

Oracle® Essbase

Using Oracle Essbase



F17137-05
March 2020



Oracle Essbase Using Oracle Essbase,

F17137-05

Copyright © 2019, 2020, Oracle and/or its affiliates.

Primary Author: Essbase Information Development Team

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

1 Access Oracle Essbase

Access Tools and Tasks from the Console	1-1
Essbase, REST, and Smart View Client URLs	1-2
Set Up Your Client	1-2

2 Top Tasks for Oracle Essbase

Top Tasks Prerequisites	2-1
Understand Application Workbooks	2-1
Download the Sample Dynamic Application Workbook	2-1
Examine the Structure of the Sample Dynamic Application Workbook	2-2
Create an Application in the Essbase Web Interface and Provision a User to Access and Query the Cube	2-4
Create an Application in the Essbase Web Interface	2-4
Provision a User to Access and Query the Cube	2-4
Analyze an Application in Smart View	2-5
Connect to the Cube from Smart View	2-5
Perform an Ad hoc Analysis	2-6
Modify an Essbase Outline	2-9
Analyze Forecast Data in Smart View	2-10
Create an Application and Cube in Cube Designer	2-13
Open the Application Workbook in Cube Designer	2-14
Create, Load, and Calculate the Cube	2-14
View the Application in the Web Interface	2-14
Analyze Data and Perform an Incremental Update in Cube Designer	2-15
Analyze Data in the Sample Basic Cube	2-15
Perform an Incremental Update on the Sample Basic Cube	2-15
Transform Tabular Data into a Cube	2-18
Export and Modify Metadata and Data	2-19

3 Manage Essbase Files and Artifacts

Explore the Files Catalog	3-1
---------------------------	-----

Explore the Gallery Templates	3-1
Applications Templates	3-2
Technical Templates	3-3
System Performance Templates	3-3
Access Files and Artifacts	3-4
Explore the Application Directories	3-4
Work with Files and Artifacts	3-5

4 Understand Your Access Permissions in Essbase

User Role	4-1
Database Access Permission	4-1
Database Update Permission	4-2
Database Manager Permission	4-2
Application Manager Permission	4-3
Power User Role	4-3
Service Administrator Role	4-4
About Filters	4-4
Create Filters	4-5
Create Efficient Dynamic Filters	4-6
Dynamic Filter Syntax	4-6
Workflow to Create Dynamic Filters	4-7
Example of a Dynamic Filter	4-8

5 Design and Create Cubes Using Application Workbooks

About Application Workbooks	5-1
Download a Sample Application Workbook	5-2
Create a Cube from an Application Workbook	5-2
Export a Cube to an Application Workbook	5-3
Connect to a Cube in Smart View	5-4

6 Design and Manage Cubes from Tabular Data

Transform Tabular Data to Cubes	6-1
Use Intrinsic Headers to Transform Tabular Data to Cubes	6-1
Use Forced Designation Headers to Transform Tabular Data Into Cubes	6-2
Create and Update a Cube from Tabular Data	6-5
Export a Cube to Tabular Data	6-6

7 Create and Manage Cube Outlines Using the Web Interface

About Cube Outlines	7-1
View and Edit Outline Properties for a Newly Created Cube	7-1
Work with General and Attribute-related Outline Properties	7-2
Understand and Create Alias Tables	7-5
Understand and Work With Dynamic Time Series Outline Properties	7-5
Understand and Create Textual Measures Outline Properties	7-6
Create a Sample Cube to Explore Outline Properties	7-7
Set Outline Properties in your Sample Cube	7-7
Add Dimensions and Members to Outlines	7-8
Add Dimensions to Outlines Manually	7-8
Add Members to Outlines Manually	7-9
Work with Attributes	7-9
About Duplicate Member Names	7-10
Set Dimension and Member Properties	7-11
Open the Outline in Edit Mode	7-11
Set Member Properties while in Edit Mode	7-11
Set Properties in the Member Inspector	7-12
Set General Properties	7-12
Create Aliases	7-16
Create Member Formulas	7-17
Set Attribute Associations	7-18
Associate an Attribute Dimension with a Base Dimension	7-18
Associate an Attribute Member with a Member of the Base Dimension	7-18
Create User-Defined Attributes	7-19
Select the Member Properties to Display in the Outline	7-19
Name Generations and Levels	7-20
Generate Aggregate Views Automatically	7-20
Set Advanced Cube Properties	7-21
Unlock Objects	7-21
Remove Data Locks	7-21

8 Use Connections and Datasources

About Connections and Datasources	8-1
Create Connections and Datasources	8-2
Create a Connection and Datasource to Access Oracle Database	8-2
Create a Connection and Datasource for Oracle Autonomous Data Warehouse	8-4
Create a Connection and Datasource to Access Another Cube	8-6

9 Build Dimensions and Load Data

Typical Workflow for Dimension Builds and Data Loads	9-1
About Dimension Builds	9-1
About Data Loads	9-2
Work with Rules	9-2
Global and Field Options	9-3
Global Options	9-4
Field Options	9-5
Build Dimensions and Load Data Using a Rule File	9-6
Build Dimensions Using a Rule File	9-6
Load Data Using a Rule File	9-9
Upload Files to a Cube	9-11
Build Dimensions and Load Data by Streaming from a Remote Database	9-12
Build Dimensions and Load Data Using SQL	9-14
Build Dimensions Using SQL	9-15
Load Data Using SQL	9-20

10 Calculate Cubes

Access to Calculations	10-1
Create Calculation Scripts	10-1
Execute Calculations	10-2
Use Substitution Variables	10-3
Set Two-Pass Calculation Properties	10-5
Trace Calculations	10-5
Calculate Selected Tuples	10-8
Tuple-Based Calculation	10-10
Select Tuples for Point of View Calculation	10-10
Examples of Tuple Selection to Reduce Calculation Scope	10-11
No Tuple Selection	10-12
Selection of Named Sparse Dimensions	10-12
Selection of Contextual Sparse Dimensions	10-13

11 Run and Manage Jobs Using the Web Interface

View Job Status and Details	11-1
Execute Jobs	11-1
Build Aggregations	11-2
Clear Aggregations	11-3

Export to Table Format	11-3
Run Calculation	11-4
Build Dimension	11-4
Clear Data	11-6
Export Data	11-6
Export Excel	11-7
Export LCM	11-7
Import LCM	11-8
Load Data	11-9
Run MDX	11-9

12 Adopt Hybrid Mode for Fast Analytic Processing

Benefits of Hybrid Mode	12-2
Comparison of Hybrid Mode, Block Storage, and Aggregate Storage	12-2
Get Started with Hybrid Mode	12-3
Optimize the Database for Hybrid Mode	12-3
Limitations and Exceptions to Hybrid Mode	12-4
Solve Order in Hybrid Mode	12-4

13 Model Data in Private Scenarios

Understand Scenarios	13-1
View and Work with Scenario Data	13-2
View and Work With Scenario Data From the Essbase Web Interface	13-2
View and Work With Scenario Data From a Smart View Private Connection	13-3
About Scenario Calculations	13-4
About Data Loads to Scenario-enabled Cubes	13-4
About Data Exports from Scenario-enabled Cubes	13-5
About Transparent and Replicated Partitions in Scenario-enabled Cubes	13-5
About XREF/XWRITE in Scenario-enabled Cubes	13-5
About Audit Trail in Scenario-enabled Cubes	13-6
About Scenario Limitations	13-7
Scenario Workflow	13-8
Enable Email Notifications for Scenario Status Changes	13-9
Create a Scenario	13-10
Model Data	13-10
Submit a Scenario for Approval	13-11
Approve or Reject Scenario Changes	13-11
Apply or Discard Data Changes	13-11
Copy a Scenario	13-12

Delete the Scenario	13-12
Understand Scenario User Roles and Workflow	13-12
Enable Scenario Modeling	13-13
Create a Scenario-Enabled Cube	13-14
Create a Scenario-Enabled Sample Cube	13-14
Enable an Existing Cube for Scenario Management	13-14
Create Additional Sandbox Members	13-15
Work with Scenarios	13-15
View Base Member Data	13-15
Compare Scenario Values to Base Values	13-16
Set Scenario Cells to #Missing	13-16
Revert Scenario Values Back to Base Values	13-17
Understand When to Aggregate Sandbox Dimensions	13-18
Example: Calculate Scenarios with Dynamic Upper Level Members	13-18
Example: Calculate Scenarios with Stored Upper Level Members	13-20

14 Work with Cubes in Cube Designer

About Cube Designer	14-1
About the Cube Designer Ribbon	14-1
About the Designer Panel	14-3
Manage Files in Cube Designer	14-4
Download Sample Application Workbooks	14-4
Build a Private Inventory of Application Workbooks	14-4
Open an Application Workbook	14-5
Save an Application Workbook	14-5
Export to an Application Workbook	14-5
Work with Application Workbooks in Cube Designer	14-5
Limitations of Application Workbooks	14-6
Work with the Essbase.Cube Worksheet in Cube Designer	14-6
Work with the Cube.Settings Worksheet: Alias Tables in Cube Designer	14-8
Work with the Cube.Settings Worksheet: Properties in Cube Designer	14-8
Work with the Cube.Settings Worksheet: Dynamic Time Series in Cube Designer	14-9
Work with the Cube.Settings Worksheet: Attribute Settings in Cube Designer	14-9
Work with the Cube.Settings Worksheet: Substitution Variables in Cube Designer	14-10
Work with Dimension Worksheets in Cube Designer	14-10
Work with Data Worksheets in Cube Designer	14-12
Work with Calculation Worksheets in Cube Designer	14-13
Work with MDX Worksheets in Cube Designer	14-13
Create a Cube from a Local Application Workbook in Cube Designer	14-13

