tools provided with the data source that contains OLAP Metadata Catalog. See “Viewing Permissions for OLAP Models” on page 119 or “Viewing Permissions for Metaoutlines” on page 120.

### Viewing Permissions for OLAP Models

Information about the permissions that are set for OLAP models and metaoutlines is stored in OLAP Metadata Catalog. View this information by selecting the appropriate columns from tables in OLAP Metadata Catalog.

The OM_INFO table in OLAP Metadata Catalog contains information about OLAP models, including the following columns which are relevant to permissions:

- **MODEL_ID**: a system-generated reference number.
MODEL_NAME: the name of the OLAP model; for example, TBC Model.

MODEL_DESCRIPTION: a description of the OLAP model. If you do not enter a description when you save the model, this column is blank.

MODEL_DATA_SOURCE: the name of the Open Database Connectivity (ODBC) data source on which the OLAP model is based; for example, TBC.

MODEL_OWNER: the login name of the OLAP model owner; for example, sa. The login name is specified in the data source that contains the OLAP Metadata Catalog.

MODEL_ACCESS_CODE: a reference number that indicates what level of access users, other than the owner, have to the OLAP model.

- The number 0 represents no permissions—other users can neither read nor write to the OLAP model.
- The number 1 represents read access—other users can read the OLAP model but cannot write to it.
- The number 2 represents read and write access—other users can both read and write to the OLAP model; this is the default.

To view access permissions for all OLAP models in OLAP Metadata Catalog:

1. Issue the following SELECT statement in the OLAP Metadata Catalog database, using the tools for the data source.

   ```sql
   SELECT MODEL_NAME, MODEL_OWNER, MODEL_ACCESS_CODE
   FROM OM_INFO
   ```

2. View the results.

   In the following example, TBC Model gives read and write permissions to other users (MODEL_ACCESS_CODE = 2). TBC_Mod_Archive gives read permissions to other users (MODEL_ACCESS_CODE = 1). TBC_Mod_Mine gives neither read nor write permissions to other users (MODEL_ACCESS_CODE = 0).

   ```sql
   MODEL_NAME         MODEL_OWNER | MODEL_ACCESS_CODE
   --------------- | ----------- | ----------------- 
   TBC Model        sa         | 2
   TBC_Mod_Archive  sa         | 1
   TBC_Mod_Mine     sa         | 0
   (3 row(s) affected)
   ```

### Viewing Permissions for Metaoutlines

OLAP Metadata Catalog stores information about the permissions set for OLAP models and metaoutlines. View this information by selecting the appropriate columns from tables in OLAP Metadata Catalog.

The MO_INFO table in OLAP Metadata Catalog contains information about metaoutlines, including the following columns which are relevant to permissions:

- MO_ID: a system-generated reference number for the metaoutline.
- **MODEL_ID**: a system-generated reference number for the OLAP model on which the metaoutline is based.

- **MO_NAME**: the name of the metaoutline; for example, TBC Metaoutline.

- **MO_DESC**: a description of the metaoutline. If you do not enter a description when you save the metaoutline, this column is blank.

- **MO_CHANGE_DATE**: the date on which changes were last made to the metaoutline.

- **MO_OWNER**: the login name of the metaoutline owner; for example, sa. The login name is specified in the database that contains the associated OLAP Metadata Catalog.

- **MO_ACCESS_CODE**: a reference number that indicates what level of access users, other than the owner, have to the metaoutline.
  - The number 0 represents no permissions—other users can neither read nor write to the metaoutline.
  - The number 1 represents read access—other users can read the metaoutline but cannot write to it.
  - The number 2 represents read and write access—other users can both read and write to the metaoutline; this setting is the default.

To determine the access permissions for all metaoutlines in OLAP Metadata Catalog:

1. **Issue the following SELECT statement in the OLAP Metadata Catalog database, using the tools for the data source.**

   ```sql
   SELECT MO_NAME, MO_OWNER, MO_ACCESS_CODE
   FROM MO_INFO
   ```

2. **View the results.**

   In the following example, TBC Metaoutline gives read and write permissions to other users (MO_ACCESS_CODE = 2). TBC_Archive gives read permissions to other users (MO_ACCESS_CODE = 1). TBC_Mine gives neither read nor write permissions to other users (MO_ACCESS_CODE = 0).

<table>
<thead>
<tr>
<th>MO_NAME</th>
<th>MO_OWNER</th>
<th>MO_ACCESS_CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBC Metaoutline</td>
<td>sa</td>
<td>2</td>
</tr>
<tr>
<td>TBC_Archive</td>
<td>sa</td>
<td>1</td>
</tr>
<tr>
<td>TBC_Mine</td>
<td>sa</td>
<td>0</td>
</tr>
</tbody>
</table>

(3 row(s) affected)
Integration Services requires several layers of software components to connect to data sources and load data into Essbase databases. Each of the components must be configured properly to run and communicate with each other. Consequently, diagnosing and correcting connection problems in the system can be challenging.

This chapter assists you in isolating and correcting connection problems that can occur during the operation of Integration Services. The first topic of this chapter lists common causes for connection problems. The subsequent topics outline a strategy for isolating and correcting connection problems not addressed by the common problems list.

This chapter assumes that you know how to perform basic database administration tasks, such as using a database client utility program to connect to a database server, navigating directories using a command prompt, and editing configuration files.

**Common Connection Problems**

If you have problems connecting to OLAP Metadata Catalog or to the external data source with Essbase Integration Services Console, review the following list of common problems:

- Are you using the correct user name and password?
  - Does the user name you used have the correct privileges to access both OLAP Metadata Catalog and the data source at the database level?
  - If you are trying to connect to OLAP Metadata Catalog, did you use the same user name and password as the user who created the tables in OLAP Metadata Catalog?

If you create an OLAP Metadata Catalog when logged in as one user name, you cannot access the tables in OLAP Metadata Catalog using a different user name unless you create
an *alias* for the user name (for Microsoft SQL Server) or synonyms for the tables (for IBM DB2 and Oracle).

- Are all required components up and running?
  - Essbase Integration Server
  - The database servers that manage OLAP Metadata Catalog and the data source databases
  - The data source database listener for OLAP Metadata Catalog and the data source

- Is the database client software installed and configured on the computer where Integration Server is installed?
  - Do the database client software utility programs, such as the command-line SQL utility, run? Can you connect to databases with them?
  - Does the user who starts Integration Server have the correct environment variables set and the necessary access privileges to run the database client software?
  - Are the required environment variables for the database client software, such as the path (and the library path on UNIX), configured on the computer where Integration Server is installed?
    - On UNIX, use `ivtestlib` (32-bit) or `ddtestlib` (64-bit) to confirm that the library path contains all files required by the open database connectivity (ODBC) driver.
      - `ivtestlib` is located in `common/odbc/Merant/6.1/bin`
      - `ddtestlib` is located in `common/odbc64/Merant/6.1/bin`
  - If necessary, is the bin directory for each database in the operating system path of the user who is running Integration Server?
  - Does the ODBC data source name (DSN) configuration identify the name of the data source client?

- Are OLAP Metadata Catalog and the data source configured as ODBC data sources on the Integration Server computer? For information on configuring ODBC data sources, see Chapter 2, “Configuring Data Sources”.

- Are you using the ODBC data source names for OLAP Metadata Catalog and the data source databases as configured on the Integration Server computer? Do not assume that the data source database name is the same as the ODBC data source name for that database.

- Are you experiencing time-out problems when connecting to the external data source?
  - Wait and retry the action when the data source is less busy. To avoid the problem, increase the ODBC driver time-out limit using ODBC Administrator on Windows systems. See the ODBC documentation.

- Did you create tables in OLAP Metadata Catalog? For information on creating tables for the catalog, see Chapter 3, “Creating, Upgrading, and Deleting OLAP Metadata Catalogs”.

- Are you having problems related to ODBC while you are creating an Essbase outline? If so, turn on tracing, perform the action again, and analyze the results in the tracing log file. See “Using ODBC Tracing on Windows Systems” on page 144.

124 Troubleshooting ODBC and Connections
If you experience problems connecting to the data source, view the log for the data source for information about why the connection failed. See your data source documentation.

If none of the above steps help, review the list of database-specific problems in “Database-Specific Problems” on page 125.

**Database-Specific Problems**

If you have problems connecting to OLAP Metadata Catalog or to the external data source with Integration Services Console, review the following list of common problems for specific databases:

- **IBM DB2**: You may be required to bind the database driver to a IBM DB2 server that is managing the databases you want. See the IBM DB2 installation documentation.

- **Microsoft SQL Server**: If you try to access a Microsoft SQL Server database with the Microsoft-supplied ODBC driver when you do not have access permission, SQL Server connects you to the default database and does not notify you. Configure the ODBC DSN to use the database that you want to access. See the Microsoft SQL Server documentation.

If none of the previous suggestions help, proceed to “ODBC and Connection Problems” on page 125 for an overview of connection problems, then see “Isolating Connection Problems” on page 126.

**ODBC and Connection Problems**

Integration Services Console must pass through three layers of software components to contact a data source: Integration Server, ODBC, and the database client software. Failure within or between any of these components can cause a lack of communication between Integration Services Console and the data source.

In some cases, the error messages received in Integration Services Console may not contain sufficient information for you to diagnose and resolve the problem, and you must go to the Integration Server to get more information and find a resolution.

The three main categories of problems that break the lines of communication between Integration Services Console and the data source are:

- **Software component problems**:
  - The components of Integration Services or the data source are not installed or are installed on the wrong computer.
  - The components do not run because they are not set up properly.
  - The components stopped running or were never started.
  - The components are not compatible with the operating system.
  - The versions of the different components are not compatible with each other.

- **Configuration and communication problems**:
The software components are not properly configured to communicate with each other.

User names, passwords, and permissions are configured or used inconsistently across the components and computers.

The communication interfaces of the software components are incompatible with each other.

Changes to components, databases, users, or permissions have put the software components out of sync with each other.

Network and security problems:

- A physical network path does not exist between the computers that must communicate with each other.
- One or both computers attempting to communicate do not have the TCP/IP network protocol installed or configured.
- The computer attempting to contact another computer does not have access to addressing information, such as a domain name server (DNS).
- Security systems prevent all communication or certain types of communication between the computers that must talk with each other. For example, this is the case when client and server computers are separated by a firewall.

**Isolating Connection Problems**

If the list under “Common Connection Problems” on page 123 and your own knowledge fail to diagnose the connection problem, the recommended strategy is to test the components and connections starting at the data source and backtracking through the connections to Integration Services Console.

The client software must pass through three layers of components to connect to the data source containing OLAP Metadata Catalog and the data source, as illustrated in Figure 4.
The numbers in Figure 4 identify the general diagnostic steps that you should take to isolate connection problems. Follow these steps as indicated, starting with the connection between the database client software and the data source (1) and working backwards through the system to isolate connection problems. After you isolate a problem, refer to the appropriate components topic in “Correcting Connection Problems” on page 136.

To isolate connection problems:

1. Attempt to connect to the data source databases using a database client utility program. See “Testing the Database Client Connection to the Data Source” on page 128.

2. Attempt to connect to the data source databases using ODBC. See “Testing the ODBC Connection to the Data Source” on page 129.

3. Attempt to connect to the data source databases using Integration Server. See “Testing the Integration Server Connection to the Data Source” on page 132.
Attempt to connect to the data source databases using Integration Services Console. See “Testing the Integration Services Console Connection to Essbase Server” on page 136.

Attempt to connect to Essbase Server using Integration Server. See “Testing the Integration Server Connection to Essbase Server” on page 134.

Attempt to connect to Essbase Server using Integration Services Console. See “Testing the Integration Server Connection to Essbase Server” on page 134.

Testing the Database Client Connection to the Data Source

If your ODBC connections require database client software, the first step toward isolating connection problems in Integration Services is to attempt to connect the database client software to the data source that you are using. After you have verified that these components are communicating properly, you have a good base from which to test the remainder of the connection chain.

The most likely cause of a connection failure at this point is that the database client software environment variables are not included in the login script for the user who runs the Integration Server program.

To test the database client connection to the data source:

1. Log on to the computer running Integration Server as the user who starts the Integration Server program (typically, hyperion).

   Note: Logging on as a different user, such as an administrative user, may fail to reveal problems in the configuration.

2. Start a database client utility program with which you can use SQL statements to connect to and query databases, such as a command-line SQL utility.

   If the database client utility does not start, check to make sure that the client software is installed and the required environment variables are set. See “Correcting Database Client Problems” on page 139.

3. Connect to the OLAP Metadata Catalog database in the data source using a valid database user account.

   If the connection fails, try to connect as a user who you know has access permission, such as an administrative user. If the administrative connection succeeds, check the other user accounts and permissions in the data source for the OLAP Metadata Catalog database, and carefully check user accounts with connection problems. See, “Correcting Data Source Problems” on page 138.

4. Execute an SQL select statement against a table in the OLAP Metadata Catalog database; for example, if you are connected to the sample OLAP Metadata Catalog, type

   ```sql
   SELECT * FROM TBC.OV_INFO
   ```

   If the select statement fails, check the permissions for the user name you are using to connect to the data source. See “Correcting Data Source Problems” on page 138. Also, check that the version of the database client software you are using is compatible with the version of the data source. See “Correcting Database Client Problems” on page 139.
After you have successfully completed the preceding steps, try connecting to OLAP Metadata Catalog from Integration Services Console.

See “Testing the Integration Services Console Connection to the Data Source” on page 134. If the connection fails, proceed to “Testing the ODBC Connection to the Data Source” on page 129.

If you still cannot complete a connection and select statement with the user who runs Integration Server, contact technical support for the data source.

Repeat Step 3 through Step 5 for the data source database.

After you have successfully connected to OLAP Metadata Catalog, you can rule out setup and environment problems on the Integration Server computer and concentrate on problems specific to the data source database, including required database server names, database names, user names, passwords, and access permissions.

Testing the ODBC Connection to the Data Source

After you have established that the database client software and data source are communicating properly (if applicable), the next step in isolating connection problems is to test the ODBC connection to the data source databases.

On Windows systems, use ODBC Administrator to test connections to the data source databases. On UNIX, manually inspect the ODBC configuration files using ivtestlib (for 32-bit) or ddtestlib (for 64-bit) provided in the appropriate ODBC directory under common. You can also use third-party utilities included with your data source.

On UNIX systems, the most likely causes for problems in the ODBC component are that the environment variables for ODBC are not set up or that the odbc.ini file is not configured properly.

ODBC tracing can also be used to track down connection problems. See “Using ODBC Tracing” on page 143.

Testing ODBC on Windows Systems

To test ODBC connections to the data source databases:

1. On the Windows desktop, select Start, then Settings, then Control Panel to open the Control Panel window.
2. In the Control Panel window, double-click the Administrative Tools icon, and then double-click the Data Sources (ODBC) icon.
3. In the ODBC Data Source Administrator dialog box, click the System DSN tab.
4. In the System Data Sources list, select the data source created for OLAP Metadata Catalog or the data source database, and click Configure.

A data source configuration dialog box is displayed. Depending on the data source and the driver that you are using, you may be asked to log on to the database immediately, or you may be asked to review all the settings before testing the connection. In either case, after
attempting to log on to the database, observe the resulting message boxes to determine if you connected to the data source successfully.

**Note:** To be accessible from other computers, ODBC data sources must be configured as System DSNs, not User DSNs.

5 If you cannot connect to the database, check the Integration Server name, database name, user name, and password information for accuracy, and make sure that you are using the correct ODBC driver for the data source.

See “Correcting ODBC Problems” on page 140.

6 After you have successfully completed the preceding steps, try connecting to the data source from Integration Services Console.

See “Testing the Integration Services Console Connection to the Data Source” on page 134.

If the connection from Integration Services Console fails, proceed to “Testing the Integration Server Connection to the Data Source” on page 132.

**Testing ODBC on UNIX Systems**

On UNIX systems, you must manually inspect the configuration files for ODBC or use an data source utility to perform thorough testing. These configuration files include the login scripts that set the environment variables for ODBC and the odbc.ini file. See the documentation for the data source or the ODBC driver.

➢ To inspect the login script files:

1 Log on to the computer that runs Integration Server as the user who starts the Integration Server software.

2 In the current home directory, find the main login script file (typically,.profile for Korn Shell and Bourne Shell users,.login for C Shell users) and open it in a text editor.

**Note:** The primary login script file may be named differently, depending on the UNIX operating system and the system administrator.

3 Check the main login script file for the inclusion of the Integration Services script (is.sh or is.csh).

If one of these scripts is included in the main login script, make sure that the inclusion of the script file name follows the syntax for executing a script file within another script, and that the script file is specified with a complete directory path. See “Correcting ODBC Problems” on page 140.

**Note:** If you make a change to the login scripts, log out and then log back on to reset the environment variables.

4 After completing the preceding steps, test the connection by following the steps listed in “Testing the Integration Services Console Connection to Essbase Server” on page 136.
If you cannot make a connection, proceed to “Testing the Integration Server Connection to the Data Source” on page 132.

To inspect the odbc.ini file:

1. Log on to the computer that runs Integration Server as the user who starts the Integration Server software.

2. To determine the location of the odbc.ini file, type

   ```
   echo $ODBCINI
   ```

   If this command does not display a file name and directory location, then you have not included the Integration Services script in the login script for the user. See “Correcting Integration Server Problems” on page 141.

3. Move to the directory specified by $ODBCINI and open the odbc.ini file with a text editor.

4. Ensure that you are using the exact file name and directory that you have set. For example, type

   ```
   vi $ODBCINI
   ```

   to display the file name and path of the $ODBCINI environment variable.

5. Check that the name of the data sources you are using with Integration Services are listed in the ODBC Data Sources section; for example:

   ```
   [ODBC Data Sources]TBC_MD=
   Essbase Integration Services sample catalog
   TBC=Essbase Integration Services sample database
   ```

6. For each data source, check that there is a section starting with the name listed in the ODBC Data Sources section, enclosed in brackets; for example:

   For Oracle:

   ```
   [myoracle]
   Driver=common/ODBC/Merant/6.1/Drivers/ARor825.so
   Description=my oracle source
   ServerName=mytnsServerName
   ```

   For IBM DB2:

   ```
   [TBC_MD]
   Driver=common/ODBC/Merant/6.1/Drivers/ARdb225.so
   Description=DB2 database for sample testing
   ```

7. Within the section for each data source, verify that an ODBC driver file is specified.

   a. Make sure that the driver file exists by exiting the text editor and attempting to get a directory listing for that file; for example:

      ```
      ls /home/db2inst1/sqllib/lib/db2.so
      ```

   b. If the file is listed, use ivtestlib (32-bit) or ddtestlib (64-bit) to check the actual name and location of the driver, then make the appropriate changes to the odbc.ini file.
Tip: Copy the directory and file name for the driver from the odbc.ini file and paste it into an ls command.

Within the section for each data source, verify that the database name, computer name, and other required information are specified.

See “Correcting ODBC Problems” on page 140.

Testing the Integration Server Connection to the Data Source

After verifying the connections between the ODBC, the database client, and the data source, the next step in isolating a connection problem is to connect to the data source databases from Integration Server. Test this connection by running Integration Services Shell on the same computer that runs Integration Server. See Chapter 8, “Using Integration Services Shell.”

Note: The following procedure uses the sample OLAP Metadata Catalog and sample database. For your tests, substitute the OLAP Metadata Catalog and data source information you are using.

This test uses the LOADMEMBER command to test connections. When the loadmember starts a load, it attempts to make connections in the following order:

1. Data source, using ODBC
2. OLAP Metadata Catalog, using ODBC
3. Essbase Server, using TCP/IP

In this test, you set up these connections one at a time and execute a LOADMEMBER command each time. By examining the errors each time, you can determine which connections are successful.

➢ To test the Integration Server connection to the data source:

1 Verify that Integration Server is running.

   If Integration Server is not running, start it. If Integration Server will not start, check that the Integration Services environment variables are set. See “Correcting Integration Server Problems” on page 141.

2 On the same computer that is running Integration Server, start Integration Services Shell at a command prompt by typing

   olapicmd

   If Integration Services Shell does not start, make sure that the software is installed and that the Integration Services environment variables are set. See “Correcting Integration Server Problems” on page 141.

3 At the Integration Services Shell command prompt, use the login command to connect to the Integration Server computer; for example, type

   login cypress
4 Check the release of Integration Services by typing

`version`

Make sure that the version of the product you are using is compatible with the ODBC drivers and the version of Essbase you are using. For information on component compatibility, see Oracle Hyperion Enterprise Performance Management System Installation Start Here.

5 Set the data source to the sample data source by typing

`setsource "DSN=TBC;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=password"`

The DSN parameter requires the ODBC data source name, which may be different from the database name in the data source.

6 Set OLAP Metadata Catalog to the sample OLAP Metadata Catalog by typing

`setcatalog "DSN=TBC_MD;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=password"`

The DSN parameter requires the ODBC data source name, which may differ from the OLAP Metadata Catalog database name in the data source.

7 Set the instance of Essbase Server to which you want to connect by typing

`settarget "DSN=sequoia;UID=TBC;PWD=password;"`

8 Test the connection to the data source database. Attempt to perform a member load by typing

`loadmember "APP=OLAP_TBC;DBN=Basic;OTL=TBC Metaoutline;"`

9 Check the status of the preceding command by typing

`status`

During a member load, Integration Services Shell connects first to the data source and then to OLAP Metadata Catalog. If the connection to the catalog is unsuccessful, the message “IS Error: Unable to log in to Catalog” is displayed. If the connection to the data source fails, the message “IS Error: Unable to login to Datasource” is displayed.

If the connection to the data source failed, verify the ODBC data source name, user name, and password. If the connection continues to fail, see “Correcting Integration Server Problems” on page 141 and “Correcting ODBC Problems” on page 140. Repeat Step 4 through Step 8 of this procedure to retest the connection. When you successfully connect to the data source, continue to the next step.

10 Test the connection to the OLAP Metadata Catalog database by attempting to perform a member load. Type:

`loadmember "APP=OLAP_TBC;DBN=Basic;OTL=TBC Metaoutline;"`

11 Check the status of the preceding command by typing

`status`

During a member load, Integration Services Shell connects to OLAP Metadata Catalog and then to the Essbase Server. If the connection to OLAP Metadata Catalog is successful, the message “IS Error: Unable to login to Essbase Server” is displayed. If the connection to OLAP Metadata Catalog fails, the message “IS Error: Unable to login to Catalog” is displayed.
If the connection to OLAP Metadata Catalog failed, verify the ODBC data source name, user
name, and password. If the connection continues to fail, see “Correcting Integration Server
Problems” on page 141 and “Correcting ODBC Problems” on page 140. Repeat Step 9 and
Step 10 of this procedure to retest the connection. When you successfully connect to OLAP
Metadata Catalog, continue to the next step.

12 Test the connection from Integration Services Console to the data source by proceeding to “Testing the
Integration Services Console Connection to the Data Source” on page 134.

Testing the Integration Services Console Connection to the
Data Source

After you have corrected problems with the components that enable Integration Services
Console to connect to the data source, attempt a connection from the console to an OLAP
Metadata Catalog and a database in the data source to prove the corrections are successful. To
isolate possible computer-to-computer connection problems, run Integration Services Console
on a different computer than the one running Integration Server.

To test the Integration Services Console connection to the data source:

1 Verify that Integration Server is running.

If Integration Server is not running, start it. If Integration Server will not start, check that
the Integration Services environment variables are set. See “Correcting Integration Server
Problems” on page 141.

2 On a computer other than the one running Integration Server, start Integration Services Console.

3 Connect to the Integration Server computer and OLAP Metadata Catalog; for example, cypress and
TBC_MD.

If the connection fails, verify that you have specified the correct ODBC data source name
for the OLAP Metadata Catalog database and the correct Integration Server computer name.
Also verify that you specified the correct user name and password for the OLAP Metadata
Catalog database on the data source. See “Correcting Integration Services Console
Problems” on page 142.