Work with Text Lists Worksheets in Cube Designer	14-14
Create a Cube from Tabular Data in Cube Designer	14-14
Update Cubes Incrementally in Cube Designer	14-17
Create and Validate Member Formulas in Cube Designer	14-18
Load Data in Cube Designer	14-19
Calculate Data in Cube Designer	14-20
Work with Jobs in Cube Designer	14-20
View Jobs in the Cube Designer Job Viewer	14-20
Monitor Cube Designer Jobs	14-21
Troubleshoot Jobs in the Cube Designer Job Viewer	14-21
Clear and Archive Cube Designer Jobs	14-21
View Dimension Hierarchies in Cube Designer	14-21
Export Cubes to Application Workbooks in Cube Designer	14-22
Delete Applications and Cubes in Cube Designer	14-23
Unlock Objects in Cube Designer	14-24
View Logs in Cube Designer	14-24

15 Track Changes to Data

Turn on Data Audit Trail and View the Data Audit Trail	15-1
Link a Report Object to a Cell	15-2
Export Logs to a Sheet	15-2
Refresh the Audit Log	15-3
View and Manage Audit Trail Data in the Essbase Web Interface	15-3

16 Link Cubes Using Partitions and XREF/XWRITE

Define a Reusable Connection for Partitions or XREF/XWRITE	16-1
Understand Transparent and Replicated Partitions	16-2
Create a Transparent Partition	16-2
Create a Replicated Partition	16-3
Refresh a Replicated Partition	16-4
Understand XREF/XWRITE	16-4
Create a Location Alias Based on a Defined Connection	16-5

17 Configure Oracle Essbase

Set Application-Level Configuration Properties	17-1
Set Provider Services Configuration Properties	17-2

18 Essbase Command-Line Interface (CLI)

Download and Use the Command-Line Interface	18-1
CLI Command Reference	18-2
Login/Logout: CLI Authentication	18-3
Calc: Run a Calculation Script	18-4
Clear: Remove Data from a Cube	18-5
Createlocalconnection: Save a JDBC Connection	18-6
Dataload: Load Data to a Cube	18-7
Deletefile: Remove Cube Files	18-9
Deploy: Create a Cube from a Workbook	18-10
Dimbuild: Load Dimensions to a Cube	18-11
Download: Get Cube Files	18-12
Help: Display Command Syntax	18-13
LcmExport: Back Up Cube Files	18-14
LcmImport: Restore Cube Files	18-14
Listapp: Display Applications	18-16
Listdb: Display Cubes	18-16
Listfiles: Display Files	18-16
Listfilters: View Security Filters	18-17
Listlocks: View Locks	18-18
Listvariables: Display Substitution Variables	18-18
Setpassword: Store CLI Credentials	18-19
Start: Start an Application or Cube	18-19
Stop: Stop an Application or Cube	18-20
Unsetpassword: Remove Stored CLI Credentials	18-20
Upload: Add Cube Files	18-20
Version: Display API Version	18-22

19 Manage Essbase Using the MaxL Client

Prerequisites to Set Up the MaxL Client	19-1
Download and Use the MaxL Client	19-2

20 Analyze Data in the Web Interface

Perform Ad Hoc Analysis in the Web Interface	20-1
Work with Layouts	20-2
Access to Layouts	20-3
Analyze and Manage Data with MDX	20-3
Analyze Data with MDX Reports	20-3
Access to MDX Reports	20-4

Examples of MDX Reports	20-5
Insert and Export Data with MDX	20-7
Run MDX Scripts	20-7
Write, Upload, and Run an MDX Script	20-7
Write an MDX Script in the Script Editor and Run It	20-7
Create an MDX Script in Cube Designer and Run it	20-8
Guidelines for MDX Scripts	20-8
Examples of MDX Scripts	20-8

21 Use Logs to Monitor Performance

Download Application Logs	21-1
About Performance Analyzer	21-1
Enable Performance Analyzer and Set the Data Collection Interval	21-2
Understand and Work With Performance Analyzer Data	21-2

22 Analyze Cube Data with Drill Through Reports

About Drill Through Reports	22-1
Access to Drill Through Reports	22-1
Typical Workflow for Drill Through Reports	22-2
Use Cases and Column Mapping	22-2
Map Generation Name to a Data Source Column	22-3
Map Dimension to a Data Source Column	22-5
Map Level 0 to a Data Source Column	22-6
Map Multiple Cells and Regions	22-6
Create Drill Through Reports	22-8
Create a Drill Through Connection and Data Source	22-8
Define Report Columns and Drillable Regions	22-9
Execute Drill Through Reports	22-10
Format Drill Through Reports	22-10
Run Drill Through Reports	22-10

A Application Workbooks Reference

Understand the Essbase.Cube Worksheet	A-1
Understand the Cube.Settings Worksheet	A-3
Understand the Cube.Settings Worksheet: Alias Tables	A-3
Understand the Cube.Settings Worksheet: Properties	A-4
Understand the Cube.Settings Worksheet: Dynamic Time Series	A-5
Understand the Cube.Settings Worksheet: Attribute Settings	A-6
Understand the Cube.Settings Worksheet: Substitution Variables	A-8

Understand the Cube.Generations Worksheet	A-8
Understand the Cube.Textlists Worksheet	A-11
Understand Dimension Worksheets	A-12
Understand Data Worksheets	A-17
Understand Calculation Worksheets	A-20
Understand MDX Worksheets	A-22

B Set up Cube Designer

Workflow to Set up Cube Designer	B-1
Download and Run the Smart View Installer	B-1
Connect to Essbase	B-2
Install the Smart View Cube Designer Extension	B-2
Update the Smart View Cube Designer Extension	B-3
Delete Smart View Connection URLs	B-4

Accessibility and Support

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 that have purchased support 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.

1

Access Oracle Essbase

Oracle Essbase is a business analytics solution that uses a proven, flexible, best-in-class architecture for analysis, reporting, and collaboration. Essbase delivers instant value and greater productivity for your business users, analysts, modelers, and decision-makers, across all lines of business within your organization.

Access Essbase using credentials supplied by your Service Administrator.

To access Essbase, you must have the following information:

- URL to access the Essbase web interface
- User name
- Password
- Identity domain to which you belong

After you log in to the Essbase web interface, the **Applications** home page is displayed.

Access Tools and Tasks from the Console

As a user or service administrator, you can access various tools and tasks that you will need.

Users and administrators have access to Console actions from the Essbase web interface. Note that the bold terms below represent the options listed on the Console.

As a non-service administrator user, you can:

- Download **desktop tools** that you will install locally and use for administration, import, and export. See [Set Up Your Client](#).
- Monitor your own user **sessions**.
- View **database size statistics** for applications for which you're a provisioned user.

As a service administrator, you can:

- Download **desktop tools** that you install locally and use for administration, import, and export. See [Set Up Your Client](#).
- Set platform-based **email configuration** for email notifications of scenario status changes. See [Enable Email Notifications for Scenario Status Changes](#).
- View Essbase **logs**. See [Use Logs to Monitor Performance](#).
- Enable **file scanner** to scan files and ensure they're virus-free before they're uploaded into Essbase.
- Monitor and manage all user **sessions**.
- View **database size statistics** for all applications.
- View agent and server **configuration** and add Provider Services.

- Add substitution **variables** that apply to all Essbase applications. See [Use Substitution Variables](#).
- View **ODBC settings** on the server, which can be used for debugging database issues.
- Enable **Performance Analyzer** to capture incremental log data according to the interval you set in the Console. See [About Performance Analyzer](#) and [System Performance Templates](#).

Essbase, REST, and Smart View Client URLs

Get the URL for the Oracle Essbase instance you are using from your Service Administrator. The basic format of the URL is:

```
https://IP-address:port/essbase
```

The default SSL port is 443, unless it was changed during stack creation.

For example:

```
https://192.0.2.1:443/essbase
```

Essbase components, such as the Smart View client and the REST API, have their own URLs.

Sample Smart View client URL:

```
https://192.0.2.1:443/essbase/smartview
```

You can access Smart View if you have valid credentials. You can also configure the Smart View URL. See [Connect to Essbase](#).

A discovery URL has `/agent` appended to the end. You can use it to log in to the MaxL Client, and to access Essbase from Oracle BI and Data Visualization. Example:

```
https://192.0.2.1:443/essbase/agent
```

The following is an example of a REST API URL:

```
https://192.0.2.1:443/essbase/rest/v1
```

Set Up Your Client

In the Console, you can download desktop tools to use for administration, import, and export. Set up your local client computer using these tools. Many of your interactions with Essbase originate from your local machine. Be sure you're using the latest versions provided in the Console, as older, previously downloaded versions may not work correctly.

- **Command Line Tools**

- **Command-line Tool (CLI)**—Provides a command line interface for common Essbase administrative tasks.
See [Download and Use the Command-Line Interface](#).
- **Life Cycle Management (LCM)**—Backs up Essbase 11g on-premises cubes and artifacts so you can import them to Essbase 19c.
See [Migrate an On-Premises Application Using LCM Utility](#).
- **Migration Utility**—Provides a command line interface for migrating Essbase applications, cubes, artifacts, and users between Essbase instances.
See [Migrate Cloud Service Applications Using Migration Utility](#).
- **Export Utility**—Creates an application workbook, from an existing cube, that you can use to import the cube and its artifacts. Use this to migrate Essbase 11g on-premises cubes to Essbase 19c.
See [Export On-Premises Cubes to Import to the Cloud](#).
- **Smart View**
 - **Smart View for Essbase**—Provides a Microsoft Office interface for data analysis. It is the out-of-box query interface for Essbase.
 - **Cube Designer Extension**—Deploys Essbase cubes from formatted application workbooks. Cube Designer is an add-in to Smart View that enables desktop design of Essbase cubes. It can also be used to deploy cubes from tabular data in an Excel worksheet.
See [Set up Cube Designer](#).
- **Essbase MaxL Clients**—Provides Linux and Windows clients to enable scripting of Essbase administrative tasks. MaxL is an administrative, language-based interface for managing Essbase cubes and artifacts.
See [Manage Essbase Using the MaxL Client](#).
- **Essbase Clients**—Provides libraries for Essbase C API.
- **Essbase Java API**—Enables development of Essbase client tools in Java, and provides libraries, samples and documentation for the Essbase Java API.

2

Top Tasks for Oracle Essbase

These topics take you through a series of workflows that cover many of the top tasks that you can do in the Essbase web interface and in Cube Designer, depending on your access.