4 After connecting to OLAP Metadata Catalog, open an OLAP model and connect to the data source for
the OLAP model.

If the connection fails, verify that you have specified the correct ODBC data source name
for the data source and the correct Integration Server computer name. Also verify that you
specified the correct user name and password for the data source database on the data source.
See “Correcting Integration Services Console Problems” on page 142.

Testing the Integration Server Connection to Essbase Server

When isolating connection problems between Integration Server and Essbase Server, use
Integration Services Shell to establish a connection to a data source and OLAP Metadata Catalog,
and then attempt to load members into an Essbase database.
To test the Integration Server connection to an Essbase Server:

1. Verify that Integration Server is running.
   
   If Integration Server is not running, start it. If Integration Server will not start, check to ensure that the Integration Services environment variables are set. See “Correcting Integration Server Problems” on page 141.

2. Verify that the Essbase Server is running.

3. At a command prompt, start Integration Services Shell by typing
   
   olapicmd
   
   If Integration Services Shell does not start, make sure that the software is installed and that the Integration Services environment variables are set. See “Correcting Integration Server Problems” on page 141.

4. At the Integration Services Shell command prompt, use the login command to connect to Integration Server; for example, type
   
   login cypress
   
5. Set the data source to the sample data source by typing
   
   setsource "DSN=TBC;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=password"
   
   The DSN parameter requires the ODBC data source name, which may differ from the database name in the data source.

6. Set OLAP Metadata Catalog to the sample OLAP Metadata Catalog by typing
   
   setcatalog "DSN=TBC_MD;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=password"
   
   The DSN parameter requires the ODBC data source name, which may differ from the OLAP Metadata Catalog database name in the data source.

7. Set the instance of Essbase Server to which you want to connect by typing
   
   settarget "DSN=sequoia;UID=TBC;PWD=password;"
   
   Note: If you are using Integration Services Console on a client computer that is outside the firewall for your network, and you require access to an Integration Server and Essbase Server located inside the firewall for your network, be sure to use a name or an IP address for the Essbase Server that is accessible from both sides of the firewall.

8. Test the connection to the Essbase Server by attempting to perform a member load by typing
   
   loadmember "APP=OLAP_TBC;DBN=Basic;OTL=TBC Metaoutline;"

9. Check the status of the preceding command by typing
   
   status
   
   If the connection failed, the message IS Error: Unable to login to Essbase Server is displayed. Make sure that the Essbase Server is running. Verify the name of the metaoutline in Essbase. Also verify the name of the test application and database in Essbase. Repeat Step 7 and Step 8 of this procedure to retest the connection.
After you can connect successfully to the Essbase Server, proceed to “Testing the Integration Services Console Connection to Essbase Server” on page 136.

Testing the Integration Services Console Connection to Essbase Server

When isolating the connection problems between Integration Services Console and Essbase Server, connect to OLAP Metadata Catalog and a data source, and then attempt to load members into an Essbase database.

Note: The following procedure uses the sample application, including the sample database, OLAP Metadata Catalog, OLAP model, and metaoutline. For your tests, substitute the application and databases you are using.

To test the Integration Services Console connection to an Essbase Server:

1. Verify that Integration Server is running.
2. Verify that Essbase Server is running.
3. Start Integration Services Console.
4. From Integration Services Console, connect to the Integration Server computer and OLAP Metadata Catalog; for example, cypress and TBC_MD.
5. After connecting to OLAP Metadata Catalog, open a metaoutline and connect to the data source for the metaoutline; for example, TBC Metaoutline and TBC.
6. Start a member load by selecting Outline, then Member Load.
7. If prompted, provide the name of the Essbase Server, user name, and password.

If the Essbase Application and Database dialog box is displayed, the connection is successful. If the connection failed, verify that you have entered the correct computer name for the Essbase Server and the correct user name and password. See the Essbase documentation.

If the Essbase Server is located on a large network, you may need to provide a more specific server name address, such as “cypress.hyperion.com” rather than simply “cypress.”

Note: If you are using Integration Services Console on a client computer that is outside the firewall for your network, and you require access to an Integration Server and Essbase Server located inside the firewall for your network, be sure to use a name or an IP address for the Essbase Server that is accessible from both sides of the firewall.

Correcting Connection Problems

In correcting connection problems with Integration Services, consider both component-level and system-level problems. The following topics list possible problems and areas for investigation within the software components of an Integration Services system.
To correct system-level problems that can occur across software components, review the following issues:

- Are all user names and passwords being used consistently across all the components and computer user accounts? See “User Name and Password Problems” on page 137.

- Do all user accounts used in the system have the appropriate permissions and privileges?
  System administrators typically use highly privileged administrative accounts when setting up and testing a system. After the system is set up, users with fewer privileges may be unable to use the system successfully. Check permissions on the following components:
  - User account on the client computer
  - User account on the Integration Server computer
  - OLAP Metadata Catalog database
  - Data source database
  - User account for the Essbase Server

- Are the software components using adequate addressing to identify the computers to which they are connecting?
  Depending on the size and configuration of the computer network, it may be necessary to use a fully-qualified host address to connect to a particular computer. For example, instead of the host name “cypress,” the complete host name “cypress.mydivision.mycompany.com” may be required. Alternatively, you might need to use the IP address number for the server computer; for example, 127.0.0.1.

- Is the user who starts Integration Server (for example, hyperion) configured to run all the required software?
  The user (for example, hyperion) must have all required environment variables set and must have permissions to run all the following software components:
  - Integration Server
  - ODBC software
  - Database client software
  - Data source server (this may not be required)

**User Name and Password Problems**

Consistent use of user names and passwords is often a problem in installing, configuring, and using Integration Services, because of the numerous software components and computers that make up the system.

**Tip:** During installation, configuration, and testing, keep a list of all user names and passwords that you use to create or to modify the system, including the applications with which the user names and passwords are associated and the purpose of each application.
In a smaller, single-server implementation of Integration Services, where Integration Server, Essbase Server, and the data source are all run on a single computer, you must manage five user name and password sets and use them consistently:

- User account on the Integration Services client computer
- User account on the computer that runs all the server software
- OLAP Metadata Catalog database in the data source
- Data source database in the data source
- Essbase Server software

In a fully distributed, three-server implementation, where Integration Server, Essbase Server, and the data source are all run on separate computers, you must manage seven user name and password sets and use them consistently:

- User account on the Integration Services client computer
- User account on the computer that runs Integration Server
- User account on the computer that runs the data source
- OLAP Metadata Catalog database in the data source
- Data source database
- User account on the computer that runs Essbase Server
- Essbase Server software

Adding aliases or synonyms to the data source databases adds to the number of user name and password sets you must manage.

**Tip:** During installation and connection testing, use a single set of user names and passwords to avoid confusion. Add user aliases or synonyms after the system is configured and running.

### Correcting Data Source Problems

To correct problems connecting to the data source, investigate the following possible causes:

- Is the data source server computer and software running? Has the server locked up, stopped, or failed?

- Is the data source software installed? Are all components required for ODBC connections installed?

  Some data sources may require additional software components beyond the basic server software to enable ODBC connections. The database client software is usually required on the computer that connects to the data source server using ODBC and also may be required on the data source server computer.

- Do the OLAP Metadata Catalog and data source databases to which you are trying to connect exist? Have the tables for OLAP Metadata Catalog been created?
You can create the OLAP Metadata Catalog tables either automatically or by using the creation scripts provided with Integration Services. See Chapter 3, “Creating, Upgrading, and Deleting OLAP Metadata Catalogs”.

- What are the exact names of the OLAP Metadata Catalog and data source databases? Are the names case-sensitive?

- Does the user name that you are using to connect to the data source have adequate permissions to access the database? Does the user have read and write permissions?
  
  For the OLAP Metadata Catalog database, users need full read and write permissions. For the data source database, users need read-only permissions.

- If the data source server is on a computer by itself, is there a corresponding user account on the computer for the user names you are using to connect to the databases?

  To allow connections to the database server from other computers, you may need to set up user accounts on the data source server computer, separate from the database users you create.

- Does the data source require any addressing beyond the name of the database if you are using the database client software or ODBC to connect?

  Some data sources may have specific naming requirements for connecting to their databases, including different naming conventions for third-party applications (such as Integration Services). See the documentation for the data source.

## Correcting Database Client Problems

If you have problems connecting to the data source from the database client software, investigate the following possible causes:

- Is the database client software installed? Are all database client components required for the ODBC connections installed?

  To enable ODBC connections, some database client software may require additional software components beyond the basic server software. The database client software usually is required on the computer that uses ODBC to connect to the data source server and may also be required on the data source server computer.

- Is the database client software compatible with the data source?

  Earlier or later versions of the database client software may not work with the data source version that you are running. See your data source documentation.

- Do the database client software utilities, such as the command-line SQL interface, start? Do they start when you are logged on as the user who runs Integration Server?

  When you run the database client software, the user account must have the appropriate file access permissions and must have set the path and other environment variables. Verify that the user who runs Integration Server (typically, hyperion) is configured to run the database client software.

  On UNIX systems, a script file is usually provided by the data source vendor to set the required environment variables and should be included in the main user login script. If you
include one of these files in the main login script, make sure that you specify a complete directory path and that you follow the syntax for executing a script file within a script.

- Are the data source server names, database names, user names, and passwords specified correctly?

Make sure you are using the correct names, including the exact spelling, uppercase or lowercase letters, and any required data source-specific identification syntax. See “Correcting Data Source Problems” on page 138.

- Is the database client software configured correctly?

File access permissions, path, and other environment variables must be set for the user who runs the database client software. Additional configuration steps also may be required for specific database clients.

See the installation documentation for the database client software.

**IBM DB2:** You may be required to bind the database client software utilities to the IBM DB2 databases. See the IBM DB2 installation documentation.

**Oracle:** The database name for Oracle databases can be specified in a *tnsnames.ora* file, which must be configured by the database administrator. See the Oracle installation documentation.

In the *sqlhosts* file, you must specify a TCP protocol for your operating system as follows:

- AIX and HP-UX: onsoctcp
- Solaris: ontlitzcp

### Correcting ODBC Problems

To correct problems using ODBC to connect to the data source, investigate the following possible causes:

- Is the ODBC software installed? Are all required ODBC components installed?

  On Windows systems, make sure the ODBC core components and any required drivers are installed. On UNIX systems, the Integration Services setup program installs the required ODBC core components if you choose to install the DataDirect ODBC drivers.

- Is the ODBC driver compatible with the data source? Is the driver compatible with the operating system?

  Verify that the ODBC driver you are using is supported by Integration Services by referring to the [Oracle Hyperion Enterprise Performance Management System Certification Matrix](http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html).

- Are the data source server names, database names, user names, and passwords specified correctly in the ODBC data source?

  Make sure you are using the correct names, including the exact spelling, uppercase or lowercase letters, and any required data source-specific identification syntax. On Windows
systems, configure ODBC data sources by using ODBC Administrator. On UNIX systems, configure ODBC data sources by editing the odbc.ini file manually.

**IBM DB2:** On Windows systems, the IBM DB2 Client Configuration Assistant shows the ODBC DSN. See “Correcting Data Source Problems” on page 138.

- On UNIX systems, have you specified the full path and file name of the ODBC driver for the data source in the odbc.ini file? Does the ODBC driver actually exist in the specified directory? Can ivtestlib (32-bit) or ddtestlib (64-bit) load it?

  Try copying the driver path and file name from the odbc.ini file and pasting it into an ls command. If the file is not listed when you execute the command, check the accuracy of the driver path and file name in the odbc.ini file.

- Are any parameters missing in the ODBC data source configuration?

  On Windows systems, make sure you have filled in all required parameters for the data source in ODBC Administrator. On UNIX systems, the Integration Services installation provides a sample odbc.ini file, which may not include data source parameters required for the data source that you are using. For information on required data source-specific ODBC driver parameters, see the ODBC documentation provided with Integration Services. For examples of odbc.ini file configurations for the supported data source platforms, see the Chapter 2, “Configuring Data Sources.”.

- Can other applications connect to the data source by using ODBC?

  If you can identify another application that is successfully using ODBC to connect to databases on the data source, analyze the configuration of the computer from which the connection is made and use it as a model for the computer connections you are trying to fix.

  **Note:** Most data source vendors provide utilities to test ODBC connections. For information, see the documentation for the data source.

If the preceding checks do not enable you to correct the problems connecting from ODBC to the data source, try using ODBC tracing to isolate the problem. See “Using ODBC Tracing” on page 143.

### Correcting Integration Server Problems

To correct problems connecting to the data source from Integration Server, investigate the following possible causes:

- Is Integration Server installed? Are all the required Integration Server components installed?

  The following components must be installed to have a functional instance of Integration Server:

  - Integration Server software
Related base components (on UNIX, these components are included as part of Integration Server software)

DataDirect ODBC drivers (unless ODBC and drivers are already installed)

- Is Integration Server running? Is it configured properly?

Start Integration Server by opening a command prompt window and typing `olapisvr`.

If Integration Server does not start, review the following possible problems:

- Are the required environment variables set? Are they set for the correct user name?
  
  On Windows systems, if you decide not to enable the setup program to update environment variables automatically, you must update the variables manually.

  On UNIX systems, a script file (`is.sh` or `is.csh`) that sets environment variables must be included in the login script of the user who starts Integration Server.

- What release of Essbase is installed?

  If an incompatible version of Essbase is installed on the same computer as Integration Server and the \essbase\bin directory is in the current path, Integration Server does not start. For information on compatible versions of Essbase, see the Oracle Hyperion Enterprise Performance Management System Installation Start Here.

- Is port 3388 in use?

  Integration Server uses port 3388 to communicate with the Integration Services client software. If this port is being used by another program, Integration Server does not start.

- On UNIX, is more than one copy of `olapisvr` running? If so, all servers except one will hang indefinitely. To correct the problem, run the following command to display all programs that are currently running:

  ```
  ps -fe | grep olapisvr
  ```

Many of the problems related to Integration Server not working properly may be traced to configuration steps that you may have missed. See the Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide.

### Correcting Integration Services Console Problems

If you have problems connecting to the data source from Integration Services Console and have ruled out problems with the data source, database client, ODBC, and Integration Server, the problem likely originates in the Integration Services Console component.

To correct problems connecting to the data source from the console, investigate the following possible causes:

- Is Integration Services Console installed? Are all required components installed?

  The following components must be installed to have a functional Integration Services Console client:

  - Integration Services Console software
Related base components

- **Is the Integration Services Console configured properly?**
  Make sure the required environment variables are set. Run the Oracle Hyperion Enterprise Performance Management System Configurator as described in the *Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide*.

- **Are the Integration Server computer name, ODBC data source names, user names, and passwords specified correctly?**
  For the Integration Server computer name, verify that you are using the correct spelling and uppercase and lowercase letters for the server name. For OLAP Metadata Catalog and the data source, make sure you are using an ODBC data source name, which may be different from the data source database name.

  **Note:** The user names and passwords for the OLAP Metadata Catalog database may be different from the data source. Verify that you are using the correct login information for each database.

  See “Correcting Data Source Problems” on page 138.

- **Does the user account with which you are running Integration Services Console have permission to connect to the computer running Integration Server?**
  You must have a user account and appropriate access permissions on the server computer running Integration Server.

- **If you have Integration Server installed on more than one computer, have you specified the correct server name?**
  A particular Integration Server may not be set up to access the same OLAP Metadata Catalogs and data sources used by other servers.

- **Are you using adequate addressing to identify the Integration Server computer?**

  Depending on the size and configuration of the computer network, it may be necessary to use a fully-qualified host address to connect to a particular computer. For example, instead of the host name “cypress,” the complete host name “cypress.mydivision.mycompany.com” may be required. Alternatively, you might need to use the IP address number for the server computer; for example, 127.0.0.1.

### Using ODBC Tracing

If you do not have access to ODBC testing utilities or third-party applications that use ODBC, tracking down ODBC problems can be difficult. Using the tracing utility provided with ODBC can help identify and resolve connection problems.
Using ODBC Tracing on Windows Systems

On Windows systems, if you cannot connect to the data source or OLAP Metadata Catalog, use the tracing utility in ODBC Administrator to learn which ODBC call is failing.

To use the tracing utility in ODBC Administrator:

1. Start ODBC Administrator on the computer running Integration Server.
2. Click the Tracing tab, specify a log file for the tracing utility, then click Start Tracing Now.

   **Note:** Tracing quickly creates a large log file. Disable tracing when you are finished: On the Tracing tab, click Stop Tracing Now.

3. Start Integration Server and Integration Services Console.
4. From Integration Services Console, select Connections, then OLAP Metadata Catalog, then Connect, and try to connect to OLAP Metadata Catalog.
5. If you can connect to OLAP Metadata Catalog, open an OLAP model and try to connect to a data source.
6. If you cannot connect to OLAP Metadata Catalog (step 4) or to a data source (step 5), see the file that you specified as the log file in the root directory.

   If necessary, share the information in the log file with Oracle Technical Support to help solve the connectivity problem.

Using ODBC Tracing on UNIX Systems

On UNIX systems, if you cannot connect to the data source or OLAP Metadata Catalog, use the tracing to learn which ODBC call is failing. On UNIX systems, you must edit the `odbc.ini` file to enable and disable ODBC tracing.

To use ODBC tracing on UNIX systems:

1. On the computer running Integration Server, open the `odbc.ini` file by using a text editor such as vi.
2. Find the section starting with `[ODBC]`, as shown in the following example:

   ```
   [ODBC]
   Trace=0
   TraceFile=odbctrace.out
   TraceDll=/export/home/users/hyperion/is/odbclib/odbctrac.so
   InstallDir=/export/home/users/hyperion/is/odbclib
   ```

3. Set the Trace setting to 1 to enable ODBC tracing.

   **Note:** Tracing quickly creates a large log file. Disable tracing when you are finished by setting the Trace parameter to 0.

4. Start Integration Server and Integration Services Console.
5. From Integration Services Console, select Connections, then OLAP Metadata Catalog, then Connect, and try to connect to OLAP Metadata Catalog.
6 If you can connect to OLAP Metadata Catalog, open an OLAP model and try to connect to a data source.

7 If you cannot connect to OLAP Metadata Catalog step 5 or to a data source step 6, read the odbctrace.out file.

If necessary, share the information in the log file with Oracle Technical Support to help solve the connectivity problem.
This chapter describes the commands supported by Essbase Integration Services Shell, the command-line tool provided with Integration Services. The majority of these commands provide the same functionality available through Essbase Integration Services Console. You use Integration Services Shell commands and scripts to create Essbase outlines and load Essbase databases.

If you schedule member or data loads using Integration Services Console, Integration Services automatically creates an Integration Services Shell batch file and a corresponding .cbs file in the Batch directory. You can modify and reuse the .cbs file for future member and data loads.

### Starting Integration Services Shell

Before you start Integration Services Shell, make sure that the following programs are running. These programs can be on network server computers and do not need to be on your local computer to be available to Integration Services Shell:

- A database that contains the OLAP Metadata Catalog where the metadata is stored
- One or more data sources that you want to use to create OLAP models and metaoutlines
- Essbase Integration Server
  
  See "Starting Integration Server" on page 14.

- Essbase Server
  
  See the Oracle Hyperion Enterprise Performance Management System Installation and Configuration Guide.

To start Integration Services Shell, use one of the following methods:

- From the Windows Start menu, select All Programs, then Oracle EPM System, then Essbase, then Integration Services, then Shell.
Enter the executable name on the command line; for example, type
olapicmd

To set the TCP port number with which Essbase Integration Server communicates with its
clients to a port different from the default 3388, enter the executable name and -
Portnumber when you start Integration Services Shell; for example, type
olapicmd -P3390

Note: If Integration Services Shell does not start when olapicmd is executed from the
command line, the operating system path may not be updated correctly. See the Oracle
Hyperion Enterprise Performance Management System Installation and Configuration
Guide.

The Integration Services Shell window opens, as shown in Figure 5:

Figure 5 Essbase Integration Services Shell Window

In the Integration Services Shell window, enter commands at the command prompt. For
example, type the following commands:

- Type login machine_name to log on to a server running on the same machine.
- Type shutdown to stop the server.
- Type exit to stop Integration Services Shell.
- Type help to get a list of Integration Services Shell commands.

Integration Services Shell

Integration Services Shell is a command-line tool that enables you to access Integration Server
to perform operations on an Essbase outline and the data in an Essbase database.

With Integration Services Shell, you can perform server operations at the command line in either
batch or interactive mode.

- Batch mode. To automate routine server maintenance and diagnostic tasks, write a script
  or batch file and run it from the command line.
  Batch mode is convenient if you use a particular series of commands frequently or if the task
  requires many commands.
- Interactive mode. To perform tasks interactively, enter OLAP commands at the command
  line.
Interactive mode is convenient for short operations that require few commands, such as checking for information on the fly and error checking.

**Tip:** It can be difficult to enter commands correctly when using the command line. The easiest method is to use the Integration Services Console OLAP Metaoutline standard user interface to perform a data or member load, and then choose to create a script file. You can then edit the script file as needed. Run all the commands in a script file by redirecting the standard input to use your command file. See “Running OLAP Command Script Files” on page 151.

### SET and LOAD Commands

SET commands tell Integration Server which data sources are involved in a load operation. LOAD commands tell Integration Server to execute a load operation.

**Note:** To perform a successful member and data load, the SET commands must be executed before the LOAD commands.

SET commands provide Integration Server with the location of the source databases, Essbase Server, and OLAP Metadata Catalog. The commands can be issued in any order, but you must issue all three SET commands before executing a LOAD command.

Use SET commands as follows:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETSOURCE</td>
<td>Specifies the data source databases from which an Essbase outline is built</td>
</tr>
<tr>
<td>SETTARGET</td>
<td>Specifies the name of the Essbase Server computer on which an outline for an Essbase application and database is built</td>
</tr>
<tr>
<td>SETCATALOG</td>
<td>Specifies the OLAP Metadata Catalog that you created to store OLAP models and metaoutlines</td>
</tr>
<tr>
<td>SETLOCALE</td>
<td>Specifies the locale for OLAPICMD.</td>
</tr>
</tbody>
</table>

The STATUS command (see “STATUS” on page 154) does not check or validate the syntax for SET commands.

LOAD commands use the information from the SET commands to load members, data, or both.

Use the LOAD commands as follows:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOADMEMBER</td>
<td>Loads members into an Essbase outline. You must load members before you load data.</td>
</tr>
<tr>
<td>LOADDATA</td>
<td>Loads data into an Essbase database. You cannot load data without loading members.</td>
</tr>
<tr>
<td>LOADALL</td>
<td>Loads both members and data into an Essbase database.</td>
</tr>
</tbody>
</table>

LOAD commands for small outlines (fewer than 1000 members) take a few seconds to execute. Larger outlines (1000 or more members) take a few minutes to a few hours, depending on the
amount of data being loaded, the configuration of the metaoutline, and the computer system capabilities. After executing LOAD commands, wait a few seconds until the command prompt (->) is displayed and then use the STATUS command to check the success or failure of the command execution.

See “Performing Member Loads Using Integration Services Shell” on page 165 for a sample sequence of commands for loading members.

See “Loading Data Using Integration Services Shell” on page 166 for a sample sequence of commands for loading data.

**Informational Commands**

The STATUS and VERSION commands provide processing status and system release version information for you and can be used at any time in the sequence of command execution.

**Command Syntax and Execution**

Integration Services Shell commands are not case-sensitive, but named data sources, user IDs, and passwords might be case-sensitive, depending on the operating system and the data source you are using. Enter the syntax exactly as shown in this chapter for each command. Keyword parameters can appear in any order.

The commands require semicolons and parameters such as data source name (DSN), code page (CODEPAGE), user ID (UID), and password (PWD) to separate the command into statements that indicate what kind of information you are providing; for example:

```
SETSOURCE "DSN=database;CODEPAGE=English_UnitedStates.Latin1@Binary;
UID=Smith;PWD=password;"
```

You must separate the statements with a semicolon. The final semicolon is optional. Do not use semicolons as part of data source names, because Integration Services Shell recognizes semicolons as field separators. It is recommended that you use double quotation marks around the parameters, as shown in the preceding example.

In this guide, all optional syntax parameters are indicated by their enclosure in brackets [ ]; however, do not type the brackets when you enter the commands.

Press Enter to execute the commands.

**Note:** If you are unsure of a specific syntax, use Integration Services Console to schedule member or data loads, or to create load scripts. An Integration Services Shell batch file and a corresponding .cbs file are created automatically in the eis\server\Batch directory. You can reference, modify, and reuse the .cbs file for future member and data loads. In the Schedule Essbase Load dialog box, be sure to check the Save Script Only option instead of the Schedule option. See “Recording Member and Data Load Scripts” on page 151.
Recording Member and Data Load Scripts

If you are creating scripts to perform loads or other tasks, you may want a script generated by Integration Services to use as a model or you may want to check your script syntax for Open Database Connectivity (ODBC) data source connection details. Record a load script with Integration Services by performing a load and recording the load script to a script file.

To record a load script:
1. Start Integration Services Console and open a metaoutline.
2. Select Outline > and one of the following menu items:
   - Member Load
   - Data Load
   - Member and Data Load
3. In the Essbase Application and Database dialog box, type or select the appropriate application and database names.
4. Click Next to display the Schedule Essbase Load dialog box.
5. Save the load to a script file by clicking Only Save Load Script and then click Save Scripts.
6. In the Save Script As dialog box, type a name for the script that you want to save, and then click OK.
7. Click Finish to start the load or to complete scheduling of the load.

Running Integration Services Shell Script and Batch Files

If you use a series of commands frequently or you must enter many commands to complete a task, you can automate the task with a script or batch file. Both are text files.

- A script file (.cbs extension) contains Integration Services Shell commands. You can run a script file from the operating system command line or from within an operating system batch file.
- On Windows platforms, a batch file (.bat extension) is an operating system file that calls multiple Integration Services Shell scripts and is used to run multiple sessions of Integration Services Shell commands. You can run a batch file on the server from the operating system prompt.
- On UNIX, shell scripts are used in place of batch or script files.

When you run a script file, Integration Services Shell executes the commands in the order specified in the script until the program reaches the end of the file.

Running OLAP Command Script Files

Enter the following command at the command prompt to run a script file in Integration Services Shell:

olapicmd -f scriptFileName [ > logFileName]
Replace `scriptFileName` with the name of the script file you are using. Replace `logFileName` with the name of the file where you want the feedback from the script to be recorded.

For example, the following sample script file, `olap_tbc.cbs`, was created in a text editor. This script connects to Essbase from the Integration Server computer and generates outlines for a sample database. In the following example, the `status` command (see “STATUS” on page 154) is used to check the success or failure of each command. The `status` command returns the execution status in the Integration Services Shell window.

```
login labmachine1
status
settarget "DSN=labmachine1;UID=hyperion;PWD=password"
status
setcatalog "DSN=TBC_MD;CODEPAGE=
    English_UnitedStates.Latin1@Binary;UID=tbc;PWD=password"
status
setsource "DSN=TBC;CODEPAGE=
    English_UnitedStates.Latin1@Binary;UID=tbc;PWD=password"
status
loadall "OTL=TBC Metaoutline;APP=olaptbc;DBN=Tbf1;FLT_ID=1;
    CALC_SCRIPT=#DEFAULT#;"
status
loadall "OTL=TBC Metaoutline;APP=olaptbc;DBN=Tbf2;FLT_ID=2;
    OTL_CLEAR=Y;"
status
loadall "OTL=TBC Metaoutline;APP=olaptbc;DBN=Tbf3a;FLT_ID=3;
    OTL_CLEAR=N;CALC_SCRIPT=#DEFAULT#;"
status
loadall "OTL=TBC Metaoutline;APP=olaptbc;DBN=Tbf3ec;FLT_ID=3;
    OTL_CLEAR=N;ESSC_SCRIPT=mytest1"
status
exit
```

To execute the `olap_tbc.cbs` script file, type

```
olapicmd -folap_tbc.cbs
```

The following sample batch file, `olap_tbc.bat`, uses input from a script file named `olap_tbc.cbs` and saves the feedback in a file named `olap_tbc.log`:

```
olapicmd -folap_tbc.cbs > olap_tbc.log
```

## Integration Services Shell Commands

The following subtopics describe each command and provide the syntax for entering the command. Each subtopic includes an example command entry for your reference.

The following commands are described in the subtopics:

- “LOGIN” on page 153
- “STATUS” on page 154
- “VERSION” on page 154
- “SETLOCALE” on page 154
Unless otherwise noted, you must be logged in to Integration Server to execute these commands.

The load commands in this topic are shown with an abbreviated syntax, which includes the syntax required for incremental loads. To see the full syntax for standard or incremental loads, use the Integration Services Console to record a script for a load. See “Recording Member and Data Load Scripts” on page 151.

**LOGIN**

The LOGIN command connects you to the Integration Server computer. On a successful connection, the Integration Services Shell prompt changes to show the name of the Integration Server computer to which Integration Services Shell is connected. You can also use the LOGIN command to connect to an Integration Server computer that is running on a non-default port. Whether you are logging in to an Integration Server computer that is running on the default port or on a non-default port, this command returns an error if you already are connected to the Integration Server computer.

Logging in to Integration Server Running on the Default Port

Syntax

LOGIN IShost

Parameter Description

| IShost | TCP/IP name or number of the computer where Integration Server is running |

Example

LOGIN cypress

Logging in to Integration Server Running on a Non-default Port

Syntax

LOGIN IShost:non-default_port_number
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IHost</td>
<td>TCP/IP name or number of the computer where Integration Server is running</td>
</tr>
<tr>
<td>non-default_port_number</td>
<td>The nonstandard port number on which Integration Server listens</td>
</tr>
</tbody>
</table>

### Example

```
LOGIN cypress:3390
```

## STATUS

The STATUS command displays the processing status of the most recently issued command. Use it to check the success or failure of all commands. After issuing a command, wait a few seconds until the command prompt (->) is displayed, and then enter the STATUS command.

The STATUS command does not check or validate the syntax for SET commands.

### Syntax

```
STATUS
```

### Example

Typing STATUS after issuing the LOADMEMBER command returns the following message in the Integration Services Shell window:

```
LOADMEMBER executed successfully, elapsed time is '22' seconds.
```

## VERSION

The VERSION command returns the release number of the Integration Services software you are running.

### Syntax

```
VERSION
```

### Example

VERSION returns the release number for Integration Services software.

## SETLOCALE

The SETLOCALE command enables you to override the default locale for OLAPICMD. The default locale for OLAPICMD is UTF-8 (Unicode).

Although a script file saved from Integration Services Console does not have the UTF-8 signature, OLAPICMD nevertheless reads the script as UTF-8.

If you create a script file in Microsoft Notepad and save the script file as a UTF-8 file, you do not need to use the SETLOCALE command to select a locale. You can use the default locale. If you choose to select a locale, you must select one that is UTF-8. A native locale causes load operations to fail.
If you create a script file in Microsoft Notepad and save the script file as an ANSI file, you must select a native locale. A UTF-8 locale causes load operations to fail.

If you intend to enter Integration Services Shell commands manually (rather than executing them through a script) on a native computer, you must select the appropriate native locale using the SETLOCALE command. Failure to do so will cause load operations to fail.

Syntax

```
SETLOCALE <LANGUAGE_TERRITORY.CODEPAGE@SORT>
SETLOCALE .UTF8@default
```

Examples

```
SETLOCALE Japanese_Japan.MS932@Binary
SETLOCALE .UTF8@default
```

**SETSOURCE**

The SETSOURCE command enables you to identify one or more source databases for Integration Server to use with a load command.

Syntax

```
SETSOURCE "DSN=PrimaryODBCdatasource;CODEPAGE=Codepage;UID=username;PWD=password[;][DSN=SecondaryODBCdatasource;CODEPAGE=Codepage;UID=username;PWD=password] [;]...
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PrimaryODBC datasourcex</td>
<td>The primary ODBC data source name configured on the computer where Integration Server runs—case-sensitive</td>
</tr>
<tr>
<td>Codepage</td>
<td>The code page of the language you want to use during the current Integration Services Console session</td>
</tr>
<tr>
<td>username</td>
<td>The name for logging on to a primary or secondary ODBC data source—case-sensitive</td>
</tr>
<tr>
<td>password</td>
<td>The password for logging on to a primary or secondary ODBC data source—case-sensitive</td>
</tr>
<tr>
<td>SecondaryODBC datasourcex</td>
<td>The secondary ODBC data source name configured on the computer where Integration Server runs—case-sensitive</td>
</tr>
</tbody>
</table>

Example

```
SETSOURCE "DSN=TBC;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=Password;DSN=TBC2;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=Password;DSN=TBC3;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=Password"
```

**SETTARGET**

The SETTARGET command enables you to identify a target Essbase Server computer for Integration Server to use with a load command.

Syntax

```
Integration Services Shell Commands 155
```
SETTARGET DSN=EssbaseServicesdataservername;UID=username;PWD=password

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>EssbaseServices dataservername</em></td>
<td>The TCP/IP name or number of the computer where Essbase Server is running</td>
</tr>
<tr>
<td><em>username</em></td>
<td>The name for logging on to Essbase Server</td>
</tr>
<tr>
<td><em>password</em></td>
<td>The password for logging on to Essbase Server—case-sensitive</td>
</tr>
</tbody>
</table>

Example

SETTARGET "DSN=FIG;UID=TBC;PWD=Password"

**SETCATALOG**

The SETCATALOG command enables you to identify an OLAP Metadata Catalog database for Integration Services to use with a load command.

**Syntax**

```
SETCATALOG DSN=ODBCdatasource;CODEPAGE=Codepage;UID=username;PWD=password
```

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>ODBCdatasource</em></td>
<td>The ODBC data source name configured on the computer where Integration Server runs—case-sensitive</td>
</tr>
<tr>
<td><em>Codepage</em></td>
<td>The code page of the language you want to use during the current Integration Services Console session</td>
</tr>
<tr>
<td><em>username</em></td>
<td>The name for logging on to the ODBC data source—case-sensitive</td>
</tr>
<tr>
<td><em>password</em></td>
<td>The password for logging on to the ODBC data source—case-sensitive</td>
</tr>
</tbody>
</table>

Example

SETCATALOG "DSN=TBC_MD;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=Fred;PWD=Password"

**LOADMEMBER**

The LOADMEMBER command initiates a member load operation on Integration Server. The load environment must be set up properly before you invoke the LOADMEMBER command. To set up the load environment, issue the SET commands in any order. See “SET and LOAD Commands” on page 149.

To learn the status of the member load invoked by the LOADMEMBER command, wait until you see the command prompt (->), then use the STATUS command.

**Syntax**

```
LOADMEMBER "OTL=Metaoutline;APP=Essbase Application;
DBN=Essbase Database;[FLT_ID_MEMBER=Member_Load_Filter_ID;]
[DELETE_DB=Delete and Restore Database Y|N;]
[OTL_CLEAR=Delete All Members First Y|N;]
```

156  Using Integration Services Shell
[INCUPD=DimID-MbrID,DimID-MbrID,...DimID-MbrID;]
[INCUPD_MEMBER=Dynamic Restructuring Member Load Options;]
[ATTR_UPDATE_LEVEL=Attribute Update Level;]
[@@USERS=Username List;][ESSCMD_SCRIPT=Esscmd Script File;]
[UNICODE=Create Unicode Application]
[UPDATE_DTDATA_ONLY=Update Drill-Through Data Only Y/N]"

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Metaoutline</strong></td>
<td>The name of the metaoutline—case-sensitive. This metaoutline is used to extract data and create an Essbase outline.</td>
</tr>
<tr>
<td><strong>Essbase Application</strong></td>
<td>The name of the Essbase application on the target Essbase Server computer where the Essbase outline is created.</td>
</tr>
<tr>
<td><strong>Essbase Database</strong></td>
<td>The name of the Essbase database on the target Essbase Server computer. Integration Server applies this name to the Essbase outline.</td>
</tr>
<tr>
<td><strong>Member Load Filter ID</strong></td>
<td>Optional. You can specify a member load filter ID to use when loading members. If you do not specify a filter ID, the default name (*DEFAULT) is used. The default filter ID is 1; any additional filters are numbered sequentially, starting with the number 2. For information about creating filters, see the Integration Services Console Help.</td>
</tr>
<tr>
<td><strong>Delete and Restore Database</strong></td>
<td>Optional. The Delete and Restore Database parameter directs whether Integration Server should delete all members in the Essbase database before performing a member load. The default is N (No).</td>
</tr>
<tr>
<td><strong>Delete All Members First</strong></td>
<td>Optional. When the Delete All Members First parameter is selected (OTL_CLEAR=Y), Integration Server removes all members from the existing database and then re-creates the database by using the members in the metaoutline. This process is slower than creating or updating an Essbase outline without deleting the members; therefore, do not delete all members first unless it is necessary. You should delete all members if, for example, you know that some members have been removed from the OLAP metaoutline and you want to build an Essbase outline containing a smaller set of members. The default is N (No).</td>
</tr>
<tr>
<td><strong>Values</strong></td>
<td>This is the format: DELETE_DB=[Y</td>
</tr>
<tr>
<td><strong>OTL_CLEAR</strong></td>
<td>This is the format: OTL_CLEAR=[Y</td>
</tr>
</tbody>
</table>
**Parameters**

**DimID-MbrID, DimID-MbrID,..., DimID-MbrID**

**Description**

**Optional.** Incremental Update. The level at which dimensions are updated during an incremental member load. The format is:

`INCUPD=DimID-MbrID,DimID-MbrID,...,DimID-MbrID;[;]`

Values for incremental update are as follows:

- **DimID** represents the dimension ID of the dimension to which the incremental update applies.
- **MbrID** represents the member ID of the level zero member of the hierarchy to which the incremental update applies.

**Dynamic Restructuring Member Load Options**

**Optional.** Specify 1, 2, 3, or 4. Dynamic restructuring member load options set the parameters for restructuring the database during a member load. The option that you select affects any subsequent data loads.

Values are as follows:

1. **Preserve all data.** Select to preserve all existing data that applies to the changed outline when restructuring occurs. This is the default (`INCUPD_MEMBER=1`).
2. **Discard all data.** Select to clear all data from the database.
3. **Preserve level 0 data.** Select to preserve data only for level zero members.

This is the optimal restructure option if you change the source database and need to recalculate the data, and if all data required for the calculation is in level zero members.

Selecting this option deletes all upper-level blocks before restructuring. This reduces the disk space for restructuring and improves calculation time when the database is recalculated. The upper-level blocks are re-created when you calculate the database.

4. **Preserve input data.** Select to preserve only those blocks containing data that is loaded. Many applications contain data that is entered at parent levels. Selecting this option prevents deletion of any blocks that are created by data load, whether they are non-level zero or level zero (leaf member) blocks.

**Attribute Update Level**

**Optional.** Use with the Incremental Update parameter. The level at which attribute dimensions are updated when a member load is performed. Values are as follows:

0. **Delete all attribute dimensions after the member load is performed.**
1. **Do not update existing attribute dimensions while updating other types of dimensions (for example: standard, time, or measures dimensions) during a member load.**
2. **Update all existing attribute dimensions and add new attribute dimensions during a member load.**

**Username List**

**Optional.** Lists all user IDs that can access the Essbase application and database into which you are loading the members. Commas separate user IDs. For example, `@USERS="admin", "ljones", "ksmith"`.

**Esscmd Script File**

**Optional.** The name of an ESSCMD script. The script that you define instructs Essbase to perform specified actions on the Essbase database. These actions might include changing a member tag or loading data from a worksheet into user-defined members after you load the data from the data source. The ESSCMD script must be in the `eis/server/esscript` directory.

**Create Unicode Application**

**Optional.** You can specify that a Unicode application is created on Essbase Server. The default is that no Unicode application is created.

This parameter cannot be overwritten. For example, if a previous application exists and you are overwriting that application with the current member load, you cannot overwrite the original Unicode or non-Unicode setting.

This is the format:

```
UNICODE=[Y|N]
```

For example:

```
UNICODE=Y
```
**Parameters**  
**Description**

*Update Drill-Through Data Only*  
*Optional.* Update drill-through information or hybrid analysis information.

There is no separate option for hybrid analysis. Use this command for both drill-through information and hybrid analysis information.

This is the format:

```
UPDATE_DTDATA_ONLY= [Y|N]
```

For example:

```
UPDATE_DTDATA_ONLY=Y
```

**Example**

```
LOADMEMBER "OTL=ProductsAnalysis;APP=Products;
DBN=Analysis;FLT_ID_MEMBER=1;OTL_CLEAR=Y;
INCUPD=3-2,4-2,5-1,5-2,5-3;INCUPD_MEMBER=3;
@@USERS="TBC","CFO";UNICODE=Y;"
```

**LOADDATA**

The LOADDATA command initiates a data load operation on Integration Server. The load environment must be set up properly before you invoke this command. To set up the load environment, issue the SET commands in any order. See “SET and LOAD Commands” on page 149.

LOADDATA takes a few seconds to execute. After executing LOADDATA, use the STATUS command to learn the status of the command execution.

**Syntax**