- [Top Tasks Prerequisites](#)
- [Understand Application Workbooks](#)
- [Create an Application in the Essbase Web Interface and Provision a User to Access and Query the Cube](#)
- [Analyze an Application in Smart View](#)
- [Modify an Essbase Outline](#)
- [Analyze Forecast Data in Smart View](#)
- [Create an Application and Cube in Cube Designer](#)
- [Analyze Data and Perform an Incremental Update in Cube Designer](#)
- [Transform Tabular Data into a Cube](#)
- [Export and Modify Metadata and Data](#)

Top Tasks Prerequisites

Before you start reviewing the top tasks topics, be sure you have met these prerequisites:

1. Be sure that you can log in to Essbase.
2. Be sure that Smart View and Cube Designer extension are installed on client computers.

See [Set up Cube Designer](#).

Understand Application Workbooks

The **gallery** section of the File Catalog provides a collection of sample application workbooks that you can modify for your own use to quickly deploy an application and cube.

Now you'll learn about the structure of an application workbook.

Download the Sample Dynamic Application Workbook

In the Block Storage Sample (Dynamic) application workbook, all non-leaf level members in the cube are dynamically calculated. Dynamically calculated values are not stored in the cube; the values are recalculated and rendered for each user retrieval.

To download the Block Storage Sample (Dynamic) application workbook:

1. On the Applications page, click **Files**, then click **Gallery, Applications, Demo Samples**, and **Block Storage**.
2. On the Block Storage page, click the Actions menu next to **Sample_Dynamic_Basic.xlsx**.
3. Save the application workbook file, `Sample_Dynamic_Basic.xlsx`, to a local drive.

Examine the Structure of the Sample Dynamic Application Workbook

Application workbooks contain a number of worksheets that define the metadata for the cube.

1. In Microsoft Excel, open `Sample_Basic_Dynamic.xlsx`.
2. On the `Essbase.Cube` worksheet, the application name (`Sample_Dynamic`), cube name (`Basic`), the names of 10 dimensions, and other information about the dimensions, are defined.

	A	B	C	D	E
1	Application Name	Sample_Dynamic			
2	Database Name	Basic			
3	Version	1.0			
4					
5	Dimension Definitions				
6					
7		Dimension Type	Storage Type	Outline Order	Base Dimension
8	Year	Time	Dense	1	
9	Measure	Accounts	Dense	2	
10	Product	Regular	Sparse	3	
11	Market	Regular	Sparse	4	
12	Plan	Regular	Dense	5	
13	Caffeinated	Attribute-Boolean		6	Product
14	Ounces	Attribute-Numeric		7	Product
15	Pkg Type	Attribute-Text		8	Product
16	Population	Attribute-Numeric		9	Market
17	Intro Date	Attribute-Date		10	Product

3. Each dimension has a separate worksheet, `Dim.dimname`, in which the dimension is further defined with information such as the build method and incremental mode. Because the build method for each dimension in this sample application workbook is PARENT-CHILD, members are defined in PARENT and CHILD columns.

On the `Dim.Year` worksheet, months roll up to quarters, and quarters roll up to years. For example, child members Jan, Feb, Mar roll up to parent member Qtr1. Child member Qtr1 rolls up to parent member Year.

	A	B	C
1	Dimension Name	Year	
2			
3	Definitions		
4	File Name	Dim_Year	
5	Rule Name	Dim_Year	
6	Build Method	PARENT-CHILD	
7	Incremental Mode	Merge	
8			
9	Members		
10	Columns	PARENT	CHILD
11			Year
12		Year	Qtr1
13		Qtr1	Jan
14		Qtr1	Feb
15		Qtr1	Mar

The Dim.Product and Dim.Market worksheets are similarly structured. In Dim.Product, SKUs roll up to product families, and product families roll up to Product. For example, child members 100-10, 100-20, and 100-30 (SKUs) roll up to parent member 100 (product family). Child member 100 rolls up to parent member Product.

	A	B	C
1	Dimension Name	Product	
2			
3	Definitions		
4	File Name	Dim_Product	
5	Rule Name	Dim_Product	
6	Build Method	PARENT-CHILD	
7	Incremental Mode	Merge	
8			
9	Members		
10	Columns	PARENT	CHILD
11			Product
12		Product	100
13		100	100-10
14		100	100-20
15		100	100-30

- This sample application workbook includes data. Scroll to the last worksheet, Data.Basic, to review the structure of the columns and the data.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Definitions												
2	File Name	Cube_Basic											
3	Rule Name	Basic											
4	Data Load Option	Add											
5													
6	Data												
7	Columns	Dimensio	Dimensio	Dimensio	Dimensio	Measure.	Measure.	Measure.	Measure.	Measure.	Measure.	Measure.	Measure.E
8		100-10	New York	Jan	Actual	678	271	94	51	0	2101	644	2067
9		100-10	New York	Jan	Budget	640	260	80	40	#Missing	2030	600	1990
10		100-10	New York	Feb	Actual	645	258	90	51	1	2067	619	2041
11		100-10	New York	Feb	Budget	610	240	80	40	#Missing	1990	600	1980
12		100-10	New York	Mar	Actual	675	270	94	51	1	2041	742	2108
13		100-10	New York	Mar	Budget	640	250	80	40	#Missing	1980	700	2040

In this topic, you learned about the structure of an application workbook. Next, learn how to access additional templates using the Gallery section of the File Catalog.

Create an Application in the Essbase Web Interface and Provision a User to Access and Query the Cube

In [Understand Application Workbooks](#), you learned about the structure of an application workbook by exploring `Sample_Basic_Dynamic.xlsx`.

Now, you use this workbook to learn how to create an application in the Essbase web interface and provision a user to access and query the cube.

Create an Application in the Essbase Web Interface

Use this workbook to learn how to create an application from a workbook in the Essbase web interface.

1. In the web interface, on the Applications page, click **Import**.
2. On the Import dialog box, click **File Browser** (as the workbook was downloaded to the local file system). Open the Block Storage Sample (Dynamic) application workbook, `Sample_Basic_Dynamic.xlsx`, that you saved in [Understand Application Workbooks](#).
3. Expand **Advanced Options** and **Build Option**, select **Create Database**, and then check the box to load data. You do not need to select **Execute Scripts**, because all measures and aggregations along hierarchies in the cube are dynamically calculated at query time.
4. Click **OK**. In a few moments, the `Sample_Dynamic` application and Basic cube are created.
5. On the Applications page, expand the **Sample_Dynamic** application, and select the cube, **Basic**.
6. In the Actions list for the cube, select **Outline**. The outline is a representation of the dimensions in the Basic cube as defined in the application workbook. The outline opens in a separate browser tab, allowing you to navigate between the outline and other web interface actions.
7. View a cube dimension, and then drill down into the children of that dimension:
 - a. Expand the **Year** dimension to view the quarters.
 - b. Expand the individual quarters to view months.

Now all of the information from the application workbook is represented in the new cube.

Provision a User to Access and Query the Cube

Now, you provision a user to access and query the cube.

1. Log in as a power user. This allows you to provision other users to the applications you have created.
2. Return to the web interface browser tab and go to **Applications**.
3. Select the application for which you want to provision the user; in this example, select **Sample_Dynamic**. If you select the cube instead of the application, then you won't be able to provision user roles.

4. Use the Actions menu to open the application inspector.
5. Select the **Permissions** tab within the application inspector.
6. Select **+** to see the list of users on the system and select the **+** next to each user to assign their access.
7. Use the radio button controls next to each user to assign their access. Select **Database Manager** for each added user. The Database Manager has full control of the cube, but no control over the application.
8. Click **Close**.

In [Analyze an Application in Smart View](#), you'll go to Smart View, log in as the user you just provisioned, and then query a cube.

Analyze an Application in Smart View

In [Create an Application in the Essbase Web Interface and Provision a User to Access and Query the Cube](#), you created an application and a cube with data, and provisioned users.

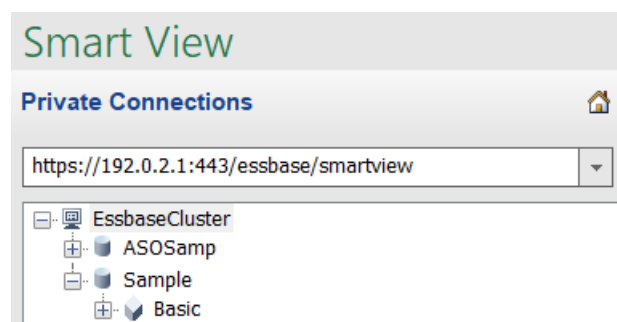
Now you'll learn how to connect to the cube from Smart View, and do some analysis of the data.

This task assumes that you installed Smart View. See [Download and Run the Smart View Installer](#).

Connect to the Cube from Smart View

Connect to a cube from Smart View so that you can perform analysis.

1. Open Microsoft Excel.
If Smart View is installed, you can see the Smart View ribbon.
2. On the Smart View ribbon, click **Panel**.
3. On the Smart View Home dialog box, click the arrow next to the **Home** button, then select **Private Connections**.
4. Make a private connection using the same URL that you used to connect to Essbase, and append `/essbase/smartview` to the end of that URL. For example, `https://192.0.2.1:443/essbase/smartview`.
5. Log in as the user you created.
6. Expand EssbaseCluster.



7. Highlight the Basic cube, and click **Connect**.

Perform an Ad hoc Analysis

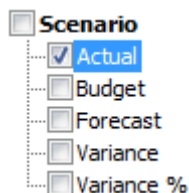
Once you're connected to the Basic cube, you're ready to begin analyzing the data.


You can specify the ancestor position for a hierarchy as top or bottom, in the Member Options tab of the Options dialog in Smart View. SSANCESTORONTOP must first be enabled by an administrator in application configuration in the Essbase web interface. You can see the change in an existing grid when you perform a zoom-in operation. Here, just use the default bottom position.

1. On the **EssbaseCluster** tree, under **Sample_Dynamic**, select the **Basic** cube, then click **Ad hoc analysis**.
2. In the resulting grid, you can see one aggregated data value for all five dimensions of this dynamic cube.