```
LOADDATA "OTL=Metaoutline;APP=Essbase Application;
DBN=Essbase Database;[FLT_ID_DATA=Data Load Filter ID;]
[REPLACE_ZEROS=Replace Zeros with #MISSING;]
[INCUPD=DimID-MbrID,DimID-MbrID,...DimID-MbrID;]
[INCUPD_DATA=Dynamic Restructuring Data Load Options;]
[@@USERS=Username List;][CALC_SCRIPT=Calc Script Name;]
[ESSCMD_SCRIPT=Esscmd Script File;]
[FT_COLUMN=FactTable Column for Incremental Update;]"
```

**Parameters**  
**Description**

*Metaoutline*  
The name of the metaoutline. Integration Server uses the specified metaoutline to extract data from the data source to create an Essbase outline.

*Essbase Application*  
The name of the Essbase application on the target Essbase Server computer where the Essbase outline is created.

*Essbase Database*  
The name of the Essbase database on the target Essbase Server computer. Integration Server applies this name to the Essbase outline.

*Data Load Filter ID*  
*Optional.* You can specify a data load filter ID to use when loading data. If you do not specify a filter ID, the default filter (*DEFAULT) is used. The default filter ID is 1; any additional filters are numbered sequentially, starting with the number 2. For information about creating filters, see the Integration Services Console Help.
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replace Zeros with #MISSING</td>
<td>Optional. The Replace Zeros with #MISSING parameter replaces Essbase account dimension values of zeros with #MISSING. The default is N (No).</td>
</tr>
<tr>
<td></td>
<td>This is the format: REPLACE_ZEROS={Y</td>
</tr>
<tr>
<td></td>
<td>Here are the reasons that you may want to replace zeros with #MISSING:</td>
</tr>
<tr>
<td></td>
<td>● Fewer input blocks are created during the data load.</td>
</tr>
<tr>
<td></td>
<td>● The calculation time required in the Essbase database is decreased significantly.</td>
</tr>
<tr>
<td></td>
<td>● Because of differences in calculator behavior depending on whether a value is zero or missing, faulty data is not generated.</td>
</tr>
<tr>
<td>DimID-MbrID, DimID-MbrID,... DimID-MbrID</td>
<td>Optional. Incremental Update. The level at which dimensions are updated during an incremental data load. This is the format:</td>
</tr>
<tr>
<td></td>
<td>INCUPD=DimID-MbrID, DimID-MbrID, ...DimID-MbrID;[]</td>
</tr>
<tr>
<td></td>
<td>Values for incremental update are as follows:</td>
</tr>
<tr>
<td></td>
<td>● DimID represents the dimension ID of the dimension to which the incremental update applies.</td>
</tr>
<tr>
<td></td>
<td>● MbrID represents the member ID of the level zero member of the hierarchy to which the incremental update applies.</td>
</tr>
<tr>
<td>Dynamic Restructuring Data Load Options</td>
<td>Optional. Dynamic restructuring data load options determine how Essbase loads values from a data source to the database. Specify 1, 2, or 3.</td>
</tr>
<tr>
<td></td>
<td>Values are as follows:</td>
</tr>
<tr>
<td></td>
<td>1 Overwrite. Select to replace the values in the database with the values in the data source. This is the default (INCUPD_ DATA=1).</td>
</tr>
<tr>
<td></td>
<td>2 Add. Select to add values in the data source to the existing values in the database.</td>
</tr>
<tr>
<td></td>
<td>3 Subtract. Select to subtract the values in the data source from the existing values in the database.</td>
</tr>
<tr>
<td>Username List</td>
<td>Optional. Lists all user IDs that can access the Essbase application and database into which you are loading the members. Commas separate user IDs. For example, @USERS=“admin”,“ljones”,“ksmith”.</td>
</tr>
<tr>
<td>Calc Script Name</td>
<td>Optional. The name of the calculation script. The calculation script determines how Essbase calculates the data values in the database. If you do not select a calculation script, Integration Server does not perform a calculation.</td>
</tr>
<tr>
<td></td>
<td>Note: If the data storage model property of the metaoutline is aggregate storage, this parameter is not applicable.</td>
</tr>
<tr>
<td>Esscmd Script File</td>
<td>Optional. The name of an ESSCMD script. The script that you define instructs Essbase Server to perform specified actions on the Essbase database. These actions might include changing a member tag or loading data from a worksheet into user-defined members after you load the data from the data source. The ESSCMD script must be in the eis/server/esscript directory.</td>
</tr>
</tbody>
</table>
Parameters | Description
--- | ---
FactTable Column for Incremental Update | **Optional.** You can specify that a time-based incremental data load be performed based on the date the last data load was performed. The DateTimeColumn Name parameter is the name of the datetime column in the fact table that contains each record's create date or modified date.

You can perform time-based incremental loads only if the fact table on which a metaoutline is based has a datetime column that contains each record's creation date or modified date.

This is the format:

\[
\text{FT\_COLUMN=} \text{DataSourceName.\_TableName.\_ColumName}, \\
\]

For example:

\[
\text{FT\_COLUMN=} \text{TBC.Sales.Timestamp} ;
\]

Example

LOADDATA "OTL=ProductsAnalysis;APP=Products;DBN=Analysis;FLT_ID_DATA=2;REPLACE_ZEROS=Y;CALC_SCRIPT=#DEFAULT#;INCUPD=1-2,2-1,3-2,4-2,5-1,5-2,5-3;INCUPD_DATA=3;@@USERS="TBC","CFO";"

**LOADALL**

The LOADALL command initiates a load of both members and data on Integration Server. The load environment must be set up properly before you invoke this command. To set up the load environment, issue the SET commands in any order. See “SET and LOAD Commands” on page 149.

LOADALL takes a few seconds to execute. After executing LOADALL, use the STATUS command to learn the status of the command execution.

**Note:** If you want an ESSCMD script to execute between a member load and a data load, use the LOADMEMBER and LOADDATA commands instead of the LOADALL command.

**Syntax**

LOADALL "OTL=Metaoutline;APP=Essbase Application;DBN=Essbase Database; 
[FLT_ID_MEMBER=Member Load Filter ID;] 
[FLT_ID_DATA=Data Load Filter ID;] 
[DELETE_DB=Delete and Restore Database;] 
[REPLACE_ZEROS=Replace Zeros with #MISSING;] 
[OTL_CLEAR=Delete All Members First;] 
[INCUPD=DimID-MbrID,DimID-MbrID,...DimID-MbrID;] 
[INCUPD_MEMBER=Dynamic Restructuring Member Load Options;] 
[INCUPD_DATA=Dynamic Restructuring Data Load Options;] 
[ATTR_UPDATE_LEVEL=Attribute Update Level;] [@@USERS=Username List;] 
[CALC_SCRIPT=Calc Script Name;] [ESSCMD_SCRIPT=Esscmd Script File;] 
[UNICODE=Create Unicode Application;]"
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Metaoutline</strong></td>
<td>The name of the metaoutline. Integration Server uses the specified metaoutline to extract data from the data source to create an Essbase outline.</td>
</tr>
<tr>
<td><strong>Essbase Application</strong></td>
<td>The name of the Essbase application on the target Essbase Server computer where the Essbase outline is created.</td>
</tr>
<tr>
<td><strong>Essbase Database</strong></td>
<td>The name of the Essbase database on the target Essbase Server computer. Integration Server applies this name to the Essbase outline.</td>
</tr>
<tr>
<td><strong>Member Load Filter ID</strong></td>
<td><strong>Optional.</strong> You can specify a member load filter ID to use when loading members. If you do not specify a filter ID, the default name (<em>DEFAULT</em>) is used. The default filter ID is 1; any additional filters are numbered sequentially, starting with the number 2. For information about creating filters, see the Integration Services Console Help.</td>
</tr>
<tr>
<td><strong>Data Load Filter ID</strong></td>
<td><strong>Optional.</strong> You can specify a data load filter ID to use when loading data. If you do not specify a filter ID, the default filter (<em>DEFAULT</em>) is used. The default filter ID is 1; any additional filters are numbered sequentially, starting with the number 2. For information about creating filters, see the Integration Services Console Help.</td>
</tr>
<tr>
<td><strong>Delete and Restore Database</strong></td>
<td><strong>Optional.</strong> The Delete and Restore Database parameter directs whether Integration Server should delete all members in the Essbase database before performing a member load. The default is N (No). This is the format: DELETE_DB=[Y</td>
</tr>
</tbody>
</table>
| **Replace Zeros with #MISSING**  | **Optional.** The Replace Zeros with #MISSING parameter replaces Essbase account dimension values of zeros with #MISSING. The default is N (No). This is the format: REPLACE_ZEROS=[Y | N] Here are the reasons that you may want to replace zeros with #MISSING:  
  - Fewer input blocks are created during the data load.  
  - The calculation time required in the Essbase database is decreased significantly.  
  - Because of differences in calculator behavior depending on whether a value is zero or missing, faulty data is not generated. |
### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Delete All Members First**      | **Optional.** When the Delete All Members First parameter is selected (OTL_CLEAR=Y), Integration Server removes all members from the existing database and then re-creates the database by using the members in the metaoutline. This process is slower than creating or updating an Essbase outline without deleting the members; therefore, do not delete all members first unless it is necessary. You should delete all members if, for example, you know that some members have been removed from the OLAP metaoutline and you want to build an Essbase outline containing a smaller set of members. The default is N (No). This is the format: OTL_CLEAR=[Y | N] Values are as follows:  
Y Remove all members from the existing database and then re-create the database by using the members in the metaoutline. 
N Do not remove members from the existing database; only update the Essbase database (this is the default). |
| DimID-MbrID, DimID-MbrID, ...    | **Optional.** Incremental Update. The level at which dimensions are updated during an incremental member and data load. This is the format: INCUPD=DimID-MbrID, DimID-MbrID, ... DimID-MbrID; [] Values for incremental update are as follows:  
- DimID represents the dimension ID of the dimension to which the incremental update applies. 
- MbrID represents the member ID of the level zero member of the hierarchy to which the incremental update applies. |
| Dynamic Restructuring Member Load Options | **Optional.** Specify 1, 2, 3, or 4. Dynamic restructuring member load options set the parameters for restructuring the database during a member load. The option that you select affects any subsequent data loads. Values are as follows:  
1 Preserve all data. Select to preserve all existing data that applies to the changed outline when restructuring occurs. This is the default (INCUPD_MEMBER=1).  
2 Discard all data. Select to clear all data from the database.  
3 Preserve level 0 data. Select to preserve data only for level zero members. This is the optimal restructure option if you change the source database and need to recalculate the data, and if all data required for the calculation is in level zero members. Selecting this option deletes all upper-level blocks before restructuring. This reduces the disk space for restructuring and improves calculation time when the database is recalculated. The upper-level blocks are re-created when you calculate the database.  
4 Preserve input data. Select to preserve only those blocks containing data that is loaded. Many applications contain data that is entered at parent levels. Selecting this option prevents deletion of any blocks that are created by data load, whether they are non-level zero or level zero (leaf member) blocks. |
| Dynamic Restructuring Data Load Options | **Optional.** Dynamic restructuring data load options determine how Essbase loads values from a data source to the database. Specify 1, 2, or 3. This is the default (INCUPD_DATA=1). Values are as follows:  
1 Overwrite. Select to replace the values in the database with the values in the data source.  
2 Add. Select to add values in the data source to the existing values in the database.  
3 Subtract. Select to subtract the values in the data source from the existing values in the database. |
### Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Attribute Update Level</strong></td>
<td>Optional. Use with the Incremental Update parameter. The level at which attribute dimensions are updated when a member load is performed. Values are as follows:</td>
</tr>
<tr>
<td></td>
<td>0 Delete all attribute dimensions after the member load is performed.</td>
</tr>
<tr>
<td></td>
<td>1 Do not update existing attribute dimensions while updating other types of dimensions (for example: standard, time, or measures dimensions) during a member load.</td>
</tr>
<tr>
<td></td>
<td>2 Update all existing attribute dimensions and add new attribute dimensions during a member load.</td>
</tr>
<tr>
<td><strong>Essbase Users</strong></td>
<td>Optional. Lists all user IDs that can access the Essbase application and database into which you are loading members. Commas separate user IDs. For example, @@USERS=&quot;admin&quot;,&quot;ljones&quot;,&quot;ksmith&quot;.</td>
</tr>
<tr>
<td><strong>Calc Script Name</strong></td>
<td>Optional. The name of the calculation script. The calculation script determines how Essbase calculates the data values in the database. If you do not select a calculation script, Integration Server does not perform a calculation. If the data storage model property of the metaoutline is aggregate storage, this parameter is not applicable.</td>
</tr>
<tr>
<td><strong>Esscmd Script File</strong></td>
<td>Optional. The name of an ESSCMD script. The script that you define instructs Essbase Server to perform specified actions on the Essbase database. These actions might include changing a member tag or loading data from a worksheet into user-defined members after you load the data from the data source. The ESSCMD script must be in the <code>eis\server\esscript</code> directory.</td>
</tr>
<tr>
<td><strong>Create Unicode Application</strong></td>
<td>Optional. You can specify that a Unicode application is created on Essbase Server. The default is that no Unicode application is created.</td>
</tr>
<tr>
<td></td>
<td>This parameter cannot be overwritten. For example, if a previous application exists and you are overwriting that application with the current member load, you cannot overwrite the original Unicode or non-Unicode setting.</td>
</tr>
<tr>
<td></td>
<td>This is the format:</td>
</tr>
<tr>
<td></td>
<td>UNICODE=[Y</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>UNICODE=Y</td>
</tr>
</tbody>
</table>

### Example

```
LOADALL OTL=ProductsAnalysis;APP=Products;DBN=Analysis;FLT_ID_MEMBER=2;FLT_ID_DATA=2;REPLACE_ZEROS=Y;OTL_CLEAR=Y;CALC_SCRIPT=#DEFAULT#;INCUPD=1-2,2-1,3-2,4-2,5-1,5-2,5-3;INCUPD_MEMBER=3;INCUPD_DATA=3;@@USERS="TBC","CFO";UNICODE=Y;
```

---

### LOGOUT

The LOGOUT command logs you out of Integration Server. On a successful logout, the Integration Services Shell prompt changes to LOCAL.

**Syntax**

`LOGOUT`

**Example**

LOGOUT logs you out of Integration Server.

**Note:** If you log out of Integration Server, you no longer have an active session. To execute another command, you must log on to Integration Server again.
**SHUTDOWN**

The SHUTDOWN command shuts down Integration Server. You must be logged on to Integration Server before you can use this command.

**Syntax**

SHUTDOWN

**Example**

SHUTDOWN shuts down Integration Server.

**EXIT**

The EXIT command exits Integration Services Shell and closes the Integration Services Shell window. Before you exit from Integration Services Shell, Integration Server automatically executes a LOGOUT command.

**Syntax**

EXIT

**Example**

EXIT logs you out of Integration Server and closes the Integration Services Shell window.

**Performing Member Loads Using Integration Services Shell**

You can perform a member load by using Integration Services Shell instead of Integration Services Console.

You can manually schedule the batch file by using the AT service in Windows or the cron scheduling daemon on UNIX systems. If you are not sure how to schedule the batch file manually, see the operating system documentation.

1. Log on to the Integration Server computer by using the LOGIN command; for example
   
   ```
   LOGIN cypress
   ```

2. Connect to the external data source by using the SETSOURCE command; for example
   
   ```
   SETSOURCE DSN=TBC;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=password
   ```

3. Connect to the Essbase Server computer by using the SETTARGET command; for example
   
   ```
   SETTARGET DSN=sequoia;UID=sys;PWD=password
   ```

4. Connect to OLAP Metadata Catalog by using the SETCATALOG command; for example

   ```
   SETCATALOG DSN=sequoia;UID=sys;PWD=password
   ```
Start the member load by using the \texttt{LOADMEMBER} command; for example

\begin{verbatim}
LOADMEMBER "OTL=TBC Metaoutline;APP=OLAP_TBC;DBN=Basic;FLT_NAME=Filter1"
\end{verbatim}

\section*{Loading Data Using Integration Services Shell}

You can perform a data load by using Integration Services Shell instead of Integration Services Console.

You can manually schedule the batch file by enabling the Task Scheduler service on Windows or the cron scheduling daemon on UNIX systems. If you are not sure how to schedule a batch file manually, see the operating system documentation.

To load data with Integration Services Shell, follow the same steps for loading data that you use with Integration Services Console:

\begin{enumerate}
\item Log on to Integration Server by using the \texttt{LOGIN} command; for example
  \begin{verbatim}
  LOGIN FIG
  \end{verbatim}
\item Connect to the external data source by using the \texttt{SETSOURCE} command; for example
  \begin{verbatim}
  SETSOURCE DSN=TBC;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=password
  \end{verbatim}
\item Connect to the Essbase Server computer by using the \texttt{SETTARGET} command; for example
  \begin{verbatim}
  SETTARGET DSN=sequoia;UID=sys;PWD=password
  \end{verbatim}
\item Connect to OLAP Metadata Catalog by using the \texttt{SETCATALOG} command; for example
  \begin{verbatim}
  SETCATALOG DSN=TBC_MD;CODEPAGE=English_UnitedStates.Latin1@Binary;UID=TBC;PWD=password
  \end{verbatim}
\item Start the data load by using the \texttt{LOADDATA} command; for example
  \begin{verbatim}
  LOADDATA "OTL=TBC Metaoutline;APP=OLAP_TBC;DBN=Basic;FLT_NAME=Filter1"
  \end{verbatim}
\end{enumerate}
This chapter describes the rules for naming applications, databases, dimensions, members, and aliases in Essbase. For detailed information on creating Essbase applications and databases, see the Essbase product documentation.

**Naming Restrictions for Applications and Databases**

When naming applications and databases, follow these rules:

- Use no more than 8 bytes when naming non-Unicode-mode applications and databases; use no more than 30 characters when naming Unicode-mode applications and databases.
- Do not use spaces anywhere in the name.
- Do not use the following special characters anywhere in the name:
  
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>* asterisk</td>
<td>+ plus</td>
</tr>
<tr>
<td>\ back slash</td>
<td>? question mark</td>
</tr>
<tr>
<td>[ ] brackets</td>
<td>&quot; quotation marks</td>
</tr>
<tr>
<td>: colon</td>
<td>; semicolon</td>
</tr>
<tr>
<td>, comma</td>
<td>' apostrophe</td>
</tr>
<tr>
<td>= equals</td>
<td>/ forward slash</td>
</tr>
<tr>
<td>&gt; greater than</td>
<td>tabs</td>
</tr>
<tr>
<td>&lt; less than</td>
<td></td>
</tr>
<tr>
<td>. period</td>
<td></td>
</tr>
</tbody>
</table>

- For aggregate storage databases only, do not use any of the following words as application or database names:
Enter the name in the case in which you want the word displayed. The application or database name is created exactly as you enter it. If you enter the name as all capital letters (for instance, NEWAPP), Essbase does not automatically convert it to upper- and lowercase (for instance, Newapp).

**Naming Restrictions for Dimensions, Members, and Aliases**

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

- Use no more than 80 characters when naming dimensions, members, or aliases.
- Names are not case-sensitive unless case-sensitivity is enabled. See “Setting Outline Properties” in *Essbase Administration Services Online Help*.
- Do not use quotation marks (""), brackets ([ ]), or tabs anywhere in a name.

  **Note:** Brackets ([ ]) are supported in block storage outlines, but are not recommended because their use causes errors when converting to an aggregate storage outline.

- At the beginning of a dimension or member name, do not use the following characters:

<table>
<thead>
<tr>
<th>Character</th>
<th>Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>@ at</td>
<td>() parentheses</td>
</tr>
<tr>
<td>\ backslash</td>
<td>. period</td>
</tr>
<tr>
<td>() braces</td>
<td>+ plus</td>
</tr>
<tr>
<td>, comma</td>
<td>' apostrophe</td>
</tr>
<tr>
<td>- dash, hyphen, or minus</td>
<td>_ underscore</td>
</tr>
<tr>
<td>= equals</td>
<td></td>
</tr>
<tr>
<td>&lt; less than</td>
<td></td>
</tr>
</tbody>
</table>

- Do not place spaces at the beginning or end of a name. Essbase ignores spaces at the beginning or end of a name.

- Do not use the following words as dimension or member names:
  
  - Calculation script commands, operators, and keywords. For a list of commands, see the *Essbase Technical Reference*.
  
  - Report writer commands. For a list of commands, see the *Essbase Technical Reference*. 

168  Naming Restrictions for Essbase Applications, Databases, and Members
- Function names and function arguments. For a list of functions, see the *Essbase Technical Reference*.
- Names of other dimensions and members (unless the member is shared), and generation names, level names, and aliases in the database.
- Any of the following words:

<table>
<thead>
<tr>
<th>ALL</th>
<th>GENRANGE</th>
<th>OR</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>GROUP</td>
<td>PAREN</td>
</tr>
<tr>
<td>ASSIGN</td>
<td>GT</td>
<td>PARENPARAM</td>
</tr>
<tr>
<td>CALC</td>
<td>ID</td>
<td>PERCENT</td>
</tr>
<tr>
<td>CALCMBR</td>
<td>IDERROR</td>
<td>PLUS</td>
</tr>
<tr>
<td>COPYFORWARD</td>
<td>INTEGER</td>
<td>RELOP</td>
</tr>
<tr>
<td>CROSSDIM</td>
<td>LE</td>
<td>SET</td>
</tr>
<tr>
<td>CURMBRNAME</td>
<td>LEVELRANGE</td>
<td>SKIPBOTH</td>
</tr>
<tr>
<td>DIM</td>
<td>LOOPBLOCK</td>
<td>SKIPMISSING</td>
</tr>
<tr>
<td>DIMNAME</td>
<td>LOOPARMS</td>
<td>SKIPNONE</td>
</tr>
<tr>
<td>DIV</td>
<td>LT</td>
<td>SKIPZERO</td>
</tr>
<tr>
<td>DYNAMIC</td>
<td>MBR</td>
<td>TO</td>
</tr>
<tr>
<td>EMPTYPARM</td>
<td>MBRNAME</td>
<td>TOLOCALRATE</td>
</tr>
<tr>
<td>EQ</td>
<td>MBRONLY</td>
<td>TRAILMISSING</td>
</tr>
<tr>
<td>EQOP</td>
<td>MINUS</td>
<td>TRAILSUM</td>
</tr>
<tr>
<td>EXCEPT</td>
<td>MISSING</td>
<td>UMINUS</td>
</tr>
<tr>
<td>EXP</td>
<td>MUL</td>
<td>UPPER</td>
</tr>
<tr>
<td>EXPPERROR</td>
<td>MULOP</td>
<td>VARORXMBR</td>
</tr>
<tr>
<td>FLOAT</td>
<td>NE</td>
<td>XMBRONLY</td>
</tr>
</tbody>
</table>
| FUNCTION  | NON      | $$$UNIVERSE$$$
| GE        | NONINPUT | #MISSING |
| GEN       | NOT      | #MI |

**Note:** If you enable Dynamic Time Series members, do not use the associated generation names, including History, Year, Season, Period, Quarter, Month, Week, or Day.
Using Dimension and Member Names in Calculation Scripts, Report Scripts, Formulas, and Filters

In calculation scripts, report scripts, filter definitions, partition definitions, or formulas, you must enclose member names in quotation marks (" ") for block storage databases, and in brackets ([ ]) for aggregate storage databases, in the following situations:

- The name starts with one or more numerals (for example, 100).
- The name contains spaces or any of the following characters:

<table>
<thead>
<tr>
<th>Character</th>
<th>Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp; ampersand</td>
<td>&gt; greater than</td>
</tr>
<tr>
<td>* asterisk</td>
<td>&lt; less than</td>
</tr>
<tr>
<td>@ at</td>
<td>() parentheses</td>
</tr>
<tr>
<td>\ back slash</td>
<td>% percent</td>
</tr>
<tr>
<td>{ braces</td>
<td>. period</td>
</tr>
<tr>
<td>: colon</td>
<td>+ plus</td>
</tr>
<tr>
<td>, comma</td>
<td>; semicolon</td>
</tr>
<tr>
<td>- dash, hyphen, or minus</td>
<td>/ forward slash</td>
</tr>
<tr>
<td>= equals</td>
<td>! exclamation point ~ tilde</td>
</tr>
</tbody>
</table>

- In calculation scripts and formulas, you must enclose the following member names in quotation marks (" ") for block storage databases, and in brackets ([ ]) for aggregate storage databases:

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>BEGIN</td>
<td>MEMBER</td>
</tr>
<tr>
<td>DOUBLE</td>
<td>RANGE</td>
</tr>
<tr>
<td>END</td>
<td>STRING</td>
</tr>
<tr>
<td>MACRO</td>
<td>THEN</td>
</tr>
</tbody>
</table>
This appendix provides a partial list of the return codes and error messages generated by Essbase Integration Server and Essbase Server. Return codes and error messages are grouped into two categories: return codes and errors generated by Integration Server and errors generated by Essbase Server for member and data loads.

The complete lists of OLAPICMD return codes (unixrtcd.txt and winrtcd.txt) are located in eis/server/docs on UNIX and eis\server\docs on Windows.

The complete list of Integration Server error messages is located in eis/server/bin/error.txt on UNIX and eis\server\bin\error.txt on Windows.

For errors generated by Essbase for member and data loads, see “Essbase Error Messages Generated During Data Loads” on page 178.

**OLAPICMD Session Return Codes**

At the end of an Integration Services Shell (OLAPICMD) session, OLAPICMD may return a code generated by Integration Server. This return code may be a status code or may be an error code.

To interpret the meaning of a return code, refer to the unixrtcd.txt file or the winrtcd.txt file. The unixrtcd.txt file is located in the eis/server/docs directory on UNIX, and the winrtcd.txt file is located in the eis\server\docs directory on Windows.

- The unixrtcd.txt file lists the return codes for UNIX, the corresponding Windows return code, and the associated message text. Because of a limitation in the number of return code values available for use on UNIX, some code numbers are used more than once. When you look up a return code number and find that it is used multiple times, you need to examine the messages for all instances of that return code number. Then you need to determine which
message applies to your situation by understanding the context in which the problem occurred.

- The `winrtcd.txt` file lists the return codes for Windows and the associated message text. The return code values in Windows are unique.

Table 15 shows examples of return codes on UNIX that are used more than once. Italicized words represent variable names inserted in the message text. Refer to the return code value and make a note of it in case you need to contact Oracle Technical Support with a problem.

For the complete list of return codes, refer to the `unixrtcd.txt` (located in the `eis/server/docs` directory on UNIX) and the `winrtcd.txt` file (located in the `eis\server\docs` directory on Windows).

Table 15  
Examples of Multiple Instances of Integration Services Shell (OLAPICMD) Return Code Values for UNIX

<table>
<thead>
<tr>
<th>UNIX Return Code</th>
<th>Windows Return Code</th>
<th>Message Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>061</td>
<td>1195069</td>
<td>Failed to add Essbase member <code>member_name</code> (DUPLICATE, ignored).</td>
</tr>
<tr>
<td>061</td>
<td>1195325</td>
<td>There must be an active session to process this command.</td>
</tr>
<tr>
<td>061</td>
<td>2003005</td>
<td>Data source error. Could not locate the metadata attributes.</td>
</tr>
<tr>
<td>069</td>
<td>1195077</td>
<td>Failed to create a local context.</td>
</tr>
<tr>
<td>069</td>
<td>1195333</td>
<td>ODBC Error. Encountered unknown ODBC exception while opening database.</td>
</tr>
<tr>
<td>094</td>
<td>1195358</td>
<td>Failed to get user attributes.</td>
</tr>
<tr>
<td>094</td>
<td>2002014</td>
<td>Cube builder error. Client request error. Invalid number of parameters.</td>
</tr>
</tbody>
</table>

## Member Load Error Messages

Table 16 lists some of the error messages that Integration Server may generate during a member load. Italicized words represent variable names inserted in the error message. Refer to the error message number in the first column and make a note of it if you need to contact Technical Support with a problem.

Table 16  
Essbase Integration Server Member Load Error Messages

<table>
<thead>
<tr>
<th>Windows Message Number</th>
<th>Message Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1195049</td>
<td>Detected metadata join problem (<code>table.column &lt; - &gt; table.column</code>).</td>
</tr>
<tr>
<td>1195050</td>
<td>Encountered unknown ODBC exception while opening database. Aborting the command.</td>
</tr>
<tr>
<td>1195064</td>
<td>Failed to add database <code>Essbase database</code>.</td>
</tr>
<tr>
<td>1195065</td>
<td>Failed to add dimension <code>dimension</code>.</td>
</tr>
<tr>
<td>1195066</td>
<td>Failed to add Essbase member <code>member</code> (%d).</td>
</tr>
<tr>
<td>Windows Message Number</td>
<td>Message Content</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>1195067</td>
<td>Failed to add Essbase member member (child of shared parent).</td>
</tr>
<tr>
<td>1195068</td>
<td>Failed to add Essbase member member (DUPLICATE).</td>
</tr>
<tr>
<td>1195069</td>
<td>Failed to add Essbase member member (DUPLICATE, ignored).</td>
</tr>
<tr>
<td>1195070</td>
<td>Failed to add Essbase member member (no parent or sibling members found).</td>
</tr>
<tr>
<td>1195071</td>
<td>Failed to add Essbase member member, %d.</td>
</tr>
<tr>
<td>1195072</td>
<td>Failed to add Essbase member member; the member exists.</td>
</tr>
<tr>
<td>1195073</td>
<td>Failed to add formula formula.</td>
</tr>
<tr>
<td>1195074</td>
<td>Failed to add member member.</td>
</tr>
<tr>
<td>1195075</td>
<td>Failed to add metaooutline member member.</td>
</tr>
<tr>
<td>1195076</td>
<td>Failed to add user attribute user-defined attribute.</td>
</tr>
<tr>
<td>1195077</td>
<td>Failed to create a local context.</td>
</tr>
<tr>
<td>1195078</td>
<td>Failed to create a local outline.</td>
</tr>
<tr>
<td>1195079</td>
<td>Failed to create application Essbase application.</td>
</tr>
<tr>
<td>1195080</td>
<td>Failed to create dimension dimension.</td>
</tr>
<tr>
<td>1195083</td>
<td>Failed to get information for parent of recursive hierarchy member.</td>
</tr>
<tr>
<td>1195085</td>
<td>Failed to get metadata for dimension.member.</td>
</tr>
<tr>
<td>1195088</td>
<td>Failed to initialize Essbase API.</td>
</tr>
<tr>
<td>1195089</td>
<td>Failed to open outline Essbase outline.</td>
</tr>
<tr>
<td>1195093</td>
<td>Failed to restructure Essbase outline.</td>
</tr>
<tr>
<td>1195095</td>
<td>Failed to update Essbase server (NULL HEADER).</td>
</tr>
<tr>
<td>1195097</td>
<td>Filter metaooutline filter doesn’t exist for metaooutline metaooutline.</td>
</tr>
<tr>
<td>1195098</td>
<td>Filter expression length exceeded the limit of maximum bytes bytes.</td>
</tr>
<tr>
<td>1195115</td>
<td>Incremental update specification incremental update specification string is incorrect.</td>
</tr>
<tr>
<td>1195126</td>
<td>Invalid aggregation function aggregate function for table.column.</td>
</tr>
<tr>
<td>1195129</td>
<td>Invalid esscmd script name esscmd script.</td>
</tr>
<tr>
<td>1195130</td>
<td>Invalid parameter parameter.</td>
</tr>
<tr>
<td>1195132</td>
<td>Invalid parameter string.</td>
</tr>
<tr>
<td>1195142</td>
<td>Metadata error: property ID is not a valid property for member or dimension.</td>
</tr>
<tr>
<td>Windows Message Number</td>
<td>Message Content</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>1195143</td>
<td>Metaoutline validation error. <em>Recursive hierarchy member</em>, a recursive member cannot have aggregation filter.</td>
</tr>
<tr>
<td>1195144</td>
<td>Metaoutline validation error. Filter = <em>member filter</em>, is not associated with any member.</td>
</tr>
<tr>
<td>1195188</td>
<td>Unknown exception encountered while opening dimension <em>dimension</em>.</td>
</tr>
<tr>
<td>1195189</td>
<td>Unknown exception encountered while opening member <em>member</em>.</td>
</tr>
<tr>
<td>1195190</td>
<td>Unknown exception encountered while opening metaoutline <em>metaoutline</em>.</td>
</tr>
<tr>
<td>1195341</td>
<td>Failed to Login to Essbase Server.</td>
</tr>
<tr>
<td>1195342</td>
<td>Failed to Get Essbase Settings.</td>
</tr>
<tr>
<td>1195344</td>
<td>Failed to unload database.</td>
</tr>
<tr>
<td>1195345</td>
<td>Failed to delete database.</td>
</tr>
<tr>
<td>1195346</td>
<td>Failed to close outline.</td>
</tr>
<tr>
<td>1195347</td>
<td>Failed to read outline.</td>
</tr>
<tr>
<td>1195349</td>
<td>Failed to unlock Database <em>database</em>.</td>
</tr>
<tr>
<td>1195350</td>
<td>Failed to save outline.</td>
</tr>
<tr>
<td>1195351</td>
<td>Unable to process state for <em>Essbase command</em>.</td>
</tr>
<tr>
<td>1195352</td>
<td>Failed to get access permissions for application.</td>
</tr>
<tr>
<td>1195353</td>
<td>Failed to set access permissions for application.</td>
</tr>
<tr>
<td>1195354</td>
<td>Failed to get access permissions for database.</td>
</tr>
<tr>
<td>1195355</td>
<td>Failed to set access permissions for database.</td>
</tr>
<tr>
<td>1195356</td>
<td>No member found.</td>
</tr>
<tr>
<td>1195357</td>
<td>Essbase Error: <em>error definition</em>.</td>
</tr>
<tr>
<td>1195358</td>
<td>Failed to get user attributes.</td>
</tr>
<tr>
<td>1195360</td>
<td>Failed to add member to outline.</td>
</tr>
<tr>
<td>1195361</td>
<td>Failed to calculate the data.</td>
</tr>
<tr>
<td>1195362</td>
<td>Failed to get parent.</td>
</tr>
<tr>
<td>1195363</td>
<td>Failed to get member.</td>
</tr>
<tr>
<td>1195364</td>
<td>Failed to move member.</td>
</tr>
<tr>
<td>1195365</td>
<td>Failed to get shared member.</td>
</tr>
<tr>
<td>1195366</td>
<td>Failed to get member information.</td>
</tr>
</tbody>
</table>
### Data Load Error Messages

Table 17 lists some of the error messages that Integration Server may generate during a data load. Italicized words represent variable names inserted in the error message. Refer to the error message number in the first column and make a note of it if you need to contact Oracle Technical Support with a problem.

<table>
<thead>
<tr>
<th>Windows Message Number</th>
<th>Message Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1195034</td>
<td>Cell load exceptions encountered.</td>
</tr>
<tr>
<td>1195047</td>
<td>Data load exceptions encountered. ? , amount.</td>
</tr>
<tr>
<td>1195063</td>
<td>Failed to activate Essbase application.Essbase.database.</td>
</tr>
<tr>
<td>1195082</td>
<td>Failed to execute esscmd script esscmd script.</td>
</tr>
<tr>
<td>1195369</td>
<td>Data load completed with errors.</td>
</tr>
<tr>
<td>1195370</td>
<td>Data load terminated with errors.</td>
</tr>
<tr>
<td>1195388</td>
<td>Failed to get data load SQL.</td>
</tr>
<tr>
<td>1195390</td>
<td>Number of data load SQLs not equal to number of hierarchies.</td>
</tr>
</tbody>
</table>
Drill-Through Report Error Messages

Drill-through reports are created using the Integration Services Console OLAP Metaoutline standard user interface and are viewed using Essbase Spreadsheet Add-in for Excel or Lotus 123 or any other Hyperion drill-through client.

Table 18 lists some of the error messages that Integration Server may generate to Essbase users viewing drill-through reports. Italicized words represent variable names inserted in the error message. Refer to the error message number in the first column and make a note of it if you need to contact Oracle Technical Support with a problem.

Table 18  Essbase Integration Server Drill-Through Report Error Messages

<table>
<thead>
<tr>
<th>Windows Message Number</th>
<th>Message Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1195081</td>
<td>Failed to enable DTS Member <em>member</em>.</td>
</tr>
<tr>
<td>1195087</td>
<td>Failed to get the universal member handle.</td>
</tr>
<tr>
<td>1195125</td>
<td>Intersection element <em>member</em> does not exist in <em>table.column</em>.</td>
</tr>
<tr>
<td>1195131</td>
<td>Invalid parameter count for the drill-through request. Aborting...</td>
</tr>
<tr>
<td>1195153</td>
<td>Received NULL pointer in DTAtributes( ) for Report = <em>Drill-Through report</em>.</td>
</tr>
<tr>
<td>1195154</td>
<td>Received NULL pointer in GetDTData ( ) for Report = <em>Drill-Through report</em>.</td>
</tr>
<tr>
<td>1195155</td>
<td>Received NULL pointer in GetDTDomain ( ).</td>
</tr>
<tr>
<td>1195156</td>
<td>Received NULL pointer in GetDTReport ( ).</td>
</tr>
<tr>
<td>1195157</td>
<td>Received NULL pointer in SetDTReport ( ).</td>
</tr>
<tr>
<td>1195183</td>
<td>Too many Drill-Through reports defined (exceeded Essbase metadata size limit). Drill-Through is disabled for this outline.</td>
</tr>
<tr>
<td>1195186</td>
<td>Unknown exception in GetDTReport ( ).</td>
</tr>
<tr>
<td>1195359</td>
<td>Essbase Error: Invalid Drill-Through Metadata.</td>
</tr>
<tr>
<td>1195369</td>
<td>IS Error: Data load completed with errors.</td>
</tr>
<tr>
<td>1195370</td>
<td>IS Error: Data load terminated due to errors.</td>
</tr>
<tr>
<td>1195371</td>
<td>IS Error: Unable to log in to data source.</td>
</tr>
<tr>
<td>1195372</td>
<td>IS Error: Unable to log in to OLAP Metadata Catalog.</td>
</tr>
<tr>
<td>1195373</td>
<td>IS Error: Unable to log in to Essbase Server.</td>
</tr>