	Product	Market	Scenario
	Measures		
Year	105522		

3. Navigate into the member Scenario and narrow it down to a specific scenario type of Actual data.
 - a. Click the cell containing Scenario.
 - b. On the Essbase ribbon, click **Member Selection**.
 - c. In the Member Selection dialog box, check the box next to the Actual member.



- d. Click **Add**  to move Actual to the right pane.
- e. If Scenario is already included in the right pane, highlight it and use the left arrow to remove it, and then click **OK**.

On the Essbase ribbon, click **Refresh**. The grid should now look like this:

	Product	Market	Actual
	Measures		
Year	105522		

4. Navigate into Measures and narrow it down to the Sales member, to look at sales data.
 - a. Highlight the cell containing Measures.

- b. On the Essbase ribbon, click **Zoom In**.
- c. Highlight the cell containing Profit, and click **Zoom In**.
- d. Highlight the cell containing Margin, and click **Zoom In**.
- e. Highlight the cell containing Sales, and click **Keep Only**.

The grid should now look like this:

	Product	Market	Actual
	Sales		
Year	400855		

- 5. Zoom in to Year by double-clicking the cell containing Year.
The grid should now look like this:

	Product	Market	Actual
	Sales		
Qtr1	95820		
Qtr2	101679		
Qtr3	105215		
Qtr4	98141		
Year	400855		

- 6. Zoom in to Product by double-clicking the cell containing Product.
The grid should now look like this:

		Market	Actual
		Sales	
Colas	Qtr1	25048	
Colas	Qtr2	27187	
Colas	Qtr3	28544	
Colas	Qtr4	25355	
Colas	Year	106134	
Root Beer	Qtr1	26627	
Root Beer	Qtr2	27401	
Root Beer	Qtr3	27942	
Root Beer	Qtr4	27116	
Root Beer	Year	109086	
Cream Soda	Qtr1	23997	
Cream Soda	Qtr2	25736	
Cream Soda	Qtr3	26650	
Cream Soda	Qtr4	25022	
Cream Soda	Year	101405	
Fruit Soda	Qtr1	20148	
Fruit Soda	Qtr2	21355	
Fruit Soda	Qtr3	22079	
Fruit Soda	Qtr4	20648	
Fruit Soda	Year	84230	
Water Beve	Qtr1	#Missing	
Water Beve	Qtr2	#Missing	

- Enhance your data display to show time periods per product. Pivot Qtr1 of Colas by highlighting it, right-clicking and holding, then dragging it from B3 to C3. The grid should now look like this:

	Market	Actual			
	Sales	Sales	Sales	Sales	Sales
	Qtr1	Qtr2	Qtr3	Qtr4	Year
Colas	25048	27187	28544	25355	106134
Root Beer	26627	27401	27942	27116	109086
Cream Soda	23997	25736	26650	25022	101405
Fruit Soda	20148	21355	22079	20648	84230
Water Beve	#Missing	#Missing	#Missing	#Missing	#Missing
Product	95820	101679	105215	98141	400855

- Look at each product by region. Double-click Market in B1. The grid should now look like this:

		Actual				
		Sales	Sales	Sales	Sales	Sales
		Qtr1	Qtr2	Qtr3	Qtr4	Year
East	Colas	6292	7230	7770	6448	27740
East	Root Be	5726	5902	5863	6181	23672
East	Cream S	4868	5327	5142	4904	20241
East	Fruit So	3735	3990	4201	3819	15745
East	Water E	#Missing	#Missing	#Missing	#Missing	#Missing
East	Product	20621	22449	22976	21352	87398
West	Colas	6950	7178	7423	6755	28306
West	Root Be	8278	8524	8885	8513	34200
West	Cream S	8043	8982	9616	8750	35391

- Drill in to a region to view product sales by state. Double-click East in A4. Because not every product is sold in every state, some cells have the #Missing label instead of a data value.

In this task, you navigated through a data grid easily, zooming in and pivoting by clicking in the grid itself. You can also use the tools on the Essbase ribbon to perform the same actions. For more help on using Smart View, click the Smart View tab, and then click **Help**.

In [Modify an Essbase Outline](#), you'll go back to the web interface and modify an outline.

Modify an Essbase Outline

In [Analyze an Application in Smart View](#), you analyzed an application in Smart View.

Now you'll modify a cube outline in the web interface.

Create a New Member

You start by creating a new member.

- In the web interface, on the Applications page, select the **Basic** cube in the **Sample_Dynamic** application.
- Click the **Actions** menu, and select **Outline**.
- Click **Edit**.
- Expand the Scenario dimension by clicking the arrow next to **Scenario**.
- Insert a member:
 - Click **Edit** to put the outline in edit mode.
 - Expand the **Scenario** dimension.
 - Select the **Budget** member.
 - On the outline toolbar, under **Actions**, select **Add a sibling below the selected member**.
- Enter the member name, **Forecast**, and press **Tab**.
- Select the tilde (~) consolidation operator from the list.

The Forecast member does not aggregate with the other members in its dimension.

8. Leave the data storage type as **Store Data** because we want users to be able to input forecast data.
9. Click **Save**.

Seed the Forecast Member with Data

To seed the Forecast member with data, we'll create a calculation script and calculate forecast data.

1. In the web interface, on the Applications page, select **Basic** cube in the **Sample_Dynamic** application, click the **Actions menu**, and select **Inspect**.
2. In the Basic dialog box, select the **Scripts** tab, with **Calculation Scripts** selected, click **+** to add a calculation script.
3. In the **Script Name** field, enter `salesfcst`.
4. In the **Script Content** box, enter a simple formula:

```
Forecast(Sales=Sales->Actual*1.03;)
```

Forecast for sales is equal to actual sales multiplied by 1.03, which seeds the Forecast member for Sales with a value 3% higher than the actual sales.

5. Click **Save and Close**.
6. Close the database inspector by clicking **Close** until all tabs are closed.

Execute the Script

Calculation scripts are executed as jobs.

1. In the Web interface, select the Jobs page.
2. Click **New Job**, and select **Run Calculation**.
3. On the Run Calculation dialog box, in the **Application** field, select **Sample_Dynamic** application.

Notice that the **Database** field automatically populates the **Basic** cube.

4. On the **Scripts** menu, select the `salesfcst` calculation script that you created.
5. Click **OK**.
6. Click **Refresh** to see that the job completes.

In [Analyze Forecast Data in Smart View](#), you'll analyze this new forecast data in Excel. But first, let's take a closer look at managing jobs.

Analyze Forecast Data in Smart View

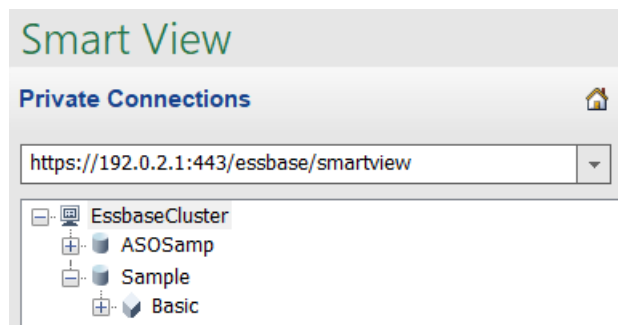
In [Analyze an Application in Smart View](#), you learned to analyze data in Smart View. In [Modify an Essbase Outline](#), you added a Forecast member to the outline, and seeded it with data.

Now you'll reconnect to the cube in Smart View, and do further analysis of the data.

1. Open Excel and create a worksheet like the following one, by typing the member names in these cells: A3=Market, B3=Product, C1=Year, C2=Actual, D1=Sales, D2=Forecast.

	A	B	C	D
1			Year	Sales
2			Actual	Forecast
3	Market	Product		

2. On the Smart View ribbon, reconnect to Basic cube in the Sample_Dynamic application.



Your previous connection URL should be shown in the list of Private Connections.

3. When prompted to log in, connect as the user you provisioned.
4. To populate cells with data values, click **Ad hoc analysis**. In the resulting grid, you should be able to see the results of your calculation. The yearly sales data refreshes for both Actual and Forecast, and the forecast is about 3% higher than the actual:

	A	B	C	D
1			Year	Sales
2			Actual	Forecast
3	Market	Product	400511	412526.3

5. To test that the calculation is correct, create this Excel formula, =D3/C3, in cell E3, which divides the forecast data by the actual data, to ensure that D3 is 3% higher than C3.

	A	B	C	D	E
1			Year	Sales	
2			Actual	Forecast	
3	Market	Product	400511	412526.3	=D3/C3

The test result should confirm the 3% increase, in which Actual is 400511, Forecast is 412526.3, and E3 is 1.0.

	A	B	C	D	E
1			Year	Sales	
2			Actual	Forecast	
3	Market	Product	400511	412526.3	1.03

6. Zoom in on Product and Market. You can see that for all products and all markets, the forecast data is present and is 3% higher than the actual.

	A	B	C	D
1			Year	Sales
2			Actual	Forecast
3	East	Colas	27740	28572.2
4	East	Root Beer	23672	24382.16
5	East	Cream Soda	20241	20848.23
6	East	Fruit Soda	15745	16217.35
7	East	Diet Drinks	7919	8156.57
8	East	Product	87398	90019.94
9	West	Colas	28306	29155.18
10	West	Root Beer	34200	35226
11	West	Cream Soda	35391	36452.73
12	West	Fruit Soda	35034	36085.02
13	West	Diet Drinks	36423	37515.69
14	West	Product	132931	136918.9
15	South	Colas	16280	16768.4

7. Now, build a worksheet that you will use to do a data analysis on the forecast, and make some changes.
- Click the cell containing Forecast, then click **Keep Only**.
 - Select cells A3-B3 containing East and Colas, then click **Keep Only**.
The grid should now look like this:

	A	B	C	D
1			Year	Sales
2			Forecast	
3	East	Colas	28572.2	

- With cells A3-B3 still selected, click **Zoom In** to view per-state information for detailed product SKUs.
The grid should now look like this:

	A	B	C	D
1			Year	Sales
2			Forecast	
3	New Yo	Cola	9208.2	
4	New Yo	Diet Cola	#Missing	
5	New Yo	Caffeine Free Cola	#Missing	
6	New Yo	Colas	9208.2	
7	Massac	Cola	6713.54	
8		Diet Cola	#Missing	

- d. Pivot the Year dimension down into the columns. Highlight member **Year**, and select the arrow next to **zoom in** on the Essbase ribbon. Select **Zoom to bottom** to see the bottom level of the months. The grid should now look like this:

	A	B	C	D	E	F	G	H	I
1									Sales
2			Forecast	Forecast	Forecast	Forecast	Forecast	Forecast	Forecast
3			Jan	Feb	Mar	Apr	May	Jun	Jul
4	New Yo	Cola	698.34	664.35	695.25	733.36	778.68	916.7	939.36
5	New Yo	Diet Col	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
6	New Yo	Caffeine	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing	#Missing
7	New Yo	Colas	698.34	664.35	695.25	733.36	778.68	916.7	939.36
8	Massac	Cola	508.82	484.1	506.76	534.57	567.53	668.47	684.95

- e. Enter some monthly values to create a Diet Cola forecast. For example, enter 500 in each of the cells in the range C5:H5.