<tr>
<td>1195374</td>
<td>IS Error: Unable to read Metaoutline information.</td>
</tr>
<tr>
<td>1195375</td>
<td>IS Error: Data calculation failed.</td>
</tr>
<tr>
<td>1195376</td>
<td>IS Error: Esscmd script execution failed.</td>
</tr>
</tbody>
</table>
### Miscellaneous Error Messages

Table 19 lists some of the miscellaneous error messages that Integration Server may generate during OLAP model and metaoutline creation and during data and member loads. Italicized words represent variable names inserted in the error message. Refer to the error message number in the first column and make a note of it if you need to contact Oracle Technical Support with a problem.

#### Table 19  Essbase Integration Server Miscellaneous Error Messages

<table>
<thead>
<tr>
<th>Windows Message Number</th>
<th>Message Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1195004</td>
<td><em>Metaoutline</em> failed validation. Database measures not specified.</td>
</tr>
<tr>
<td>1195007</td>
<td><em>Add job</em> failed.</td>
</tr>
<tr>
<td>1195017</td>
<td>Syntax Syntax error at <em>character location</em> in the filter expression <em>filter</em>.</td>
</tr>
<tr>
<td>1195018</td>
<td><em>Member</em> cannot have aggregation filter. Only leaf members can have aggregation filters.</td>
</tr>
<tr>
<td>1195054</td>
<td>Error message unavailable for this error.</td>
</tr>
<tr>
<td>1195057</td>
<td>Essbase server on <em>Essbase server computer</em> is not enabled with Integration Services option. Please inform your system administrator.</td>
</tr>
<tr>
<td>1195084</td>
<td>Failed to get job info (OS error <em>Job-ID</em>).</td>
</tr>
<tr>
<td>1195086</td>
<td>Failed to get the ODBC message.</td>
</tr>
<tr>
<td>1195090</td>
<td>Failed to remove <em>temporary batch file</em> (OS error <em>operating system error</em>).</td>
</tr>
<tr>
<td>1195091</td>
<td>Failed to remove job (OS error <em>operating system error</em>).</td>
</tr>
<tr>
<td>1195092</td>
<td>Failed to remove old job (OS error <em>operating system error</em>).</td>
</tr>
<tr>
<td>1195094</td>
<td>Failed to schedule job (OS error <em>operating system error</em>).</td>
</tr>
<tr>
<td>1195096</td>
<td>Fatal error.</td>
</tr>
<tr>
<td>1195124</td>
<td>Internal system error. Please contact Technical Support with the error number 1999999.</td>
</tr>
<tr>
<td>1195127</td>
<td>Invalid command line option <em>OLAP Integration Server switch</em>.</td>
</tr>
<tr>
<td>1195128</td>
<td>Invalid datatype specification <em>data type</em>.</td>
</tr>
<tr>
<td>1195133</td>
<td>Invalid session identifier; please log in to Integration Services again.</td>
</tr>
<tr>
<td>1195159</td>
<td>Replace <em>Failed member transformation</em>.</td>
</tr>
<tr>
<td>1195185</td>
<td>Trying to access invalid memory. Contact Technical Support.</td>
</tr>
<tr>
<td>1195187</td>
<td>Unknown ODBC exception encountered while closing database.</td>
</tr>
<tr>
<td>1195206</td>
<td>Syntax Syntax error at <em>character location</em> in the filter expression <em>filter</em>.</td>
</tr>
<tr>
<td>1195207</td>
<td>Integration Services is already running on this machine.</td>
</tr>
<tr>
<td>Windows Message Number</td>
<td>Message Contents</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1195208</td>
<td>Failed to release locks.</td>
</tr>
<tr>
<td>1195333</td>
<td>ODBC Error: Encountered unknown ODBC exception while opening database.</td>
</tr>
<tr>
<td>1195334</td>
<td>ODBC Error: Encountered unknown ODBC exception while closing database.</td>
</tr>
<tr>
<td>1195336</td>
<td>ODBC Error: Failed to get ODBC Error message for error number.</td>
</tr>
<tr>
<td>1195338</td>
<td>IS Error: Error during parsing Load Parameters.</td>
</tr>
<tr>
<td>1195343</td>
<td>Essbase Error: Essbase Server on Essbase server computer is not enabled with Integration Services option.</td>
</tr>
<tr>
<td>1195344</td>
<td>Failed to unload database.</td>
</tr>
<tr>
<td>1195348</td>
<td>Failed to get information for Database database.</td>
</tr>
<tr>
<td>1195391</td>
<td>IS Error: Unable to get tables for pattern name SQL regular expression pattern from data source.</td>
</tr>
<tr>
<td>1195392</td>
<td>IS Error: Unable to get columns for table data source table from data source.</td>
</tr>
<tr>
<td>1195393</td>
<td>IS Error: Unable to extract foreign keys for tables data source table, data source table from data source.</td>
</tr>
<tr>
<td>1195394</td>
<td>IS Error: Failed to bind columns.</td>
</tr>
<tr>
<td>1195395</td>
<td>IS Error: Failed to fetch next row.</td>
</tr>
<tr>
<td>1195396</td>
<td>IS Error: Failed to get number of columns.</td>
</tr>
<tr>
<td>1195397</td>
<td>IS Error: Failed to get column attributes.</td>
</tr>
<tr>
<td>1195398</td>
<td>IS Error: Failed to get statement handle.</td>
</tr>
<tr>
<td>1195399</td>
<td>IS Error: Failed to release statement handle.</td>
</tr>
<tr>
<td>1195400</td>
<td>IS Error: Failed to get number of rows.</td>
</tr>
<tr>
<td>1195401</td>
<td>IS Error: Unable to delete the model model name.</td>
</tr>
</tbody>
</table>

**Essbase Error Messages Generated During Data Loads**

Table 20 lists the most common errors that you will find in the `dataload.txt` file. The `dataload.txt` file for a specific data load is located in a subfolder in the `eis\server\loadinfo` directory in Windows and `eis/server/loadinfo` in UNIX. In the `loadinfo` directory, each data load generates a subfolder named in the following format:

`<application_database_timestamp_sessionnumber>`

Locate the subfolder for the data load that you want to review and open the `dataload.txt` file to view the errors.
<table>
<thead>
<tr>
<th>Message Number</th>
<th>Message Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>3303</td>
<td>Member not found in database.</td>
</tr>
<tr>
<td>3304</td>
<td>Insufficient access to store data.</td>
</tr>
<tr>
<td>3333</td>
<td>Bad data value supplied.</td>
</tr>
<tr>
<td>3335</td>
<td>Record rejected because of duplicate member names.</td>
</tr>
<tr>
<td>3336</td>
<td>Member/Data unknown.</td>
</tr>
<tr>
<td>3337</td>
<td>Record rejected because of dimension conflicts with Header Name.</td>
</tr>
</tbody>
</table>
This appendix describes Integration Services artifacts and database naming limits; Hybrid Analysis, Drill-through, and Unicode guidelines.

You should be familiar with these limits and guidelines before you begin using Integration Services.

For restrictions and guidelines on naming artifacts, see Chapter 9, “Naming Restrictions for Essbase Applications, Databases, and Members.” See also Integration Services Console online help.

**Artifact Limits**

Table 21 contains a list of limits that you may encounter when creating or manipulating Integration Services artifacts.

<table>
<thead>
<tr>
<th>Artifact</th>
<th>Limit for Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Names and Related Fields</td>
<td></td>
</tr>
<tr>
<td>Alias name</td>
<td>Non-Unicode application limit: 80 bytes</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 80 characters</td>
</tr>
<tr>
<td>Alias table name</td>
<td>Non-Unicode application limit: 30 bytes</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 30 characters</td>
</tr>
<tr>
<td>Artifact</td>
<td>Limit for Applications</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Essbase Server name</td>
<td>Non-Unicode application limit: 29 bytes</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 50 characters</td>
</tr>
<tr>
<td>Application name</td>
<td>Non-Unicode application limit: 8 bytes</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 30 characters</td>
</tr>
<tr>
<td>Application description</td>
<td>Non-Unicode application limit: 79 bytes</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 80 characters</td>
</tr>
<tr>
<td>Custom-defined function name</td>
<td>Non-Unicode application limit: 127 bytes. MaxL and the API truncate characters after 127 bytes.</td>
</tr>
<tr>
<td>Custom-defined macro name</td>
<td>Unicode-mode application limit: 128 characters. MaxL and the API truncate characters after 128 characters.</td>
</tr>
<tr>
<td>Custom-defined function specification</td>
<td>In either case, no truncation on server. No error is displayed if truncation occurs.</td>
</tr>
<tr>
<td>Custom-defined macro specification</td>
<td></td>
</tr>
<tr>
<td>Custom-defined function and macro comment</td>
<td>Non-Unicode application limit: 255 bytes. After 255 bytes, characters are truncated by MaxL and API.</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 256 characters. MaxL and the API truncate characters after 256 characters.</td>
</tr>
<tr>
<td></td>
<td>In either case, no truncation on server. No error is displayed if truncation occurs.</td>
</tr>
<tr>
<td>Data source name</td>
<td>46 characters</td>
</tr>
<tr>
<td>Database name</td>
<td>Non-Unicode application limit: 8 bytes</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 30 characters</td>
</tr>
<tr>
<td>Database description</td>
<td>Non-Unicode application limit: 79 bytes</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 80 characters</td>
</tr>
<tr>
<td>Directory path</td>
<td>Non-Unicode application limit: 256 bytes</td>
</tr>
<tr>
<td>For example: eis/server/bin</td>
<td>Unicode-mode application limit: 1024 bytes</td>
</tr>
<tr>
<td>File names for calculation scripts, report</td>
<td>Non-Unicode application limit: 8 bytes</td>
</tr>
<tr>
<td>scripts, and rules files</td>
<td>Unicode-mode application limit: If included within a path, the smaller of the following two values:</td>
</tr>
<tr>
<td></td>
<td>● 1024 bytes</td>
</tr>
<tr>
<td></td>
<td>● The limit established by the operating system</td>
</tr>
<tr>
<td></td>
<td>If not included within a path, as in some MaxL statements, 1024 bytes.</td>
</tr>
<tr>
<td>Filter name</td>
<td>Non-Unicode application limit: 30 bytes</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 30 characters</td>
</tr>
<tr>
<td>Group name</td>
<td>Non-Unicode application limit: 30 bytes</td>
</tr>
<tr>
<td></td>
<td>Unicode-mode application limit: 30 characters</td>
</tr>
<tr>
<td>Linked reporting artifact cell note</td>
<td>599 bytes</td>
</tr>
<tr>
<td>Linked reporting artifact URL</td>
<td>512 characters (always single-byte characters)</td>
</tr>
<tr>
<td>Artifact</td>
<td>Limit for Applications</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Member comment field           | Non-Unicode application limit: 255 bytes  
|                                | Unicode-mode application limit: 256 characters                                        |
| Member comment field (extended)| 8192 bytes                                                                             |
| Member name                    | Non-Unicode application limit: 80 bytes                                                |
|                                | Unicode-mode application limit: 80 characters                                           |
| OLAP model name                | 80 characters                                                                           |
| OLAP metaoutline name          | 80 characters                                                                           |
| Password                       | Non-Unicode application limit: 100 bytes                                                |
|                                | Unicode-mode application limit: 100 characters                                           |
| Substitution variable name     | 80 bytes                                                                                |
| Substitution variable value    | 255 bytes                                                                               |
| Trigger name                   | 30 bytes                                                                                |
| User-defined query             | No limitations                                                                         |
| User names                     | Non-Unicode application limit: 30 bytes                                                 |
|                                | Unicode-mode application limit: 30 characters                                           |
| Variable names                 | 32 bytes                                                                                |

**Data Load and Dimension Building Limits**

| Data load query                | No limitations                                                                         |
| Member load query              | No limitations                                                                         |
| Number of alias tables         | 9, including the Default alias table                                                   |
|                                | Number of error messages written to a data load or dimension build error log (DATAERRORLIMIT in essbase.cfg) |
|                                | Default 1000, minimum 1, maximum 65000                                                |
| Selection and rejection criteria | Number of characters that describe selection and rejection criteria: combination of all criteria limited to 32 KB |

**Other Limits**

| Caches: data, data file, index | 2 GB                                                                                    |
| Formula size                   | Created in Formula Editor: 64 KB. Formulas in calculation scripts are not subject to this limit.  
|                                | Created in MaxL, using multi-byte characters: 40 KB.  
<p>|                                | Formulas in calculation scripts are not subject to these limits.                        |</p>
<table>
<thead>
<tr>
<th>Artifact</th>
<th>Limit for Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of security filters</td>
<td>Per Essbase Server, 65535</td>
</tr>
<tr>
<td></td>
<td>Per Essbase database, 32290</td>
</tr>
<tr>
<td>Number of users</td>
<td>30,000. Errors can occur if you create more than 30,000 users.</td>
</tr>
<tr>
<td>Number of members in an Essbase outline</td>
<td>Approximately 1,000,000 explicitly defined in an Essbase outline for block storage</td>
</tr>
<tr>
<td></td>
<td>Approximately 20,000,000 explicitly defined in an Essbase outline for aggregate storage</td>
</tr>
<tr>
<td></td>
<td>Hybrid Analysis and some uses of partitions enable access to many more members than are</td>
</tr>
<tr>
<td></td>
<td>explicitly listed in an outline, the actual number of members accessible through the database</td>
</tr>
<tr>
<td></td>
<td>is much higher.</td>
</tr>
<tr>
<td></td>
<td>Longer names, which often occur if multi-byte characters are used, decrease the number of</td>
</tr>
<tr>
<td></td>
<td>members that are allowed.</td>
</tr>
</tbody>
</table>

**Source Database Artifact Naming Limits**

Integration Services does not support source database table names and column names containing the characters listed in Table 22.

*Note:* Table names may contain blank spaces. Column names may contain blank spaces and question marks.

*Note:* Table and column names may be greater than 30 characters in length.

**Table 22 Unsupported Characters in Table and Column Names**

<table>
<thead>
<tr>
<th>Character Description</th>
<th>Character Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;</td>
<td>quotation mark</td>
</tr>
<tr>
<td>&amp;</td>
<td>ampersand</td>
</tr>
<tr>
<td>,</td>
<td>comma</td>
</tr>
<tr>
<td>=</td>
<td>equal sign</td>
</tr>
<tr>
<td>@</td>
<td>at sign</td>
</tr>
<tr>
<td>#</td>
<td>pound sign</td>
</tr>
<tr>
<td>\t</td>
<td>Tab character</td>
</tr>
<tr>
<td>blank space</td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>asterisk</td>
</tr>
<tr>
<td>+</td>
<td>plus sign</td>
</tr>
<tr>
<td>-</td>
<td>dash, minus sign, or hyphen</td>
</tr>
<tr>
<td>( )</td>
<td>parenthesis</td>
</tr>
<tr>
<td>.</td>
<td>period</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>'</td>
<td>single quotation mark</td>
</tr>
<tr>
<td>\</td>
<td>backslash</td>
</tr>
<tr>
<td>/</td>
<td>forward slash</td>
</tr>
<tr>
<td>:</td>
<td>colon</td>
</tr>
<tr>
<td>&lt;</td>
<td>less than sign</td>
</tr>
</tbody>
</table>
Setting the Compression Dimension

In aggregate storage databases, the size of the compressed database changes depending on which dimension is specified as the compression dimension, and can affect retrieval performance.

When the data model property is set to aggregate storage, the Accounts dimension is specified as the compression dimension by default. However, you may determine that another dimension is the optimal choice for compression. Using the Compression Dimension check box, you can specify any single dimension as the compression dimension.

After performing a data load, use Administration Services Console or MaxL to view detailed compression and query statistics for your database.

Guidelines

- The compression dimension option is ignored if the outline is being built with Essbase versions earlier than 9.3. In versions less than 9.3, the Accounts dimension is tagged as the compression dimension internally.
- The following dimensions cannot be specified as the compression dimension:
  - Multiple hierarchy dimensions
  - Attribute dimensions
  - A base dimension with an attribute association
- If you tag a compression dimension with an Outline Hierarchy Information option other than “Dynamic at Dimension Level,” that dimension will be tagged as a dynamic dimension when it is loaded into Essbase.
- The compression dimension option applies to metaoutlines specified for aggregate storage only. To facilitate switching between aggregate and block data storage models, the option is always available, whether the metaoutline is specified for block storage or aggregate storage. If the metaoutline is specified for block storage, this option is ignored.

Hybrid Analysis Guidelines

You should be familiar with the Hybrid Analysis guidelines covered in this section.

Data Source Guidelines

- A single Essbase database can be associated with only one hybrid analysis relational data source.
A hybrid analysis data source can consist of only one relational database.

Hybrid Analysis supports data that is stored using either block storage or aggregate storage.

Hybrid Analysis supports Unicode-enabled data sources.

**Dimensions Guidelines**

- Hybrid Analysis is not supported on accounts dimensions.
- If the time dimension contains hybrid analysis-enabled members, the time dimension does not support Dynamic Time Series.
- Hybrid Analysis is not supported on user-defined dimensions.
- In an outline that is hybrid analysis-enabled, you can perform operations and analyses on dimensions that have attributes attached to one or more levels. The attribute dimension should be fully loaded into Essbase.
- Only the first hierarchy of a dimension with alternate hierarchies can have members enabled for hybrid analysis on its lowest levels.
- When building a dimension that is enabled for hybrid analysis, you must ensure that the column in the data source table that contributes to the leaf level of the Essbase portion of the dimension is non-nullable.

**Members Guidelines**

- Only the lowest level members of a dimension can be enabled for hybrid analysis.
- You should not rename a hybrid analysis-enabled member. If you rename a member, the member may not be retrieved the next time you perform a drill-through operation.
- Hybrid Analysis supports only parent-child prefixing on member names.
- Essbase does not support aliases for members that are enabled for hybrid analysis.
- Hybrid Analysis does not support scaling of measures dimension members using any of the operators + (addition), - (subtraction), * (multiplication), and / (division). If you use the scaling operators, drill-through queries into hybrid analysis data may show a mismatch between aggregated level-0 values in the Essbase database and the corresponding detail values in your data source.
- Essbase ignores all member properties, such as formulas, UDAs, and aliases for members that are enabled for hybrid analysis.
- You can associate an attribute member with a member enabled for hybrid analysis but you must make the association by metaoutline member name and not by level.
- Essbase supports drill-through operations defined on members that are enabled for Hybrid Analysis.
- You cannot apply properties of the Account Info tab to Hybrid Analysis members. Specifically, Time Balance, Skip, Variance Reporting, or Currency Conversion properties are not supported for Hybrid Analysis-enabled members.
Spreadsheet Guidelines

- Hybrid Analysis does not return numeric data in a spreadsheet if the member from the Accounts dimension is part of a ragged hierarchy.
- Hybrid Analysis is not supported with the Member Selection feature. You cannot select Hybrid Analysis members from the Member Selection dialog box.
- If you have multiple levels of hybrid analysis members in your outline, performing a zoom out operation on the bottom level hybrid analysis member takes you directly to the Essbase parent member, bypassing all other hybrid analysis levels.

Operations Guidelines

- Hybrid Analysis supports Dynamic Time Series.
- Essbase requires the OLAP Metadata Catalog created in Integration Services in order to drill down in a hybrid analysis data source.
- Hybrid Analysis does not support transparent, replicated, or linked partitions.
- Hybrid Analysis supports recursive hierarchies.

Limitations on Transformation Rules with Hybrid Analysis

Hybrid Analysis sets the following limitations on transformation rules:

- A database value cannot have a separator character that is the same as the one used for the prefix or suffix.
- A member name cannot be more than 80 characters (excluding blanks).
- A prefix or suffix must always have a separator associated with it.
- The data source database value cannot have trailing blanks.
- If spaces are converted to underscores during a transformation, then the Hybrid Analysis Manager assumes there are no underscores present in the database value.
- The use of all ancestors as a prefix or as a suffix is not supported.

Transformations Not Supported by Hybrid Analysis

Hybrid Analysis does not support the following transformations:

- Dropping spaces from around a member name
- Applying a prefix without a separator
- Applying names prefixes for all parents with or without a separator
- Applying a suffix without a separator
- Applying all suffixes of parent names with or without a separator
- Applying scaling to measures
Limitations on Using Formulas with Hybrid Analysis

Formulas used with hybrid analysis-enabled members are subject to the following limitations:

- Formulas are supported only on a measures dimension.
- Formulas cannot be attached to relational members.
- Formulas cannot reference a relational member by name.
- Member set functions (such as @CHILDREN and @DESCENDANTS), which generate member lists in a formula, execute only in the Essbase portion of the outline.

If a formula contains one or more functions that are not supported by Hybrid Analysis, Essbase returns the following error message:

```
Error executing formula for member

[member-name-to-which-formula-is-attached] (line [line# where the offending function appears inside the formula): function [Name of the offending function] cannot be used in Hybrid Analysis.
```

Unsupported Essbase Functions in Hybrid Analysis

Hybrid Analysis does not support all Essbase functions. The following topics specify the categories of significant Essbase functions not supported by Hybrid Analysis.

Relationship Functions

Hybrid Analysis does not support functions that look up specific values in the database based on current cell location and a series of parameters. Examples:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>@ANCEST</td>
<td>@SPARENT</td>
</tr>
<tr>
<td>@SANCEST</td>
<td>@CURLEV</td>
</tr>
<tr>
<td>@PARENT</td>
<td>@CURGEN</td>
</tr>
</tbody>
</table>

Member Condition Functions That Use Boolean Test

Hybrid Analysis does not support functions used to specify member conditions. Examples:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>@ISIANCEST</td>
<td>@ISLEV</td>
</tr>
<tr>
<td>@ISIPARENT</td>
<td>@ISSAMEGEN</td>
</tr>
<tr>
<td>@ISISIBLING</td>
<td>@SUDA</td>
</tr>
</tbody>
</table>
Range Functions

Hybrid Analysis does not support functions that take a range of members as arguments. Rather than return a single value, these functions calculate a series of values internally based on the range specified. Examples:

<table>
<thead>
<tr>
<th>@PRIOR</th>
<th>@MOVAVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>@SHIFT</td>
<td>@ALLOCATE</td>
</tr>
<tr>
<td>@PRIORS</td>
<td>@MDALLOCATE</td>
</tr>
<tr>
<td>@SHIFTS</td>
<td>@VAR</td>
</tr>
<tr>
<td>@NEXT</td>
<td>@VARPER</td>
</tr>
<tr>
<td>@MDSHIFT</td>
<td>@MEDIAN</td>
</tr>
<tr>
<td>@MOVSUM</td>
<td>@RANK</td>
</tr>
</tbody>
</table>

Attribute Functions

Hybrid Analysis does not support any Essbase functions that deal with attributes. Examples:

<table>
<thead>
<tr>
<th>@ATTRIBUTEVAL</th>
<th>@WITHATTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>@ATTRIBUTESVAL</td>
<td></td>
</tr>
</tbody>
</table>

Current Member and XREF

Hybrid Analysis does not support the following functions used to determine whether the current member is the member being specified:

<table>
<thead>
<tr>
<th>@CURRMBR</th>
<th>@XREF</th>
</tr>
</thead>
</table>

Using Advanced Relational Access

By default, when Integration Server creates an Essbase outline, it loads, or builds, all member levels specified in the metaoutline into a multidimensional database. You can, however, set Integration Server to build to a specified member level (Hybrid Analysis) or build only to the dimension level (Advanced Relational Access). Building down to a specified level produces a smaller Essbase outline and a smaller multidimensional database. Smaller databases can be useful for users with limited disk space who do not need to see the lowest level of detail.

Integration Services uses Advanced Relational Access to give Essbase end users direct access to data from relational databases or data warehouses. This feature enables users to perform online analytical processing (OLAP) on very large data sets.

In Integration Services Console, Advanced Relational Storage is enabled at the metaoutline level. When the Relational Storage option is selected, all members of all non-accounts dimensions are
automatically enabled for relational storage. Alternatively, you can enable relational storage on selected non-accounts dimensions.

When a metaoutline is enabled for Advanced Relational Access, end users are able to query directly on relationally-stored members. Essbase Server issues SQL queries to retrieve data from the database or data warehouse. All members of the dimension are accessed directly from the relational data source.

**Note:** For detailed information on enabling Advanced Relational Access, see Integration Services Console online help.

### Advanced Relational Access Guidelines

When users construct Advanced Relational Access queries, the points outlined in the following sections should be noted.

#### General Guidelines

Here are general guidelines to keep in mind when using Advanced Relational Access:

- Only outlines that are valid for aggregate storage can be Advanced Relational Access outlines. Block storage is not supported.
- Dynamic Time Series members are not supported.
- Advanced Relational Access requires MDX formulas.

  **Note:** An MDX query made against a ragged hierarchy in an Essbase cube returns data results that can differ significantly from the results obtained when the same MDX query is made against a ragged hierarchy in an Advanced Relational Access cube. See the section “Different Values Loaded in Ragged Hierarchies” on page 192.

- Members enabled for Advanced Relational Access are shown in Administration Services Console Outline Viewer but are not shown in the Outline Editor.
- A Time dimension from the Fact table is not supported. strongly recommends that you create a separate Time dimension table with time data only.
- When there are multiple measures defined in a metaoutline, consolidation at the highest level is based on the first measure only.

#### Data Source Guidelines

Here are guidelines to keep in mind when considering your data source for Advanced Relational Access:

- Advanced Relational Access cannot be enabled in metaoutlines that are connected to multiple data sources.
- Star and snowflake schemas are supported.
Teradata RDBMS provides different types of SQL date types, but Advanced Relational Access only supports the ‘DATE’ SQL date type.

**Dimension Guidelines**

Here are guidelines to keep in mind when working with dimensions in Advanced Relational Access:

- Advanced Relational Access does not support recursive dimensions/hierarchies.
- Alternate hierarchies in standard dimensions are not allowed.
- Attribute dimensions are ignored.
- User-defined dimensions are not supported.
- When a dimension is enabled for relational storage, Integration Server builds only to the dimension level. All members of the dimension are accessed directly from the relational data source.
- Accounts dimension must be created from the fact table.
- For the accounts dimension, no hierarchy is supported. All members should be of generation 2.
- Aliases are not supported.

**Member Guidelines**

Here are guidelines to keep in mind when working with members in Advanced Relational Access:

- The first member of the accounts dimension must be a base or stored measure. It cannot be a user-defined member.
- Aliases are not supported.
- Duplicate members are not supported.
- Shared members are not supported.
- Relational members are not automatically expanded when part of an outline is expanded. To view the relational members, you must manually click the member node containing the relational members.
- Relational members are not automatically collapsed when part of an outline is collapsed. To collapse the relational members, you must manually click the member node containing the relational members.
- In Integration Services Console, you can specify multiple sort order columns for each generation, selecting ascending or descending order.
- In Integration Services Console, you can associate a key column with each generation name and tag the key column as unique.
Unsupported Data Types

The following data types are not supported in Advanced Relational Access:

- IBM DB2: REAL
- Oracle: NCHAR and NVARCHAR2
- Microsoft SQL Server: TINYINT and SMALLINT

**Note:** The data types TINYINT and SMALLINT cannot be used as a measure; however, they can be used as members.

- Teradata: FLOAT

**Note:** Your member set column names should not be based on columns of FLOAT data types.

Essbase Databases and Advanced Relational Access

When you use Advanced Relational Access, your Essbase database has the following characteristics:

- The accounts dimension completely resides in Essbase.
- For all non-accounts dimensions, only the root members (the dimension level) reside in the Essbase database. All other non-accounts members are accessed directly from the relational database.

Different Values Loaded in Ragged Hierarchies

There is a significant difference in the way Advanced Relational Access and Essbase perform data loads in ragged hierarchies. Because of this difference, separate queries made against a ragged hierarchy can display different consolidation totals.

Essbase loads values in the upper-level members of a ragged hierarchy only when the level 0 members are non-null. Advanced Relational Access, however, loads values in the upper level members of the ragged hierarchy regardless of the value in the level 0 member.

Drill-through Report Guidelines

You should be familiar with these drill-through report guidelines:

- Drill-through operations cannot be performed on a Unicode Essbase database.
- Drill-through reports may be directed to an alternate Integration Server. Select the alternate server in the OLAP Metaoutline Properties dialog box in Integration Services Console.
- Drill-through operations can be performed with Dynamic Time Series.
- Drill-through operations can be performed on an alternate data source. An alternate data source is a source other than the primary or secondary data source used to create a
metaoutline. The alternate data source must contain the same data structure, including
column names and data types, as the primary or secondary data source originally used to
create the report. Drill-through operations can be performed on an alternate data source
that is in a second language.

- Integration Services is not capable of processing drill-through report requests from more
  than one instance of Essbase running on one server. In order for Integration Services to
  successfully process requests for drill-through reports from instances of Essbase running on
  non-standard Agent ports, an essbase.cfg file with the non-standard AGENTPORT setting
  must be present in eis/server/bin at the time Integration Services is started.

- If you create drill-through reports for a metaoutline based on multiple data sources, the SQL
  used for each drill-through report cannot include a join across a data source boundary; that
  is, each drill-through report can be based on only one data source. A metaoutline can contain
  multiple drill-through reports but all must be based on a single data source.

- Multi-cell drill-through is supported under the following conditions:
  - All members selected for multi-cell drill-through come from same physical table and
    column in the database.
  - All members selected for multi-cell drill-through come from the same level in the
    metaoutline.
  - The selected members cannot come from more than one hierarchy.

- The metaoutline in which you are designing a drill-through report cannot contain alternate
  recursive hierarchies. For example, you cannot insert the same member from a recursive
  hierarchy into a dimension more than once.

- You cannot use the $$ substitution variable in the template SQL (user-defined SQL) for a
  drill-through report when the intersection level of the dimension is defined at Generation
  1 and the dimension is built from a parent/child table.

- Oracle does not recommend using Drill-Through Template SQL for OLAP metaoutlines
  that support duplicate member names. Integration Server generates special tags internally
  to uniquely identify the duplicate members.

  If you still choose to use Drill-through Template SQL with OLAP metaoutlines that support
  duplicate member names, refer to the Essbase Integration Services Online Help for guidelines.

- Drill-through operations can be performed on an database containing multibyte character
  sets.

- There is a limit to the amount of information that can be stored in the Universal Member
  Comments (UMC) of an Essbase outline, meaning that users can typically store information
  for about 40 to 50 drill-through reports only. Because of this limitation, Oracle recommends
  that users design their drill-through reports so that the total number of reports is kept to
  under 40.

- In the following scenarios, drill-through operations may not work even after using the
  “Update Drill-through Data” command:
  - If you created an Essbase application and database using Integration Services, and then
    modified it using another tool, such as Administration Services.
If you have created an Essbase application and database using another tool, such as Oracle Essbase Administration Services.

Unicode Guidelines

- You must use the manual catalog creation procedures to create a Unicode-enabled OLAP Metadata Catalog, as described in Chapter 3, “Creating, Upgrading, and Deleting OLAP Metadata Catalogs”.
- Drill-through operations are not supported on a Unicode Essbase database.
- Integration Services does not support Unicode databases in Microsoft SQL Server RDBMS.
- If the database is UTF-8 but you are using only one language, select the UTF-8 option from the Code Page drop-down list whenever you access the database with the Login, OLAP Metadata Catalog Setup, Set Login Defaults, and Data Source dialog boxes.

Duplicate Member Names Guidelines

When a metaoutline is loaded into Essbase Server, an Essbase outline is usually created with each member name being unique. This is the default behavior of Integration Server. You can change this default and specify a metaoutline that, when loaded into an Essbase database, creates an Essbase outline containing duplicate (non-unique) member names, as shown in the following example:

US-->Massachusetts-->Springfield
US-->Missouri-->Springfield

Oracle Essbase Integration Services generates a unique internal identifier, which is applied to any duplicate member name in a metaoutline. This process enables support for duplicate member names during member loads, data loads, and drill-through operations.

When using duplicate member names, keep in mind the guidelines described in the sections that follow:

Dimensions in Duplicate Outlines

- Dimension names must be unique.
- In a dimension, a user-defined member cannot have the same name as a measure.

Members in Duplicate Outlines

- Duplicate member names are not allowed under the same parent. This is also true for user-defined members.
- A member cannot have children with duplicate names.
- Duplicate members may be shared or non-shared members.
• Duplicate members are supported in both aggregate storage and block storage.
• Duplicate member names are not supported if there are multiple data sources.
• Duplicate member names support duplicate aliases.

Other Guidelines for Duplicate Outlines

• Drill-through operations containing duplicate member names are supported.
• Oracle does not recommend using user-defined data load SQL or Drill-Through Template SQL for metaoutlines that support duplicate member names. Integration Server generates special tags internally to uniquely identify the duplicate members.

If you still choose to override data load SQL commands, exercise caution. You may have records rejected in error. It is especially recommended that you not change column names in SQL.

If you still choose to use Drill-through Template SQL with OLAP metaoutlines that support duplicate member names, refer to the Essbase Integration Services Online Help for guidelines.

• Duplicate outlines support MDX commands.

Note: For detailed information on creating an Oracle Essbase outline with duplicate member names, see Integration Services Console online help.

Limitations on 64-bit Platforms

The following features are not supported on 64-bit UNIX platforms or on the Windows 2008 64-bit platform:

• XML Import/Export
• Sample application creation (automatic or manual)
#MISSING See missing data.

**accounts dimension** A dimension type that makes accounting intelligence available. Only one dimension can be defined as Accounts.

**Add Joins mode** In Essbase Integration Services, a state in which you can draw lines to define joins between objects in the OLAP model.

**Advanced Relational Access** The integration of a relational database with an Essbase multidimensional database so that all data remains in the relational database and is mapped to summary-level data in the Essbase database.

**aggregate cell** A cell comprising several cells. For example, a data cell that uses Children(Year) expands to four cells containing Quarter 1, Quarter 2, Quarter 3, and Quarter 4 data.

**aggregate storage database** The database storage model designed to support large-scale, sparsely distributed data which is categorized into many, potentially large dimensions. Upper level members and formulas are dynamically calculated, and selected data values are aggregated and stored, typically with improvements in overall aggregation time.

**aggregate view** A collection of aggregate cells based on the levels of the members within each dimension. To reduce calculation time, values are pre-aggregated and stored as aggregate views. Retrievals start from aggregate view totals and add up from there.

**aggregation** The process of rolling up and storing values in an aggregate storage database; the stored result of the aggregation process.

**aggregation level** See consolidation level.

**alias column** In Essbase Integration Services, a column in the data source that contains the aliases for a member level in the metatemplate.

**alias table** A table that contains alternate names for members.

**alternate hierarchy** A hierarchy of shared members. An alternate hierarchy is based upon an existing hierarchy in a database outline, but has alternate levels in the dimension. An alternate hierarchy allows the same data to be seen from different points of view.

**alternate name** See alias.

**ancestor** A branch member that has members below it. For example, the members Qtr2 and 2006 are ancestors of the member April.

**application** 1) A software program designed to run a specific task or group of tasks such as a spreadsheet program or database management system; 2) A related set of dimensions and dimension members that are used to meet a specific set of analytical requirements, reporting requirements, or both.

**Architect** See Essbase Integration Services Console.

**attribute** A characteristic of a dimension member. For example, Employee dimension members may have attributes of Name, Age, or Address. Product dimension members can have several attributes, such as a size and flavor.

**base dimension** A standard dimension that is associated with one or more attribute dimensions. For example, assuming products have flavors, the Product dimension is the base dimension for the Flavors attribute dimension.

**block** The primary storage unit which is a multidimensional array representing the cells of all dense dimensions.
block storage database  The Essbase database storage model categorizing and storing data based on the sparsity of data values defined in sparse dimensions. Data values are stored in blocks, which exist only for sparse dimension members for which there are values.

branch A member of a hierarchy that may or may not contain leaf members.

Builder See Essbase Integration Services Console.

catalog See OLAP Metadata Catalog.

cell 1) The data value at the intersection of dimensions in a multidimensional database; the intersection of a row and a column in a worksheet; 2) A logical group of nodes belonging to one administrative domain.

child A member with a parent above it in the database outline.

code page A mapping of bit combinations to a set of text characters. Different code pages support different sets of characters. Each computer contains a code page setting for the character set requirements of the language of the computer user. In the context of this document, code pages map characters to bit combinations for non-Unicode encodings. See also encoding.

Command Interface See Essbase Integration Services Shell.

concatenation An operation that joins two characters or strings in the order specified, forming one string whose length is equal to the sum of the lengths of the two characters or strings. For example, the strings "New York " and "Library", when concatenated, become "New York Library".

condition In relational databases, a data extraction criterion. For example, you can apply a condition to extract only the data that begins with the letter A.

consolidation The process of aggregating data from dependent entities to parent entities. For example, if the dimension Year consists of the members Qtr1, Qtr2, Qtr3, and Qtr4, its consolidation is Year.

consolidation level The top of an aggregation hierarchy or any branch or sub-branch below the top, including the input (leaf) portion of the hierarchy.

currency conversion A process that converts currency values in a database from one currency into another. For example, to convert one U. S. dollar into the European euro, the exchange rate (for example, 0.923702) is multiplied by the dollar (1* 0.923702). After conversion, the European euro amount is .92.

data cleansing The process of making inconsistent data consistent. Examples of inconsistent data are data in which some values are incorrect or not of the correct data type.

data load rules A set of criteria that determines how to load data from a text-based file, a spreadsheet, or a relational data set into a database.

data value See cell.

database outline See outline.

DateTime transformation A set of instructions that defines how to change or reformat a relational database DateTime data type to your choice of date format.

denormalization The process of adding redundancy to data in a database, typically by joining tables to form more complete sets of data in the individual tables. This process is performed for the purpose of increasing data retrieval performance. Contrast with normalization.

dense dimension In block storage databases, a dimension likely to contain data for every combination of dimension members. For example, time dimensions are often dense because they can contain all combinations of all members. Contrast with sparse dimension.

descendant Any member below a parent in the database outline. In a dimension that includes years, quarters, and months, the members Qtr2 and April are descendants of the member Year.

detail member See leaf member.

dimension A data category used to organize business data for the retrieval and preservation of values. Dimensions usually contain hierarchies of related members grouped within them. For example, a Year dimension often includes members for each time period, such as quarters and months.

dimension branch A collection of dimension tables organized in a hierarchical structure, with one of the dimension tables joined directly to the fact table. A dimension branch defines a single, potential dimension in an Essbase Integration Services metaoutline.
dimension build rules  In Essbase, a set of operations similar to
data load rules. Instead of loading data, the dimension build
rules modify the outline based on data in the external data
source file.

dimension table  1) A table that includes numerous attributes
about a specific business process; 2) In Essbase Integration
Services, a container in the OLAP model for one or more
relational tables that define a potential dimension in
Essbase.

dimension type  A dimension property that enables the use of
predefined functionality. Dimensions tagged as time have a
predefined calendar functionality.

drill-through report  Direct access by Spreadsheet Add-in users
to data stored in the relational data source. Defined in
Essbase Integration Services, a drill-through report is based
on intersection levels (member sets) that Spreadsheet Add-
in users double-click to view detail information that is not
stored in the Essbase database.

duplicate member  The second occurrence of a member name
in a data source. Users can determine whether Essbase
Integration Server supports duplicate members, ignores
duplicate members, or adds them as shared members. See
also shared member.

duplicate member name  Multiple occurrences of a member
name in a database, with each occurrence representing a
different member. For example, a database has two
members named New York. One member represents New
York state and the other member represents New York city.

duplicate member outline  A database outline containing
duplicate member names.

Dynamic Calc and Store members  Members in a block storage
outline that Essbase calculates only upon the first retrieval
of the value. Essbase then stores the calculated value in the
database. Subsequent retrievals do not require calculating.

Dynamic Calc members  Members in a block storage outline
that Essbase calculates only at retrieval time. Essbase
discards calculated values after completing the retrieval
request.

dynamic calculation  In Essbase, a calculation that occurs only
when you retrieve data on a member that is tagged as
Dynamic Calc or Dynamic Calc and Store. The member's
values are calculated at retrieval time instead of being
precalculated during batch calculation.

Dynamic Time Series  A process that performs period-to-date
reporting in block storage databases.

encoding  A method for mapping bit combinations to
characters for creating, storing, and displaying text. Each
encoding has a name; for example, UTF-8. Within an
encoding, each character maps to a specific bit combination;
for example, in UTF-8, uppercase A maps to HEX41. See
also code page, locale.

Essbase Integration Server  The server component of the
Essbase Integration Services product family. Essbase
Integration Services uses the information stored in the
OLAP Metadata Catalog to extract the dimension names
and members names needed to build an Essbase outline
from the data source. When the Essbase outline is complete,
Integration Server extracts data from the data source,
performs the operations specified in the metaoutline, and
loads the data into the Essbase database.

Essbase Integration Services Console  The client component of
the Essbase Integration Services product family. This
graphic interface tool is used to create OLAP models and
metaoutlines, and to load data into an Essbase database.

Essbase Integration Services Shell  In Essbase Integration
Services, a command-line tool that you can use to perform
common operations on the Essbase outline and the data in
the Essbase database. For example, you can use the
ŁOADDATA command to load data.

Essbase OLAP Server  See Essbase Server.

Essbase outline  See outline.

Essbase Services database  A repository of data within Essbase
that contains a multidimensional data storage array. Each
database consists of a defined storage structure (a database
outline), data, security definitions, and other associated
files, such as calc scripts or data load rules. See also
application.

essbase.cfg  An optional configuration file for Essbase.
Administrators may edit this file to customize Essbase
Server functionality. Some configuration settings may also
be used with Essbase clients to override Essbase Server
settings.

ESSLANG  The Essbase environment variable that defines the
encoding used to interpret text characters. See also
encoding.
**fact table**  A container for one or more relational tables that define the data values for each dimension intersection in the OLAP model. For example, if the OLAP model contains Products, Region, and Year dimensions, the fact table might include data values for the number of units of Product A sold in New York in January.

**file delimiter**  A character, such as a comma or tab, that separates fields in a data source.

**filter**  A constraint on data sets that restricts values to specific criteria; for example, to exclude certain tables, metadata, or values, or to control access.

**foreign key**  In relational databases, a column whose data values correspond to the values of a key column in another relational table. See also key column, primary key.

**generation**  A layer in a hierarchical tree structure that defines member relationships in a database. Generations are ordered incrementally from the top member of the dimension (generation 1) down to the child members. Use the unique generation name to identify a layer in the hierarchical tree structure.

**Hybrid Analysis**  An analysis mapping low-level data stored in a relational database to summary-level data stored in Essbase, combining the mass scalability of relational systems with multidimensional data.

**Hyperion Integration Server Desktop**  See Essbase Integration Services Console.

**index**  1) In Essbase, a method of retrieving data based on sparse dimensions. Also refers to the index files, collectively; 2) In relational databases, pointers that are logically arranged by the values of a key. Indexes optimize access to relational data.

**index cache**  In Essbase, a buffer in memory that holds index pages.

**index entry**  In Essbase, 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**  In Essbase, a file used to store data retrieval information. It resides on disk and contains index pages.

**index page**  In Essbase, a subdivision of an index file containing entries that point to data blocks.

**input data**  Any data that is loaded from a data source and is not generated by calculation.

**integrity constraint**  In relational databases, a rule stating that each row should have an entry for each required key column.

**Intelligent Help**  In Essbase Integration Services, procedural help displayed in a dockable window that accompanies the OLAP Model and OLAP Metaoutline main windows. Intelligent Help provides numbered procedures and links to new automatic-detection options and frequently used functions.

**intersection level**  In Essbase Spreadsheet Add-in, an Essbase member combination that defines a specific value. For example, the member combination Actual, Root Beer, Sales, Jan, East represents the actual January sales value for root beer in the Eastern region.

**join**  A link between two relational database tables or topics based on common content in a column or row. A join typically occurs between identical or similar items within different tables or topics. For example, a record in the Customer table is joined to a record in the Orders table because the Customer ID value is the same in each table.

**join columns**  In Essbase Integration Services, two relational table columns that are joined from one table to another.

**key column**  In relational databases, a column or columns that form a unique identifier for each row. For example, EMPLOYEE_ID might be a key column.

**left frame**  (1) In the Essbase Integration Services Console OLAP Metaoutline main window, the area on the left that enables you to view a list of dimensions previously defined in the OLAP model. (2) In the OLAP Model main window, the area on the left that displays a list of the tables and views available in a source relational database.

**level**  A layer in a hierarchical tree structure that defines database member relationships. Levels are ordered from the bottom dimension member (level 0) up to the parent members.

**level 0 block**  A data block for combinations of sparse, level 0 members.

**level 0 member**  A member that has no children.
**load member**  In Essbase Integration Services, a member in a user-defined dimension into which data is loaded. Only user-defined dimensions require load members. For all non-user-defined dimensions, Essbase Integration Server knows how to load members and data into the Essbase database.

**load properties**  In Essbase Integration Services, a set of rules that determine what actions the product performs on member level names and data as they are loaded.

**locale**  A computer setting that specifies a location’s language, currency and date formatting, data sort order, and the character set encoding used on the computer. Essbase uses only the encoding portion. See also encoding, ESSLANG.

**logical column**  In Essbase Integration Services, a column created by manipulating the data in one or more physical columns. Contrast with physical column. See also column.

**logical table**  In relational databases, a table created by manipulating columns from one or more physical tables. The logical table is only a view of the data; the columns remain stored in the original tables and are not physically duplicated in the logical table. Contrast with physical table. See also view.

**mathematical operator**  A symbol that defines how data is calculated in formulas and outlines. Can be any of the standard mathematical or Boolean operators; for example, +, -, *, /, and %.

**MDDB**  See multidimensional database.

**measures**  Data values that a user wants to track, such as Unit_Price and Discount. By default, measures values map to the accounts dimension in the OLAP model, which maps to the measure dimension in the OLAP metaoutline, which in turn maps to the accounts dimension in the Essbase outline.

**measures dimension**  In Essbase Integration Services, a dimension that, by default, maps to the accounts dimension in the Essbase outline.

**members**  A discrete component within a dimension. A member identifies and differentiates the organization of similar units. For example, a time dimension might include members Jan, Feb, and Qtr1.

**member combination**  In Essbase, a list of member names used to specify a set of data at the intersection of two or more dimensions. A member combination is specified by using the cross-dimensional operator -> (a hyphen followed by a right-angle bracket). For example, the actual sales data for the month of January in Sample Basic is Sales->Jan->Actual.

**member level**  A hierarchical level of detail within a dimension. For example, in a dimension that defines geographic areas by nation, which are then subdivided into provinces, the nation and province categories each represent a member level. A member level corresponds to a level in an Essbase outline. The measures dimension contains actual members that are also member levels.

**member load**  In Essbase Integration Services, the process of adding dimensions and members (without data) to Essbase outlines.

**metadata**  A set of data that defines and describes the properties and attributes of the data stored in a database or used by an application. Examples of metadata are dimension names, member names, properties, time periods, and security.

**metaoutline**  In Essbase Integration Services, a template containing the structure and rules for creating an Essbase outline from an OLAP model.

**missing data (#MISSING)**  A marker indicating that data in the labeled location does not exist, contains no value, or was never entered or loaded. For example, missing data exists when an account contains data for a previous or future period but not for the current period.

**Move mode**  In Essbase Integration Services, a state in which you can pick up, move, and drop objects in the OLAP Model main window.

**multidimensional**  Describes a method of referencing data through three or more dimensions. An individual data value is the intersection of one member from each dimension.

**multidimensional database**  A method of organizing, storing, and referencing data through three or more dimensions. An individual value is the intersection point for a set of dimensions. Contrast with relational database.
**normalization** The process of grouping and removing redundancy from data so that each entity is in its appropriate place in the database and only in its appropriate place. Contrast with denormalization.

**numeric transformation** In Essbase Integration Services, a set of instructions that define how to change or reformat a relational database numeric field type. For example, you may choose to divide numeric data by 100.

**ODBC** Open Database Connectivity. A database access method used from any application regardless of how the database management system (DBMS) processes the information.

**OLAP Architect** See Essbase Integration Services Console.

**OLAP Builder** See Essbase Integration Services Console.

**OLAP Catalog** See OLAP Metadata Catalog.

**OLAP Command Interface** See Essbase Integration Services Shell.

**OLAP Integration Server** See Essbase Integration Server.

**OLAP Metadata Catalog** In Essbase Integration Services, a relational database containing metadata describing the nature, source, location, and type of data that is pulled from the relational data source.

**OLAP model** In Essbase Integration Services, a logical model (star schema) that is created from tables and columns in a relational database. The OLAP model is then used to generate the structure of a multidimensional database. See also online analytical processing (OLAP).

**OLTP** See online transaction processing.

**online analytical processing (OLAP)** A multidimensional, multiuser, client-server computing environment for users who analyze consolidated enterprise data in real time. OLAP systems feature drill-down, data pivoting, complex calculations, trend analysis, and modeling.

**online transaction processing (OLTP)** OLTP applications are commonly referred to as data capture, data entry, or data collection applications. OLTP applications enable an organization to capture the large amounts of data resulting from its daily activities but provide limited capability for reporting on the data.

**Open Database Connectivity (ODBC)** Standardized application programming interface (API) technology that allows applications to access multiple third-party databases.

**outline** The database structure of a multidimensional database, including all dimensions, members, tags, types, consolidations, and mathematical relationships. Data is stored in the database according to the structure defined in the outline.

**pass-through transformations** In Essbase Integration Services, a feature that allows you to use functions specific to your relational database management system (RDBMS) to extract data values for columns. You can provide a statement that Essbase Integration Server passes through to your RDBMS as a part of the SQL SELECT statement. You provide the statement as a property of a column and the pass-through feature returns a value for the column.

**permission** A special privilege that must be assigned to users or groups to enable them to access or modify secure data. Permissions include Read, Read/Write and None.

**physical column** A column that is stored in a relational database. See also column. Contrast with logical column.

**physical table** A combination of rows and columns stored in a relational database. Contrast with logical table.

**pointer** In relational databases, a data element indicating the location of data in storage.

**primary dimension table** A dimension table that joins directly to the fact table. Additional dimension tables may join to the primary dimension table to create a dimension branch.

**primary key** In relational databases, a column (key) that uniquely identifies a row. For example, Employee_ID.

**query governor** An Essbase Integration Server parameter or Essbase Server configuration setting that controls the duration and size of queries made to data sources.

**record** A set of information in a data source. Records are composed of fields, each of which contains one item of information. A set of records constitutes a table. A single record constitutes a row in the table. For example, a table containing personnel information might contain records (rows) that have three fields: a NAME field, an ADDRESS field, and a PHONE_NUMBER field.
recursive table  A relational source table that contains information in one row that is a parent or child of information in another row. For example, in a relational source table containing the columns EMPLOYEE_ID, NAME, and MANAGER_ID, the columns EMPLOYEE_ID and MANAGER_ID are recursive because MANAGER_ID refers back to the EMPLOYEE_ID. Using Essbase Integration Services, you can build an Essbase outline hierarchy from a recursive source table.