	A	B	C	D	E	F	G	H
1								
2			Forecast	Forecast	Forecast	Forecast	Forecast	Forecast
3			Jan	Feb	Mar	Apr	May	Jun
4	New Yo	Cola	698.34	664.35	695.25	733.36	778.68	916.7
5	New Yo	Diet Col	500	500	500	500	500	500

- f. Click **Submit Data**, and notice that the full year forecast in cell O5 changes to 3000, which is the sum of 500 in each of 6 months.

In this task, you learned how easy it is to analyze and edit the cube in Smart View, as long as you have the correct provisioning.

In [Create an Application and Cube in Cube Designer](#), you'll get familiar with Cube Designer.


Create an Application and Cube in Cube Designer

In [Analyze Forecast Data in Smart View](#), you analyzed data in Excel. Users working in Excel can design and deploy applications using Cube Designer.

Now you'll use Cube Designer to create an application and cube, similar to what we did in the web interface service in a previous task.

Open the Application Workbook in Cube Designer

Log in as a Power User and download Sample_Basic.xlsx from the Gallery.

1. In Excel, on the Cube Designer ribbon, click **Catalog**  .
If you are prompted to log in, then log in as a Power User.
2. Click **Gallery**, then **Applications/Demo Samples/Block Storage**, and double-click **Sample_Basic.xlsx**.

The Sample Basic application workbook is different from the Sample Basic Dynamic application workbook in that the Product and Market dimensions do not have dynamically calculated members.

For example, go to the Dim.Market worksheet in Sample_Basic.xlsx. Look at the **Storage** column. There are no X characters, which indicates that the members are stored. X characters in the **Storage** column indicate dynamically calculated members.


Therefore, after creating the dimensions and loading the data, you also need to calculate the cube.

Creating, loading, and calculating the cube can all be done in one step in the Build Cube dialog box.

Create, Load, and Calculate the Cube

Use Cube Designer to create, load, and calculate a cube from the Sample_Basic.xlsx application workbook.

1. On the Cube Designer ribbon, with the Sample Basic application workbook

(Sample_Basic.xlsx) still open, click **Build Cube**  .

2. On the **Build Option** menu, select **Create Cube**.
3. Click **Run**.

If there is an existing application with the same name, you are prompted to overwrite the application and cube. Click **Yes** to delete the original application and build this new application.

4. Click **Yes** to confirm your selection.

The **View Jobs** icon displays an hourglass while the job is in progress. The job runs in the background, and Cube Designer notifies you when the job is completed, which should display **Success**.

5. Click **Yes** to launch the Job Viewer and see the status of the job.

View the Application in the Web Interface

View and inspect the new application in the Essbase web interface.

1. Log into the web interface.
2. On the Applications page, expand the **Sample** application and select the **Basic** cube.
3. Click the Actions menu to the right of the **Basic** cube and select **Outline**.
View the outline, and see that the expected dimensions are present.
4. Return to the Applications page, expand the **Sample** application, and select the **Basic** cube.
5. Click the Actions menu to the right of the **Basic** cube and select **Inspect**.
6. In the inspector, select **Statistics**.
7. On the **General** tab, in the **Storage** column, you see that both level 0 and upper-level blocks exist, showing that the cube is fully calculated.

In [Analyze Data and Perform an Incremental Update in Cube Designer](#), you'll analyze data in this cube and perform incremental updates from Excel.



Analyze Data and Perform an Incremental Update in Cube Designer

In [Create an Application and Cube in Cube Designer](#), you executed a cube build, loaded data, and ran the calculation script defined in the workbook.

Now you'll analyze data, and then perform an incremental cube update.

Analyze Data in the Sample Basic Cube

Validate that the cube build was successful and take a quick look at how to analyze data.

1. In Excel, on the Cube Designer ribbon, click **Analyze**  .
2. On the **Analyze** menu, select **Connect Query Sheets**.
If you are prompted to log in, then enter your Essbase user name and password.
3. You're connected to the Basic cube in the Sample application.
4. You can now analyze the data.
 - a. Use the Essbase ribbon to zoom in on **Cream Soda** to see all of the low-level products that are part of the Cream Soda family.
 - b. Zoom out on **New York** to see all of the East region, and zoom out again to see all Markets.

Perform an Incremental Update on the Sample Basic Cube

Add a hierarchy to the product dimension and see the results in Smart View.

1. Go to the Dim.Product worksheet, where you'll update the product dimension with some extra products.
2. Insert new members into the workbook, following the 400 product family.

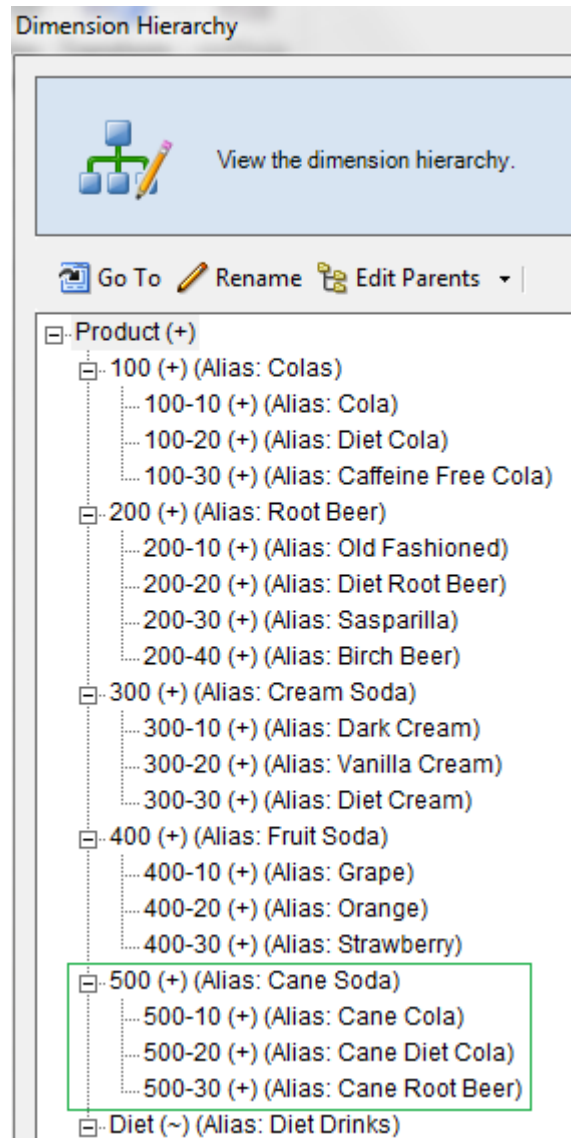
- a. Create a new parent Product with child 500 and give it the Alias Default name Cane Soda.
- b. Create three new SKUs with parent 500: 500-10, 500-20, and 500-30.
- c. Give aliases to the new SKUs. Call them Cane Cola, Cane Diet Cola, and Cane Root Beer.

Product	400			Fruit Soda
400	400-10			Grape
400	400-20			Orange
400	400-30			Strawberry
Product	500			Cane Soda
500	500-10			Cane Cola
500	500-20			Cane Diet Cola
500	500-30			Cane Root Beer
Product	Diet		~	Diet Drinks
Diet	100-20			Shared Diet Cola
Diet	200-20			Diet Root Beer
Diet	300-30			Diet Cream

3. Save the updated workbook.



4. Using the Cube Designer ribbon, click **Build Cube**.
The build option will default to **Update Cube – Retain All Data** since the application already exists on the server and you are the application owner who created it.
5. Click **Run**.
6. When the job completion notice is displayed, click **Yes** to launch the **Job Viewer**.
7. You should see **Success**. If the job returns **Error**, then you can double-click the job for more information.
8. Close the **Job Viewer**.
9. With the Dim.Product sheet active, click **Hierarchy Viewer** in the Cube Designer ribbon.
10. On the Dimension Hierarchy dialog box, see that the Cane Soda product group was created.



11. Go to the query worksheet, Query.Sample.
12. Navigate to the top of the Product dimension by highlighting Dark Cream and zooming out using the Essbase ribbon. Then zoom out on Cream Soda.
13. Select Product again and click **Zoom In**.
14. Select Cane Soda and click **Keep Only**.
15. Select Cane Soda and **Zoom In** to see the child members.

Adding members to the Product dimension does not populate those members with data. Data can be submitted using Smart View or by performing a data load.

Application workbooks are convenient tools for designing Essbase cubes when you already understand the elements needed to build a cube or when you have a sample.

In [Transform Tabular Data into a Cube](#), you will create an application using a columnar Excel worksheet without any Essbase-specific structure.

Transform Tabular Data into a Cube

You've learned to create cubes from application workbooks. You can also create cubes from tabular data. The tabular data can be from any source system (such as an ERP) or data warehouse, as long as the data contains facts and dimension information, and is contained in one worksheet in an Excel file.

In this task, you'll use the Cube Designer to create a cube from a Sales report and analyze the outline.



1. In Excel, select the Cube Designer ribbon, then click **Catalog**.
2. On the Essbase Files dialog box, in the **Gallery/Technical/Table Format** folder, double click **Sample_Table.xlsx**.