relational database  A type of database that stores data in related two-dimensional tables. Contrast with multidimensional database.

restructure  In Essbase, an operation to regenerate or rebuild the database index and, in some cases, the data files.

right frame  In the Essbase Integration Services Console OLAP Metaoutline main window, the area on the right, in which you build a metaoutline. In the OLAP Model main window, the area on the right, in which you build an OLAP model.

roll-up  See consolidation.

schema  In relational databases, a logical model that represents the data and the relationships between the data.

shared member  A member that shares storage space with another member of the same name, preventing duplicate calculation of members that occur multiple times in an Essbase outline.

Shared Services Registry  The part of the Shared Services repository that manages EPM System deployment information for most EPM System products, including installation directories, database settings, computer names, ports, servers, URLs, and dependent service data.

sibling  A child member at the same generation as another child member and having the same immediate parent. For example, the members Florida and New York are children of East and each other’s siblings.

sparse dimension  In block storage databases, a dimension unlikely to contain data for all member combinations when compared to other dimensions. Contrast with dense dimension. For example, not all customers have data for all products.

SQL  See Structured Query Language.

SQL Override  In Essbase Integration Services, a function that enables editing of the standard SQL statements generated by Integration Server for drill-through reports. ODBC SQL, Native SQL, and stored procedures can be used when editing the standard SQL. The edited, user-defined SQL can be selected to improve data load performance when loading data into an Essbase database.

staging area  A database that you create to meet the needs of a specific application. A staging area is a snapshot or restructured version of one or more RDBMS.

standard dimension  A dimension that is not an attribute dimension.

star schema  A logical model that represents your relational data in a form that mirrors that of OLAP data. A star schema contains a fact table and one or more dimension tables.

string  A sequence of characters treated as a unit.

Structured Query Language (SQL)  A computer language used to access data in relational databases.

synonym  An alternate name for an object, such as a table or a view, in a relational database management system (RDBMS). Some RDBMSs use the term "alias" to refer to a synonym. Not all RDBMSs support synonyms.

table  In relational databases, a form of data storage in which data is stored in records comprised of fields. Each record is defined by a unique, or primary, key.

transformation rules  In Essbase Integration Services, a set of instructions that define how to change or reformat the member names and data you extract from the source relational database.

two-pass  An Essbase property that is used to recalculate members that are dependent on the calculated values of other members. Two-pass members are calculated during a second pass through the outline.

unary operator  A mathematical indicator (+, -, *, /, %) associated with an outline member. The unary operator defines how the member is calculated during a database roll-up.

Unicode-mode application  An Essbase application wherein character text is encoded in UTF-8, enabling users with computers set up for different languages to share application data.
union  An SQL command that is a type of join that combines the results of two SELECT statements. A union is often used to merge lists of values contained in two tables.

user-defined attribute (UDA)  An attribute, associated with members of an outline to describe a characteristic of the members, that can be used to return lists of members that have the specified associated UDA.

user-defined dimension  Dimensions that you explicitly create in Essbase Integration Services, rather than dimensions obtained or built from the relational data source.

user-defined member  Members that you explicitly create in Essbase Integration Services, rather than obtaining and building them from the relational data source.

user-defined tables  Logical tables that you create in Essbase Integration Services, rather than in the RDBMS. These virtual tables, which behave as standard RDBMS views, can be used anywhere regular RDBMS tables or views are used. User-defined tables enable you to create and to edit OLAP models without altering your relational schema or modifying the SQL generated by Integration Services.

validation  (1) In Essbase, a process of checking a rules file against the outline to make sure the rules file is valid. (2) In Essbase Integration Services, a process of checking the OLAP model and metaoutline.

view  In relational databases, logical table created by combining columns from one or more tables. A view can contain metadata and formatting information to query an OLAP data source.

virtual tables  See user-defined tables.

virtual views  See user-defined tables.

write-back  The ability for a retrieval client, such as a spreadsheet, to update a database value.

XML Import/Export  In Essbase Integration Services, a function that enables importing data into and exporting data from an OLAP Metadata Catalog in Extended Markup Language (XML) file format. Both OLAP models and metaoutlines can be saved as XML files and imported into other OLAP Metadata Catalogs.
Index

Symbols
& (ampersand), 15
/eis/server/samples/tbcdbase, 112

A
access
codes, for metaoutlines, 121
codes, for OLAP models, 120
exclusive access mode, 116
multi-user, 116
privileges, common problems, 124
problems, correcting, 136
setting, 119
standard access mode, 116
accessing
data sources, 123
OLAP Metadata Catalog, 123, 125
servers and databases, troubleshooting, 123
accessing, servers and databases, 63
accounts
problems in the RDBMS, 128
user, administrative, 128
user, problems, 137
ADDDUPLICATESFORASO (ais.cfg parameter), 31
adding data source to odbc.ini, 47, 54, 62
administration tools, Unicode-enabled, 36
administrative accounts, using, 128
aggregate storage, 186
ais.cfg
IBM DB2 OS/390 configuration parameter (for UNIX), 62
IBM DB2 OS/390 configuration parameter (for Windows), 47
ais.cfg file, 14, 15
described, 28
JDBC data source mapping, 69, 70
storing startup switch settings, 28
ais.cfg file parameters, 31
ADDDUPLICATESFORASO, 31
UNIFORMVALUEFORMAT, 31
aliases, 39
ampersand, 15
Anaytic Services, testing connection from Integration Server to Essbase Server, 134
application, non-Unicode mode, 36
applications, Unicode-mode, defined, 35
artifact limits for source databases, 184
artifact limits in Integration Services, 181
AT service, 165, 166

B
backtracking connection problems, 126
backups, OLAP Metadata Catalog, 83
bat extension files, 151
batch files
for loading TBC database, 95
Integration Services Shell and, 148
running, 151
running Integration Services Shell scripts with, 151
scheduling manually, 165, 166
batch mode, Integration Services Shell, 148
binding, requirement for IBM DB2 databases, 140
block storage, 186
brackets, in Integration Services Shell syntax, 150
building
OLAP Metadata Catalog, 73
sample application, 89

C
CB_CONCURRENCY table
column descriptions, 117
defined, 88
CB_OBJECT_TYPE column, data defined, 117
CB_VERSION table, 88

cbs extension files, 151

checking
  common connection problems, 123
  release number of Integration Services, 133
  script syntax for loads, 151

client programs, custom, 36

client software
  port number, 142
  problems, correcting, 142
  RDBMS, common problems, 124
  RDBMS, correcting problems, 139
  RDBMS, required, 42
  troubleshooting, 123

code page, defined, 35

codes, access
  for metaoutlines, 121
  for OLAP models, 120

columns, viewing in sample application, 113

Command Center, 79

command line
  Integration Services Shell, 148
  loading data from, 166
  loading members from, 165

command syntax, Integration Services Shell, 150

Command Window, IBM DB2, 91

command-line interface, using, 147

commands, Integration Services Shell, 152

commands, status of, 154

committing records during data load, 18

common problems
  areas of failure, 125
  database-specific, 125
  with connections, 123

communication problems, 125

compatibility
  between RDBMS and RDBMS client, 139
  ODBC drivers, 140
  problems with product versions, 133

components, software, common problems, 124, 125

compression dimension, 185

concurrent users, 117

configuration parameters, 31
  ADDDUPLICATESFORASO, 31
  automatic validation, 30
  display primary keys, 30
  UNIFORMVALUEFORMAT, 31

configuration problems (ODBC), 125

configuring
  data sources on UNIX, 53
  data sources on Windows, 43
  ODBC (Open Database Connectivity) data sources, 41
  ODBC computer location, 42
  ODBC data sources on UNIX, 53
  ODBC data sources on Windows, 43
  odbc.ini file, 53, 54

correcting
  database client to RDBMS, 128
  problems with data source, 123, 125
  problems with ODBC Metadata Catalog, 123, 125
  setting defaults, 66
  to Essbase Server, 65
  to Integration Server, 63
  to ODBC Metadata Catalog, 63
  to relational data sources, 63, 66
  to servers and databases, 63

connections
  areas of failure, 125
  common problems, 123
  isolating problems, 126
  LOADMEMBER command, 132
  ODBC Metadata Catalog, 123, 125
  path through components, 127
  problems, correcting, 136
  problems, diagnosing, 125
  problems, isolating, 126
  testing, 129, 134
  troubleshooting, 123

creating
  database for TBC, 94, 110
  database for TBC_MD, 97
  ODBC server software data sources, 41
  ODBC Metadata Catalog automatically, 75
OLAP Metadata Catalog manually, 76, 79
OLAP Metadata Catalog tables, 74
OLAP Metadata Catalogs, 73, 74
sample application, 89
tables for TBC database, 95
TBC database, 94
critical server error messages, 22
cron scheduling daemon, 39, 165

D
data loads
  checking script syntax, 151
  command line and, 166
  enabling scheduled loads, 39
  error messages, 175
  incremental, script syntax, 153
  Integration Services Shell and, 165, 166
  LOADALL command, 161
  LOADDATA command, 159
  OLAP Metadata Catalog and, 13, 83
  scripts, recording, 151
  time required, 149
  with TBC sample application tables, 95
data migration
  OLAP Metadata Catalog, 77
data sources
  accessing, 123
  adding to odbc.ini (UNIX), 47, 54, 62
  configuring, 41
  configuring on UNIX systems, 53
  configuring on Windows systems, 43
  connecting to, 63, 66, 165, 166
  connection problems, 123, 125
  connections to, diagram, 127
  identifying before loading, 155
  problems, correcting, 140
  setting up TBC, 94
data warehouse, 115
data, retrieving, 13
database (source), artifact limits, 184
database client software
  environment problems, 128, 139
  information, 42
  problems, correcting, 139
  testing connection to RDBMS, 128
database client software, common problems, 124
database scripts
  for OLAP Metadata Catalog, 78
  for TBC database, 94
  for TBC_MD database, 97
database servers
  common problems, 124
  problems, correcting, 138
database users, source, 115
databases
  configuring for ODBC, 41
  connecting to, 63
  connections to, diagram, 127
  OLAP Metadata Catalog, requirements, 74
  TBC, creating, 94, 110
  TBC, requirements, 94, 97, 110
  TBC_MD, creating, 96, 97
  TBC_MD, requirements, 97
  user aliases and synonyms, 39
  vendor-specific problems, 125
DataDirect Text driver
  configuring on UNIX, 57–59
  configuring on Windows, 47–50
DataDirect, ODBC driver problems, correcting, 140
dataload.txt file, 34
DB2. See IBM DB2
ddtestlib, 141
ddtestlib program, 129
debug messages, 22
defaults
  for connections, setting, 66
  login, setting, 66
Delete Locks command, 117
deleting
  Integration Server users, 115
  locks, 117, 118
  OLAP Metadata Catalog, 81
  OLAP Metadata Catalog tables, 78
  read and write locks, 118
  TBC database tables, 94
  TBC_MD database tables, 97
  users, 115
diagnosing problems (connections, ODBC), 123
diagram, connections path, 127
dialog boxes
  Login, 64, 100
  OLAP Metadata Catalog Setup, 75
  Sample Application Setup, 92
  Set Login Defaults, 66
directories
- sample application scripts, 93, 103
- TBC data load script, 93, 101, 103, 105, 108, 111
- TBC table creation script, 93, 101, 103, 105, 108
- TBC_MD data load script, 93
- TBC_MD table creation script, 93, 102, 104, 106, 108, 111
- disconnecting, users, 115
- displaying primary keys in OLAP models, 30
- DISTINCT clause in drill-through SQL, 19
- DNS, problems, 126
- domain name server, problems, 126
- drill-through reports
  - error messages, 176
  - guidelines, 192
  - limits, 192
  - tables relating to, 86
- driver file, ODBC, specifying on UNIX, 131
- drivers, ODBC, problems, 140
- dropping tables
  - OLAP Metadata Catalog, 78
  - TBC database, 94
  - TBC_MD database, 97
  - DTR_ATTRIBUTES table, 87
  - DTR_INFO table, 87

E
- editing
  - configuration for a data source, 53
  - odbc.ini file, 54
- enabling
  - regional options, 37
  - regional options in Windows 2003, 38
  - regional options in Windows XP, 37
  - scheduled loads on UNIX, 39
  - scheduled loads on Windows, 39
- encoding, defined, 35
- environment variables
  - common problems, 124
  - for database client, problems, 128, 139
  - for ODBC on UNIX, 54
  - ODBC problems, 129
- error messages
  - data load, 175
  - drill-through reports, 176
  - member load, 172
  - miscellaneous server messages, 177
- setting logging level, 22

Essbase
- Essbase Server, connecting to, 65, 166
  - message file, 33
  - server name problems, 137
- Essbase Services
  - relationship to Integration Server, 13
  - essbase.mdb file, 23
  - examples, odbc.ini files, 55–61, 60, 61
  - exclusive access mode, 116
  - executable name
    - for Integration Server, 14
    - for Integration Services Shell, 148
- EXIT command, 165
- extra locks
  - deleting, 118
  - viewing, 117

F
- file name, Essbase messages file, 20
- file names, 23
- fixing. See troubleshooting

G
- guidelines for drill-through reports, 192
- guidelines for Hybrid Analysis, 185

H
- host names, problems, 137
- Hybrid Analysis
  - enabling in manual sample application setup, 89
  - guidelines, 185
  - limits on transformation rules, 187
  - required utilities for sample application setup, 101
sample application for IBM DB2, 101
sample application for Microsoft SQL Server, 105
sample application for Oracle, 103
sample application for Teradata, 107
sample application setup, 101
transformations not supported, 187
unsupported Essbase functions, 188
viewing the metaoutline, 114
viewing the OLAP model, 113
Hybrid Analysis, defined, 87
Hybrid Analysis, tables relating to, 87
hyperion user name, permissions required, 39

IBM DB2
-tvf utility, 79, 91
Command Center utility, 79
Command Window, 91
common problems, 125
configuring the Wire Protocol driver on Windows, 45
database binding requirement, 140
odbc.ini file example (UNIX), 55
IBM DB2 OS/390
ais.cfg entry for Windows, 47
configuration parameter in ais.cfg (for UNIX), 62
configuration parameter in ais.cfg (for Windows), 47
default port number, 45
odbc.ini file example (UNIX), 56
ignore parent during drill-through, 17
incompatibility, problems with product versions, 133
incremental loads
LOADALL command, 161
LOADDATA command, 159
LOADMEMBER command, 156
script syntax, 153
informational messages, 22
installation
manual steps, UNIX, 72
installing
OLAP Metadata Catalog, 73
Integration Server
common problems, 123
connecting to, 63
connection defaults, setting, 66
defined, 13
determining user access, 118
error message file, 33
error messages, 171
executable name, 14
executable name in UNIX, 14
log files, 32
logging on to, 153, 165, 166
logging out of, 164
message file, 33
OLAP Metadata Catalog, information in, 83
olapisvr file, 14
port number, 24, 142, 148
prerequisites for, 14
problems, correcting, 141
reconnecting to, 32
server problems, 137
shutting down, 32, 165
starting, 14, 32
starting as a Windows service, 15
starting from the command line, 14
starting from the Windows Desktop, 15
startup switches, 16
startup.bat file, 14
stopping, 32
stopping as a Windows Service, 32
stopping from command line, 32
stopping from Windows Desktop, 32
switches, 16
testing connection to Essbase Server, 134
testing connection to RDBMS, 132
users, defined, 115
Integration Server users
deleting, 115
disconnecting, 115
multiple, 117
viewing locks, 117
Integration Services
artifact limits, 181
connection path, diagram, 127
release number, checking, 133
Integration Services Console
common problems, 123
port number, 24, 142, 148
problems, correcting, 142
testing connection to Essbase Server, 136
testing connection to RDBMS, 134
Integration Services Shell
  asynchronous commands, 154
  batch mode, 148
  command syntax, 150
  commands, list, 152
  description, 148
  executable name, 148
  executing commands, 150
  interactive mode, 148
  loading data with, 166
  loading members with, 165
  running scripts with, 151
  starting, 147
  using, 147
  using to test connections, 132
Integration Services Shell commands
  EXIT, 165
  LOADALL, 161
  LOADDATA, 159
  LOADMEMBER, 156
  LOGIN, 153
  LOGOUT, 164
  SETCATALOG, 156
  SETSOURCE, 155
  SETTARGET, 155
  SHUTDOWN, 165
  STATUS, 154
  syntax, 150
  VERSION, 154
Integration Services users
  creating, 115
  viewing, 115
interactive mode, Integration Services Shell, 148
is.csh file, using, 130
is.sh file, using, 130
isolating, connection problems, overview, 126
ivtestlib, 141
ivtestlib program, 129

J
JDBC driver
  mapping a data source for Teradata, 69, 70
JOIN_HINTS table, 85

L
lc*.sql files, 99
ld*.sql files, 95
levels of access permissions, 120
library path, common problems, 124
limits
  drill-through report guidelines, 192
  Hybrid Analysis guidelines, 185
  Integration Services artifacts, 181
  source database artifacts, 184
  transformation rules with Hybrid Analysis, 187
  transformations not supported by Hybrid Analysis, 187
  unsupported Essbase functions in Hybrid Analysis, 188
listeners, setting, 26
LOAD commands, 149
LOADALL command, described, 149, 161
LOADDATA command
  described, 149, 159
  example of, 166
loading
  data, time required, 149
  from the command line, 166
  Integration Services Shell and, 165, 166
  OLAP Metadata Catalog and, 13, 83
  test, members into Essbase database, 134
loading data
  into TBC database, 95
  into TBC_MD database using XML Import utility, 99
LOADMEMBER command
  connections made by, 132
  described, 149, 156
  example of, 166
  using to test connections, 132
loads, scheduled, enabling, 39
locale, defined, 35
locks
  deleting, 117, 118
  exclusive access mode, 116
  metaoutlines, 116
  OLAP models, 116
  read, deleting, 118
  read, viewing, 118
  standard access mode, 116
  storage location in the OLAP Metadata Catalog, 117
  timestamps and, 117
unreleased, 117
viewing, 117
working with, 115, 116
write, deleting, 118
write, viewing, 118
log files
changing name and location, 20
contents, 32
Integration Server, viewing, 32
messages recording level, 22
logging level, 22
logging on, to servers and databases, 63
LOGIN command
described, 153
example of, 165, 166
login defaults, setting, 66
Login dialog box, 64, 100
login problems, correcting, 137
login script
diagnosing problems, 130
problems, 128
LOGOUT command, 164

M
managing
locks and permissions, 115
user names and passwords, 137
manual steps, UNIX installation, 72
MB_FILTER table, 84
MB_TX_RULE table, 84
member loads
checking script syntax, 151
command line, 165, 166
enabling scheduled loads, 39
error messages, 172
LOADMEMBER command, 156
OLAP Metadata Catalog and, 13, 83
scripts, recording, 151
Merant.. See DataDirect, ODBC driver problems,
correcting
message file, Essbase, 33
message file, Integration Server, 33
messages, debug, 22
messages, error, 171
Metadata Catalog, creating Unicode, 76
metaoutlines
access codes, 121
access permissions, 119
locks, 116
opening in exclusive access mode, 116
opening in standard access mode, 116
overwriting, 116
permissions, viewing, 120
restoring, 84
tables in OLAP Metadata Catalog, 85
users, 117
validating automatically, 30
Microsoft SQL Server
example odbc.ini file on UNIX, 59
Query Analyzer utility, 79, 91
Microsoft SQL Server driver, configuring on
Windows, 50–51
Microsoft SQL Server, common problems, 125
migration
OLAP Metadata Catalog, 77
MO_ACCESS_CODE column, data defined, 121
MO_DESCRIPTIONS table, 86
MO_FILTERS table, 85
MO_INFO table, 85, 120
MO_INTERSECTION table, 87
MO_PROPERTIES table, 86, 87
MOD_ATTRIBUTES table, 86
MOD_INFO table, 86
MODEL_ACCESS_CODE column, data defined, 120
MS_INFO table, 86
MS_FILTERS table, 86
MS.Predicate table, 86
MS.Predicate table, 86
MS_TX_RULE table, 86
MSR_INFO table, 86
MSR.Predicate table, 86
MSR.Tx.Rule table, 86
multi-threaded server, 13
multi-user access, 116

N
names, server problems, 137
network listeners, setting, 26
network problems, 126
non-critical error messages, 22
non-Unicode-mode application, defined, 36

O
OA_INFO table, 84
oc*_upgrd20.sql files, 76, 77
oc_create_* .sql files, 79
oc_create_*_unicode.sql files, 97, 98
oc_drop_*.sql files, 79
oc_upgrade20_*.sql files, 79
oc_upgrade61_*.sql files, 79
oc_upgrade651_*.sql files, 79
oc_upgrade65_*.sql files, 79
ocscript directory, contents of, 78
OD_DETAILS table, 84
OD_INFO table, 84
ODBC
  common problems, 124
driver compatibility, 140
driver file, specifying on UNIX, 131
environment problems, 128
problems, correcting, 140
testing connection to RDBMS, 129
testing on UNIX, 130
testing on Windows, 129
tracing on UNIX, 144
tracing utility, using, 143
troubleshooting, 123
ODBC (Open Database Connectivity)
  configuring data sources, 41
  configuring, which computer, 42
data sources, odbc.ini file heading, 55
environment variables on UNIX, 54
required database client software, 42
ODBC Administrator, 43, 44, 129
ODBC Data Sources, odbc.ini file section, 131
ODBC drivers
  adding a data source to odbc.ini file on UNIX, 54
  configuring DataDirect Text driver on UNIX, 57–59
  configuring DataDirect Text driver on Windows, 47–50
  configuring IBM DB2 OS/390 on UNIX, 56
  configuring IBM DB2 OS/390 on Windows, 45
  configuring IBM DB2 UDB Wire Protocol on UNIX, 55
  configuring IBM DB2 UDB Wire Protocol on Windows, 45
  configuring Microsoft SQL Server driver on UNIX, 59
  configuring Microsoft SQL Server driver on Windows, 50, 51
  configuring on UNIX, 53–61
configuring on Windows, 43–53
configuring Oracle Wire Protocol on UNIX, 57
configuring Oracle Wire Protocol on Windows, 46
configuring SQL Server driver on Windows, 50–51
configuring Teradata driver on AIX, 60
configuring Teradata driver on HP-UX, 60
configuring Teradata driver on Solaris, 61
configuring Teradata driver on Windows, 52–53
configuring the odbc.ini file on UNIX, 54
configuring Wire Protocol drivers on Windows, 44–45
editing a data source name on Windows, 53
testing Wire Protocol drivers on Windows, 46
odbc.ini file
  adding a data source to, 47, 54, 62
  bracketed headings, 55
  common problems, 129
  configuring data sources, 54
  editing, 53
  IBM DB2 example, 55
  IBM DB2 OS/390 example, 56
  inspecting for problems, 131
  Microsoft SQL Server example, 59
  Oracle example, 57
  Teradata example, 60, 61
  Teradata example on AIX, 60
  Teradata example on HP-UX, 60
  Teradata example on Solaris, 61
  Trace parameter, 144
ODBCINI variable, 131
OLAP Command Script files, 151
OLAP Metadata Catalog
  accessing, 115
  backups, 83
  connecting to, 63, 165, 166
  connection defaults, setting, 66
  connection problems, 123, 125
  creating, 73
  creating automatically, 75
  creating tables for manually, 76, 79
  data migration, 77
  database platforms supported, 73
  database requirements, 74
deleting, 81
deleting locks, 118
deleting tables, 76, 79
drill-through tables in, 86
Hybrid Analysis tables in, 87
identifying before loading, 156
information, 73
Integration Server and, 13
manual editing, 83
metaoutline tables, 85
OLAP model tables, 84
post-setup steps, 81
problems, database level, 138
recovery, 83
upgrading, 77
viewing permissions, 120
OLAP Metadata Catalog Setup dialog box, 75
OLAP metaoutlines, viewing in sample application, 113, 114
OLAP models
access codes, 121
access permissions, 119
accessing, 115
displaying primary keys, 30
locks, 116
opening in exclusive access mode, 116
opening in standard access mode, 116
overwriting, 116
permissions, viewing, 119
restoring, 84
tables in OLAP Metadata Catalog, 84
users, 117, 118
OLAP models, viewing in sample application, 113
OLAPICMD program
problems, 132
starting, 148
olapisrv.log file, 16
olapisrv file, using to start Integration Server, 14
OLAPISVR program
command-line switches, 16
starting in UNIX, 14
starting using startup.bat file, 14
olapisvr.log file, specifying location, 20
OM_DESCRIPTION table, 85
OM_INFO table, 84, 119
OM_PROPERTIES table, 84
OMB_DETAILS table, 84
OMB_INFO table, 84
open database connectivity. See ODBC (Open Database Connectivity)
opening metaoutlines, options, 116
opening OLAP models, options, 116
Oracle
configuring the Wire Protocol driver on Windows, 46
odbc.ini file example, 57
SQL*Loader, 95
SQL*Plus, 79, 91
tnsnames.ora file, 140
orphaned locks, deleting, 117, 118
OV_INFO table, 85
overwriting, OLAP models and metaoutlines, problem, 116
OVL_REL_DETAILS table, 85
OVL_REL_INFO table, 85
OVP_REL_DETAILS table, 85
OVP_REL_INFO table, 85

P
parentheses, adding to drill-through filters, 20
password problems, 123, 137
permissions
changing, 119
for metaoutlines, 119, 120
for OLAP models, 119
problems in the RDBMS, 128
required for hyperion user, 39
shutting down Integration Server, 165
viewing, 119
working with, 115
port 3388, 24, 142, 148
prerequisites
for Integration Server, 14
Integration Services Shell, 147
primary data source, 155
privileges, common problems, 124
problems
starting Integration Server, 14
system-level, 136
TCP/IP, 126
with connections, correcting, 123, 125
with ODBC, correcting, 123, 125, 139
profile file, 130
Q
Query Analyzer utility (Microsoft SQL Server), 79, 91

R
RDBMS
- client problems, correcting, 139
- common problems, 124
- problems, correcting, 138
- server name problems, 137
- testing connection from database client, 128
- testing connection from Integration Server, 132
- testing connection from Integration Services Console, 134
- testing connection to ODBC, 129
- testing ODBC connection on UNIX, 130
- testing ODBC connection on Windows, 129
- user account problems, 128
read access, setting, 119
read locks
  - deleting, 118
  - viewing, 118
rebuilding tables, 78
reconnecting to Integration Server, 32
recording data load scripts, 151
recording scripts, 151
records, committing during data load, 18
recovery, OLAP Metadata Catalog, 83
regional options
  - enabling, 37
  - enabling in Windows 2003, 38
  - enabling in Windows XP, 37
relational database management system. See RDBMS
relational databases
  - client software required, 42
  - connections to, diagram, 127
  - problems, correcting, 138
  - user aliases and synonyms, 39
releases
  - compatibility problems, 133
  - determining for Integration Services, 154
removing
  - OLAP Metadata Catalog, 81
  - OLAP Metadata Catalog tables, 78
  - TBC database tables, 94
  - TBC_MD database tables, 97
restoring
  - metaoutlines, 84
  - OLAP Metadata Catalogs, 83
  - OLAP models, 84
restricting access to table records, 25
rights, problems in the RDBMS, 128
round-trip problem, Unicode as solution, 36
running
  - Integration Server, 14, 32
  - Integration Services Shell, 147
scripts with Integration Services Shell, 151

S
sample application
  - creating, 89
  - creating for Hybrid Analysis, 101
  - creating for Unicode, 110
  - preparation steps, 91
  - setting up automatically, 91
  - setting up manually, 93
  - Unicode (Oracle only), 110
  - viewing, 113, 114
Sample Application Setup dialog box, 92
sample database, TBC, 90
samples\tbcdbase directory, contents of, 94
scheduled loads, enabling on UNIX, 39
scheduling, batch files for loads, 165, 166
scripts
  - for loading Hybrid Analysis data, 95
  - for loading TBC database, 95
  - for loading TBC_MD database, 99
  - for OLAP Metadata Catalog, 78
  - for TBC database, 94
  - for TBC_MD database, 97
incremental loads, syntax, 153
location, sample application, 93, 103
member and data loads, recording, 151
running with Integration Services Shell, 151
secondary data source, 155
security
  - access permissions, 119
  - locks, 116
  - problems, 126
  - semicolons, in Integration Services Shell syntax, 150
server problems, 137
server software
  - port number, 142
  - problems, correcting, 141
troubleshooting, 123
Servers menu, OLAP Metadata Catalog, Delete Locks command, 117
servers, connecting to, 63
SET commands, 149
Set Login Defaults dialog box, 66
SETCATALOG command
  described, 149, 156
  example of, 165, 166
SETLOCALE command, described, 149
SETSOURCE command
  described, 149, 155
  example of, 165, 166
SETTARGET command
  described, 149, 155
  example of, 165, 166
setting
  connection defaults, 66
  connection information for Essbase Server, 65
data source, 165, 166
Essbase Server, 165, 166
network listeners, 26
OLAP Metadata Catalog, 156, 165, 166
primary data source, 155
secondary data source, 155
target Essbase Server, 155
shell scripts
  for loading TBC database, 95
SHUTDOWN command, 32, 165
shutting down Integration Server, 32, 165
software components
  common problem areas, 125
  connection problems, 124
source database users, 115
source database, artifact limits, 184
SQL scripts
  for loading TBC database, 95
  for OLAP Metadata Catalog, 78, 79
  for TBC database, 94
  for TBC_MD database, 97
location, sample application, 93, 103
SQL Server. See Microsoft SQL Server
SQL Server driver, configuring on Windows, 50–51
SQL*Loader, 95
SQL*Plus, 79, 91
stack size of thread, specifying, 26
standard access mode, 116
starting
  Integration Server, 14, 32
  Integration Server, methods, 14
  Integration Server, programs necessary, 14
  Integration Services Shell, 147
  schedule services on Windows, 39
startup switches for Integration Server, 16
startup.bat file, 14, 15
  described, 29
  storing startup switch settings, 29
  using to start Integration Server, 14
STATUS command, described, 150, 154
stopping Integration Server, 32
storage
  metaoutline, 83
  OLAP model, 83
  switches, for OLAPISVR, 16
  synonyms, 39
  syntax, Integration Services Shell commands, 150
system-level problems, 136
T
  tables
    creating for OLAP Metadata Catalog manually, 76, 79
    creating for TBC database, 94
    creating for TBC_MD database, 97
drill-through, 86
Hybrid Analysis, 87
metaoutline, 85
miscellaneous, 87
OLAP model, 84
read-only, 83
rebuilding tables, 78
restricting access, 25
upgrading for OLAP Metadata Catalog, 77	tables, viewing in sample application, 113
Task Scheduler service
  Windows 2003, starting, 39
TBC
  data load script location, 93, 101, 103, 105, 108, 111
database requirements, 94, 97, 110
database, creating tables for, 94, 95
database, viewing, 113, 114
loading data into, 95
sample database, 90
setting up relational data source, 94
  table creation script location, 93, 101, 103, 105, 108
  viewing columns in, 113
  viewing metaoutline in, 114
  viewing OLAP model in, 113
  viewing tables in, 113
TBC Metaoutline
  information, 90
  viewing, 113, 114
TBC Model
  information, 90
  viewing, 113, 114
tbc_create_database_name_unicode.sql file, 112
TBC_MD
  data load script location, 93
  database requirements, 97
  database, creating, 96
  database, creating tables for, 97
  loading data into using XML Import, 99
  sample OLAP Metadata Catalog, 90
  table creation script location, 93, 102, 104, 106, 108, 111
TBC_Model_Unicode.xml, 112
TCP/IP problems, 126
Teradata
  example odbc.ini file for AIX, 60
  example odbc.ini file for HP-UX, 60
  example odbc.ini file for Solaris, 61
  mapping a JDBC driver, 69, 70
  odbc.ini file example, 60, 61
Teradata ODBC driver, configuring on Windows, 52–53
  testing
  connections, overview, 126
  database client, 128
  Integration Server connection to Essbase Server, 134
  Integration Server connection to RDBMS, 132
  Integration Services Console connection to RDBMS, 134
  Integration Services Console to Essbase Server, 136
  ODBC connection to RDBMS, 129
  ODBC on UNIX, 130
  ODBC on Windows, 129
Text driver
  configuring on UNIX, 57–59
  configuring on Windows, 47–50
  threads
  specifying number during data load, 23
  specifying stack size, 26
  timestamp, locks and, 117
tnsnames.ora file, 140
  Trace parameter, odbc.ini file, 144
  tracing utility
    ODBC for UNIX, 144
    ODBC for Windows, 144
  transformation rules, limits with Hybrid Analysis, 187
  transformations not supported by Hybrid Analysis, 187
  troubleshooting
    connections, 123
    ODBC, 123
    system-level problems, 136
    TCP/IP, 126
  turning off Integration Server, 32

U
  Unicode
    administration tools, 36
    application modes, 35
    creating OLAP Metadata Catalog, 76
    features not supported, 35
    parameter in LOADALL command, 161, 164
    parameter in LOADMEMBER command, 157, 158
    required utilities for sample application setup, 110
    sample application for IBM DB2, 111
    sample application setup, 110
    viewing the metaoutline, 114
    viewing the OLAP model, 114
  Unicode-enabled
    administration tools, 36
    parameter in LOADALL command, 161, 164
    parameter in LOADMEMBER command, 157, 158
  Unicode-mode application, defined, 35
  UNIFORMVALUEFORMAT (ais.cfg parameter), 31
UNIX
  environment variables for ODBC, 54
  environment variables, configuring, 54
  manual installation steps, 72
ODBC data sources, configuring, 53
ODBC environment variables, 54
ODBC tracing, 144
scheduled loads, enabling, 39
testing ODBC, 130
unload database and application from memory, 27
unreleased locks, deleting, 117, 118
upgrading
  OLAP Metadata Catalog, 77
user accounts, administrative, 128
user aliases, 39
user names
  locks and, 117
  problems with, 123, 137
user synonyms, 39
users
  deleting, 115
  disconnecting, 115
  Essbase, defined, 115
  Integration Server, defined, 115
  managing permissions and locks, 115
  multiple, 117
  source database users, 115
UTF-8 encoding, 35

V
validation of metaoutlines, 30
variables. See environment variables
VERSION command, described, 150, 154
versions, compatibility problems, 133
vi program, 54
viewing
  log files, 32
  permissions for metaoutlines, 120
  permissions for OLAP models, 119
  viewing sample application, 113, 114

W
warning messages, 22
wide area network (WAN), 64
Windows
  ODBC testing, 129
  ODBC tracing, 144
  testing ODBC, 129
Windows 2003
  enabling regional options, 38
Task Scheduler service, starting, 39
Windows service
  Task Scheduler, starting, 39
  using -I parameter, 21
  using to start Integration Server, 15
  using to stop Integration Server, 32
Windows XP
  enabling regional options, 37
  ODBC Administrator, 43
  write access, setting, 119
  write locks
    deleting, 118
    viewing, 118

X
XML files for loading TBC_MD database, 99
XML Import for loading TBC_MD database, 99