The `Sample_Table.xlsx` file contains a worksheet, `Sales`, which represents a common, simple sales report that you might receive from someone in your organization. The column headings indicate that there are measures (such as `Units` and `Discounts`), time representations (such as `Time.Month` and `Time.Quarter`), geographic regions (such as `Regions.Region` and `Regions.Areas`), and products (such as `Product.Brand` and `Product.LOB`).

From this report, you can create an application and cube by using introspection, which is a method of inspecting a physical data source (in this case, the `Sample_Table.xlsx` file) for Essbase metadata elements.

3. On the Cube Designer ribbon, click **Transform Data**.
4. On the Transform Data dialog box, you can accept the default names for the application (`Sample_Table`) and cube (`Sales`) or you can change them.
5. Cube Designer inspects the tabular data to detect relationships that determine appropriate dimensionality.
6. Click **Run** and, when prompted to create the cube, click **Yes**.
7. When the job is completed, you'll see the Job Viewer dialog box.
Click **Yes** until the status is Success.
8. Close the Job Viewer.
9. Log into the web interface.
10. On the Applications page, expand the **Sample_Table** application and select the **Sales** cube.
11. Click the Actions menu to the right of the **Sales** cube and select **Inspect**.
12. Select **Statistics**, and on the **General** tab, under **Storage**, the number 4928 for **Existing level 0 blocks** indicates that data has been loaded into the cube.
13. Use the General tab at the top of the database inspector to launch the outline.
In the outline editor, you can see that the `Sales` cube has the following dimensions: `Measures`, `Time`, `Years`, `Geo`, `Channel` and `Product`.
14. Click **Measures** to zoom in on the members in that dimension.

You'll notice that Units, Discounts, Fixed Costs, Variable Costs, and Revenue are in a flat hierarchy.

In [Export and Modify Metadata and Data](#), you'll create a hierarchy for these Measures so that you can see Revenue net of Discounts, and total costs (fixed and variable).

Export and Modify Metadata and Data

In [Transform Tabular Data into a Cube](#), you created an application and cube from tabular data.

In this task, you'll export the newly created application and cube to an application workbook.

1. In the Essbase web interface, on the Applications page, expand the **Sample_Table** application, and select the **Sales** cube.
2. From the **Actions** menu, select **Export to Excel**.
3. On the Export To Excel dialog box, select the Parent-Child **Export Build Method**.
4. Select **Export Data** and click **OK**.
 - If the data size is less than 400 MB, this exports the metadata and data to an Excel file called an application workbook. Save the application workbook, `Sales.xlsx`, to your Downloads area. The application workbook defines the cube that you exported.
 - If the data size exceeds 400 MB, the data file is saved in a compressed file and is not included in the exported Excel file. The ZIP file containing the data and the application workbook can be downloaded from **Files** page.
5. Open `Sales.xlsx`.
6. Scroll to the `Data.Sales` worksheet to view it. This is the data worksheet for the cube.

Examine the worksheets for each of the dimensions. The dimension worksheets begin with `Dim`, including the worksheet for the Measures dimension.

7. Using the exported application workbook, you can make further incremental updates. For example, you can add or remove hierarchies, append a formula to a measure, change aliases, and develop calculations, among many other tasks.

The sequenced tasks in this chapter are intended to show you how you can design and deploy cubes from application workbooks or tabular data. You can incrementally improve the design of your cubes by exporting them to application workbooks, making modifications, and rebuilding.

3

Manage Essbase Files and Artifacts

The Files catalog contains directories and files associated with using Essbase.

Topics:

- [Explore the Files Catalog](#)
- [Explore the Gallery Templates](#)
- [Access Files and Artifacts](#)
- [Explore the Application Directories](#)
- [Work with Files and Artifacts](#)

Explore the Files Catalog

The Files catalog helps you organize the information and artifacts associated with using Essbase.

You can access the Files catalog from Cube Designer or from the Essbase web interface.

The files catalog is grouped into the following folders:

- `applications`
- `gallery`
- `shared`
- `users`

What you can do in each folder depends on your permissions.

The `applications` folder is where Essbase saves applications and cubes.

The `gallery` folder contains application workbooks you can use to build sample cubes. These cubes help you learn about Essbase features, and model a variety of analytical problems across business domains.

The `shared` folder is a good location to store files and artifacts that you can use in more than one cube. Its contents are accessible to all users.

The `users` folder contains individual user directories. You can use your user folder for any files and artifacts that you use while working with Essbase.

In your own user folder, as well as in the shared folder, you can upload files and create subdirectories. No special permissions are required.

Explore the Gallery Templates

Gallery templates are application workbooks that you can use to build fully functional Essbase cubes. Think of these templates as starter kits you can use not only to build

cubes, but to learn about Essbase features, and to model a variety of analytical problems across business domains.

The gallery templates include README worksheets, describing the purpose and usage of the workbook and cube.

Gallery templates are packaged in the form of an application workbook, and may also have additional supporting files. You use an application workbook to create an application and cube using either of these methods: the **Import** button in the Essbase web interface, or the **Build Cube** button on the Cube Designer ribbon in Excel. To access the gallery from the Essbase web interface, click **Files** and navigate to the gallery section. To access the gallery from Cube Designer, use the **Essbase** button on the Cube Designer ribbon.

The gallery templates are grouped into the following categories:

- [Applications Templates](#)
- [Technical Templates](#)
- [System Performance Templates](#)

Applications Templates

Gallery templates in the Applications folder demonstrate various business use cases for Essbase across several organizational domains.

The following cubes, located in the `gallery/Applications/Sales and Operations Planning` folder, connect together to perform their respective aspects of sales and operational planning tasks:

- **Forecast Consensus**—develop and maintain an agreed-upon forecast shared across departments
- **Demand Consolidation**—forecast customer demand
- **Production Schedule**—compute a weekly master production schedule for all products and locations
- **Capacity Utilization**—ensure that existing plant capacity can handle the production schedule

Compensation Analytics illustrates how Human Resources analysts can perform headcount and compensation analysis, analyze attrition, and allocate compensation increases.

Organization Restatements demonstrates how operational expenses can be restated, after organizational changes, for internal management reporting.

Opportunity Pipe demonstrates how to manage a sales pipeline.

Spend Planning shows how procurement analysts can manage operational spending using top-down and bottom-up forecasting methods.

Project Analytics demonstrates project planning risk analysis, accounting for factors such as workforce skills and costs, revenue, margin, inventory, and schedule.

RFM Analysis demonstrates how to identify the most profitable customers based on metrics.

Consolidation Eliminations is a financial analysis application demonstrating how to identify and eliminate balances between two companies.

Organization Restatements is a financial analysis application demonstrating how to restate expenses after an organizational change.

In addition to these business applications, the Applications grouping of templates also includes:

- **Demo Samples**— simple examples of block storage and aggregate storage cubes commonly referenced in Essbase documentation.
- **Utilities**—cubes that may be utilized by other sample cubes. For example, the Currency Rates template takes currency symbols and returns the exchange rate to USD. The Currency Triangulation template uses a calculation script to triangulate currencies.

Technical Templates

The Technical templates demonstrate the use of specific Essbase features.

- **Calc: Allocation Tracing**—perform allocations and debug calculation scripts
- **Calc: Sample Basic RTSV**—pass member names into a calculation script using runtime substitution variables
- **Calc: Zigzag Calculation**—learn how Essbase performs complex calculations across a time dimension
- **Calc: CalcTuple Tuple**—optimize asymmetric grid calculations across dimensions
- **Drill Through: Drillthrough Basic**—drill through to external sources to analyze data outside of the cube
- **Filters: Efficient Filters**—design and use variable data-access filters
- **MDX: AllocationMDX Insert**—allocate and insert missing values
- **Partitions: Realtime CSV Updates**— access real-time data
- **Solve Order: UnitPrice SolveOrder**—use and understand solve order in a hybrid mode cube
- **Solve Order: Solve Order Performance**—compare query performance using dynamic calculations versus using stored members and a calculation script
- **Table Format**— build Essbase cubes from tabular data
- **UDA: Flip Sign**— learn how to flip signs of data values during a data load to meet reporting requirements

System Performance Templates

System performance templates monitor system status for optimization purposes.

The Health and Performance Analyzer helps you monitor usage and performance statistics of your Essbase applications.

The Analyzer enables you to scan Essbase logs. After parsing the data, it compiles a csv Excel worksheet, optionally on the time interval that you set in **Settings**. Then, you can use the csv files to build charts and other displays.

Access Files and Artifacts

Your access to the Files catalog in Essbase depends on your user role and application-level permissions.

You can access the Files catalog from Cube Designer or from the Essbase web interface.

If your user role in Essbase is **User** with no application permissions, you can access the `shared`, `users`, and `gallery` folders. The `applications` folder is empty.

The `gallery` folder is read-only for all users.

The `shared` folder is read-write for all users.

Within the `users` folder, users have read-write access to their own folders, and the service administrator has access to all.

If your role is **User** and you have Database Access or Database Update permission for a particular application, you can additionally view (and download from) the appropriate subdirectories beneath the `applications` folder. These subdirectories contain files and artifacts for applications and cubes you can access.

If your role is **User** and you have Database Manager permission for an application, you can additionally upload files and artifacts to the cube directory, as well as delete, copy, and rename them.

If your role is **User** and you have Application Manager permission for an application, you can do everything with files that the Database Manager can do, and your access is expanded to the application directory in addition to the cube directory.

If you are a Power User, you have the same access to files and artifacts that an Application Manager has, for applications you created. Your access to other applications is restricted according to the application permission you have been granted.

A service administrator has full access to all files and directories (except the `gallery` folder, which is read-only).

Explore the Application Directories

The `applications` directories in the Files catalog contain artifacts associated with using Essbase applications.

For each application that someone creates or imports, Essbase creates a new folder inside the `applications` folder in the Files catalog. The application folder contains the cube folder, and the cube folder contains cube artifacts.

Artifacts are files related to working with Essbase applications and cubes. Artifacts have various purposes, such as defining calculations or reports. Artifacts pertaining to a cube are stored, by default, in a folder associated with the cube -- also known as the database directory.

Common cube artifacts include:

- Text files of data or metadata that can be loaded to the cube (`.txt`, `.csv`)

- Rules files for loading data and building dimensions (.rul)
- Calculation scripts that define how to calculate data (.csc)
- Application workbooks and other Excel files (.xlsx)
- MDX scripts (.mdx)
- Stored metadata about the cube (.xml)

Work with Files and Artifacts

Depending on your level of access defined in Essbase, you can perform file operations on folders and artifacts in the Files catalog.

This topic describes working with files and artifacts using the Essbase web interface, but you can also work with files from Cube Designer or the Command Line Interface (CLI).

To upload an artifact,

1. Navigate to a directory for which you have write access.
2. Optionally, click **Create Folder** to add a subdirectory (available for `shared` and `user` directories only).
3. Click **Upload Files**.
4. Drag and drop, or select a file from the file system.
5. Click **Close**.

To download an artifact,

1. Navigate to a directory for which you have read access.
2. From the **Actions** menu to the right of the file, select **Download**.

To copy an artifact,

1. Navigate to a directory for which you have read access.
2. From the **Actions** menu to the right of the file, select **Copy**.
3. Navigate to another folder for which you have write access.
4. Click **Paste**.

To rename an artifact,

1. Navigate to a directory for which you have write access.
2. From the **Actions** menu to the right of the file, select **Rename**.
3. Enter a new file name, omitting the extension.

To move an artifact,

1. Navigate to a directory for which you have write access.
2. From the **Actions** menu to the right of the file, select **Cut**.
3. Navigate to a new directory for which you have write access.
4. Click **Paste**.

To delete an artifact,

1. Navigate to a directory for which you have write access.
2. From the **Actions** menu to the right of the file, select **Delete**.
3. Click OK to confirm that you want to delete.

4

Understand Your Access Permissions in Essbase

How you work with Essbase depends on your user role and application-level permissions.

In Essbase, there are three user roles:

- [User](#)
- [Power User](#)
- [Service Administrator](#)

The majority of Essbase users have **User** role. **Power User** and **Service Administrator** roles are reserved for those who require permission to author and maintain applications. Users with **User** role are granted application-level permissions that distinguish their access to data and permissions in each application.

User Role

If your user role in Essbase is **User** with no application permissions, you can use the Files catalog (specifically, the `shared`, `users`, and `gallery` folders), download desktop tools from the Console, and explore the Academy to learn more about Essbase.

You must be granted additional access to applications by **Power Users** or **Service Administrators**. Applications are structures that contain one or more cubes, also known as databases. You can see only applications and cubes for which you have been granted application permissions.

You can have a unique application permission for each application on the server. Application permissions, from least privileged to highest, are:

- None (no application permission has been granted)
- [Database Access](#)
- [Database Update](#)
- [Database Manager](#)
- [Application Manager](#)

Database Access Permission

If your user role in Essbase is **User** and you have Database Access permission for a particular application, you can view data and metadata in the cubes within the application.

Your ability to view data and metadata may be limited in areas that are restricted by filters. You may be able to update values in some or all areas of the cube, if someone has granted you write access using a filter. You can use drill through reports, if any

exist, to access sources of data outside the cube, as long as a filter does not restrict your access to the cells within the drillable region.

With Database Access permission, you can also view the cube outline, and download files and artifacts from the application and cube directories. Job types you can run include building aggregations (if the cube is an aggregate storage cube), and running MDX scripts. Using the Console, you can view database size and monitor your own sessions.

If you are a scenario participant, you can view base data as well as scenario changes, and if you are a scenario approver, you can approve or reject the scenario.

Database Update Permission

If your user role in Essbase is **User** and you have Database Update permission for a particular application, you can make updates to the cubes within the application.

With Database Update permission for a particular application, you can do everything that a user with Database Access permission can do. Jobs you can run include loading, updating, and clearing data in the cube. You can export the cube data to tabular format. You can run any calculation scripts that you have been granted permission to execute. You can create, manage, and delete your own scenarios in block storage cubes that are enabled for scenario management.

Database Manager Permission

If your user role in Essbase is **User** and you have Database Manager permission for a particular application, you can manage the cubes within the application.

With Database Manager permission for an application, you can do everything that a user with Database Update permission can do. Additionally, you can upload files to the cube directory, edit the cube outline, export the cube to an application workbook, and start/stop the cube using the web interface. Job types you can run include building dimensions, exporting data, and exporting the cube to a workbook.

As a Database Manager, you also have access to the database inspector, which gives you control of even more cube operations. To open the database inspector from the web interface, start with the Applications page, and expand the application. From the **Actions** menu to the right of the name of the cube you want to manage, click **Inspect** to launch the inspector.

Using the database inspector, you can:

- Enable scenarios or change the number of scenarios allowed
- Manage dimensions, including generation and level names
- Access and manage files related to the database
- Create and edit calculation scripts, drill through reports, MaxL scripts, MDX scripts, report scripts, and rules files for dimension building and data loading
- Assign users permissions to execute calculation scripts
- Create and assign filters to grant or restrict data access for specific users and groups. You can assign filters, for your cube, to any users or groups who are already provisioned to use the application (an Application Manager or higher must provision users).

- Manage cube-level substitution variables
- View locked cube objects and data blocks
- View and change database settings
- View database statistics
- View and export audit records from the web interface

Application Manager Permission

If your user role in Essbase is **User** and you have Application Manager permission for a particular application, you can manage the application and cubes.

With Application Manager permission for a particular application, you can do everything that a user with Database Manager permission can do, for all cubes in the application. Additionally, you can make copies of any cubes within the application. You can copy or delete the application if you are the owner (the power user who created it), and you can delete any of the cubes in the application, if you are the cube owner (the power user who created it). You can start/stop the application using the web interface, and you can view and terminate user sessions in the Console. Job types you can run include running MaxL scripts, and using Export LCM to back up cube artifacts to a zip file.

Using the database inspector, you can manage cubes in your application the same way that a Database Manager can, and additionally, you can purge audit records for cubes.

As an Application Manager, you also have access to the application inspector, which gives you control of even more operations. To open the application inspector from the web interface, start with the Applications page. From the **Actions** menu to the right of the name of the application that you manage, click **Inspect** to launch the inspector.

Using the application inspector, you can:

- Access and manage files related to the application
- Manage application-level connections and Datasources for access to external sources of data
- Change application configuration settings
- Provision and manage user and group permissions for the application and its cubes
- Add and remove application-level substitution variables
- Change general application settings
- View application statistics
- Download application logs

Power User Role

Power User is a special user role that enables you to create applications on an Essbase service.

If you are a power user, you are automatically granted Application Manager privilege for applications you created. Your options for creating applications and cubes include

creating them from scratch in the Applications page of the web interface, importing from an application workbook, building from Cube Designer, and using the **LCM Import** job (or the `lcmimport` CLI command).

You can delete and copy applications that you created.

As a power user, you can be assigned permission to work on applications that you did not create. If your assigned permission is lower than Application Manager, then your actions are restricted to the actions permitted for the application permission you were assigned. For example, if you are assigned Database Manager permission to an application created by another power user, then your access is restricted to what a User with Database Manager permission can do.

Service Administrator Role

A **Service Administrator** has unlimited access to Essbase.

If you are a service administrator, you can do everything that power users and Application Managers can do, for all applications and cubes. Additionally, you can manage users and groups, using the Security page in the web interface. From the **Analyze** view for any cube, you can execute MDX reports impersonating other users (using **Execute As**) to test their access.

From the Console, you can manage connections and Datasources at the server level, configure e-mail settings for scenario management, and manage logs, the antivirus scanner, all user sessions, and system configuration. You can also view statistics for all databases, add and remove global substitution variables, access Performance Analyzer to monitor service usage and performance, and view/change any service-level settings.

Unlike Power User, the Service Administrator role cannot be restricted. Service administrators always have full access to all applications and cubes on the Essbase server.

About Filters

Filters control security access to data values in a cube. Filters are the most granular form of security available.

When you create a filter, you designate a set of restrictions on particular cube cells or on a range of cells. You can then assign the filter to users or groups.

Your own security role determines if you can create, assign, edit, copy, rename, or delete filters:

- If you have the Application Manager role, then you can manage any filter for any user or group. Filters do not affect you.
- If you have the Database Update role, then you can manage filters for the applications that you created.
- If you have the Database Manager role, then you can manage filters within your applications or cubes.
- If you have the Database Access role (default), then you have read access to data values in all cells, unless your access is further restricted by filters.

Create Filters

You can create multiple filters for a cube. If you edit a filter, modifications made to its definition are inherited by all users of that filter.

See Controlling Access to Database Cells Using Security Filters.

1. On the Applications home page, expand the application.
2. From the Actions menu, to the right of the cube name, launch the inspector.
3. Select the **Filters** tab.
4. Click Add +.
5. Enter a filter name in the **Filter Name** text box.
6. In the Filter Editor, click Add +
7. Under **Access**, click and use the drop-down menu to select an access level.
 - None: No data can be retrieved or updated
 - Read: Data can be retrieved but not updated
 - Write: Data can be retrieved and updated
 - MetaRead: Metadata (dimension and member names) can be retrieved and updated

The MetaRead access level overrides all other access levels. Additional data filters are enforced within existing MetaRead filters. Filtering on member combinations (using AND relationships) does not apply to MetaRead. MetaRead filters each member separately (using an OR relationship).

8. Select the row under **Member Specification** and enter member names.

You can filter members separately, or you can filter member combinations. Specify dimension or member names, alias names, member combinations, member sets that are defined by functions, or substitution variable names, which are preceded by an ampersand (&). Separate multiple entries with commas.
9. Create additional rows for the filter as needed.

If filter rows overlap or conflict, more detailed cube area specifications apply over less detailed, and more permissive access rights apply over less permissive. For example, if you give a user Read access to Actual and Write access to Jan, then the user would have Write access to Jan Actual.
10. Click **Validate** to ensure that the filter is valid.
11. Click **Save**.

On the Filters tab in the inspector, you can edit a filter by clicking the filter name and making your changes in the Filter Editor.

You can copy, rename, or delete a filter by clicking the Actions menu to the right of the filter name and choosing an option.

Create Efficient Dynamic Filters

You can create dynamic filters based on external source data to reduce the number of filter definitions needed.

Instead of managing a set of hard-coded data-access filters for many users, you can filter access to cube cells from external source data, based on member and user names.

You do this using dynamic filter definition syntax, including the method `@datasourceLookup` and the variables `$LoginUser` and `$LoginGroup`. Your external source data is a csv file or a relational table. For relational source data, you can load the .csv to a relational table.

- [Dynamic Filter Syntax](#)
- [Workflow to Create Dynamic Filters](#)
- [Example of a Dynamic Filter](#)

Dynamic Filter Syntax

Use dynamic filter syntax to create flexible filters you can assign to multiple users and groups.

Filter rows can contain the following elements as part of their definition, in addition to member expressions.

\$loginuser

This variable stores the value of the current logged in user at runtime. It can be used in conjunction with the `@datasourceLookup` method.

\$loggingroup

This variable stores the value of all the groups that current logged-in user belongs to. It includes both direct and indirect groups. When used in conjunction with the `@datasourceLookup` method, each group is individually looked up against the Datasource.

@datasourceLookup

This method fetches records from a Datasource.

Syntax

```
@datasourceLookup (dataSourceName, columnName, columnValue,
returnColumnName)
```

Parameter	Description
<i>dataSourceName</i>	The name of the external Datasource defined in Essbase. For an application-level Datasource, prefix the name with the application name and a period.
<i>columnName</i>	The name of the Datasource column to search for a given <i>columnValue</i> .
<i>columnValue</i>	The value to search for in <i>columnName</i> .

Parameter	Description
<i>returnColumnName</i>	The name of the Datasource column from which to return a list of values.

Description

A `@datasourcelookup` call is equivalent to the following SQL query:

```
select returnColumnName from dataSourceName where columnName=columnValue
```

`@datasourcelookup` looks up the given Datasource and searches for records where *columnName* contains *columnValue*. If you specify *columnValue* as `$loginuser`, this method will search for records where *columnName* contains the name of the currently logged in user.

Essbase forms the filter definition row by combining the list elements as a comma-separated string. If any record contains special characters, spaces, or only numbers, they are enclosed in quotation marks.

Examples

Enclose the parameters within quotation marks.

The following call looks up a global Datasource, and returns a list of store names where Mary is the store manager.

```
@datasourceLookup("StoreManagersDS", "STOREMANAGER", "Mary", "STORE")
```

The following call looks up an application-level Datasource, and returns a list of store names where the currently logged in user is the store manager.

```
@datasourceLookup("Sample.StoreManagersDS", "STOREMANAGER", "$loginuser", "STORE")
```

The following call looks up an application-level Datasource, and returns a list of store names where the store department matches any of the groups to which the logged in user belongs.

```
@datasourceLookup("Sample.StoreManagersDS", "STORE_DEPARTMENT", "$loggingroup", "STORE")
```

If the logged in user belongs to 3 groups, then the above `@datasourcelookup` method returns all the matching column values for each group.

Workflow to Create Dynamic Filters

Use the following general workflow to create dynamic filters.

This dynamic filters workflow assumes you already have a cube, and have provisioned users and groups.

1. Identify a source of data, whether it is a file or a relational source.

2. Define the connection and the Datasource in Essbase, either globally or at the application level.
3. Create filters at the cube level, using the **Filters** section of the database inspector.
4. Define filter rows for each filter, using the dynamic filter syntax to employ the `$loginuser` variable, the `$loggingroup` variable, and the `@datasourcelookup` method as needed.
5. Assign the filters to users or groups.
6. If you assigned the filter to a group, assign the group to the application to be filtered, using the **Permissions** section of the application inspector.

Example of a Dynamic Filter

The following dynamic filter works with the cube called `Efficient.UserFilters`, available in the gallery as a sample template.

DSLookupFilter

Access	Member Specification
MetaRead	<code>@datasourcelookup("EFFICIENT.UserDetails","USERNAME",\$loginUser,"COUNTRY")</code>
MetaRead	<code>@datasourcelookup("EFFICIENT.UserDetails","USERNAME",\$loginUser,"BUSINESSUNIT")</code>
MetaRead	<code>@datasourcelookup("EFFICIENT.UserDetails","USERNAME",\$loginUser,"COSTCENTER")</code>

To learn how to create and apply this dynamic filter, download the workbook template, `Efficient_Filters.xlsx`, from the Technical section of the gallery, and follow the README instructions in the workbook. The gallery is available in the **Files** section of the Essbase web interface.

5

Design and Create Cubes Using Application Workbooks

You can design, create, and modify fully functional cubes using Excel-based application workbooks. You can design the cube within the application workbook, quickly import the workbook to Essbase to create a cube, load data into the cube, and calculate the cube. You can also work with application workbooks in Cube Designer, which is a Smart View extension.

- [About Application Workbooks](#)
- [Download a Sample Application Workbook](#)
- [Create a Cube from an Application Workbook](#)
- [Export a Cube to an Application Workbook](#)
- [Connect to a Cube in Smart View](#)

About Application Workbooks

Application workbooks comprise a series of worksheets, which can appear in any order, and define a cube, including cube settings and dimensional hierarchies. Optionally, you can define data worksheets to be loaded automatically when you create the cube, and calculation worksheets to be executed after you load the data. There are strict layout and syntax requirements for application workbooks, and there are many validations to ensure that workbook contents are complete and formatted correctly. If the application workbook contents are not correct, then the cube building process will not be successful.

You can modify the worksheets directly in Microsoft Excel or by using the Designer Panel.

In Japanese Excel, if you enter Kanji characters directly on the sheet, the characters are not displayed correctly. Instead, use a text editor to type the Kanji characters and then copy the content into Excel.

Essbase provides application workbook templates for creating block storage and aggregate storage applications and cubes.

- **Block Storage Sample (Stored):** Block storage application workbook. File name: `Sample_Basic.xlsx`.
- **Block Storage Sample (Dynamic):** Block storage application workbook. All non-leaf level members are dynamic. File name: `Sample_Basic_Dynamic.xlsx`.
- **Block Storage Sample (Scenario):** Block storage application workbook with scenarios enabled. All non-leaf level members are dynamic. File name: `Sample_Basic_Scenario.xlsx`.
- **Aggregate Storage Sample:** Aggregate storage application workbook. File name: `ASO_Sample.xlsx`.

- **Aggregate Storage Sample Data:** Data for the aggregate storage application workbook. File name: ASO_Sample_DATA.txt.
- **Tabular Data Sample:** Tabular data Excel file. File name: Sample_Table.xlsx.

Oracle recommends that you download a sample application workbook and examine the worksheets. See [Application Workbooks Reference](#).

Download a Sample Application Workbook

Using a sample application workbook provided in Essbase, you can quickly create sample applications and cubes. The cubes are highly portable, because they are quickly and easily imported and exported.

1. On the Applications page, select a cube and then click **Files**.
2. Decide if you want to download a sample aggregate storage application workbook, or a sample block storage application workbook:
 - a. To download a sample aggregate storage application workbook, under **All Files**, click **Gallery, Applications, Demo Samples**, and **Aggregate Storage**.
 - b. To download a sample block storage application workbook, under **All Files**, click **Gallery, Applications, Demo Samples**, and **Block Storage**.
3. From the **Actions** menu to the right of the file you want to download, select **Download**.
4. Optionally, if you download the aggregate storage application workbook, ASO_Sample.xlsx, you can also download a data file, ASO_Sample_Data.txt.
5. Save the file to a local drive.
6. Open the file and examine the worksheets to understand how you can use the workbook to create an application and cube.

Create a Cube from an Application Workbook

1. In the Essbase web interface, on the Applications page, click **Import**.
2. In the **Import** dialog box, select **File Browser** to browse to a sample application workbook you previously downloaded.

You cannot import Excel files that contain spaces in the filename.
3. Your application and cube names are populated based on the names you specified in the application workbook on the Essbase.Cube worksheet.
 - (Optional) You can change the application and cube names on this screen.
 - (Required) If an existing application in Essbase matches the name of the application you are importing, then you must ensure that the cube name is unique. For example, if the name of the application and cube in the Excel workbook is Sample Basic and Essbase already has a Sample Basic cube, then you're prompted to rename the cube.
4. (Optional) Select **Advanced Options**, which allows you to choose a build option and whether to load data and execute calculation scripts.
5. (Optional) Select **View Dimensions**, which allows you to view the mapping of workbook columns to the dimensions to be created.

6. Click **OK**.

The application and cube are listed on the Applications page.

7. To view the outline, expand the application; then click the Actions menu to the right of the cube name, and launch the outline editor.

When you import an application workbook that was created using the command-line Export Utility, some member names might be rejected. See [Review Member Names Before you Import an Application Workbook Created by the Cube Export Utility](#).

If you import an application workbook and then export the cube you created to a new application workbook, the layout of the dimension sheets in the new application workbook might differ from the original, however the new workbook functions the same as the original workbook.

Export a Cube to an Application Workbook

1. In Essbase, expand the application that contains the cube that you want to export.
2. From the Actions menu, to the right of the cube name, select **Export to Excel**.
3. On the Export to Excel dialog box:
 - Select **Export Data** if you want to export the data from the cube. How the data is exported depends on whether the cube is block storage or aggregate storage.
 - In block storage cubes, if the size of the data is 400 MB or less, it is exported to the application workbook, on the Data worksheet. If the data size exceeds 400 MB, data is exported to a flat file named *Cubename.txt*, which is included in a file named *Cubename.zip* on the **Files** page.
 - In aggregate storage cubes, regardless of the size, data is always exported to a flat file named *Cubename.txt*, which is included in a file named *Cubename.zip* on the **Files** page.
 - Select a build method, **Generation** or **Parent-Child**.
 - Select **Export Calculation Script** if you want to export each of the calculation scripts as a separate worksheet within the application workbook.
4. When prompted, save the exported application workbook to your local or network drive or download the exported application workbook and data .zip files from the **Files** page.

File names do not include spaces because files that are imported to Essbase cannot contain spaces in the file name.



If you choose the options to include data, calculation scripts, or both in an export when they do not exist in the cube, the job completes without errors, but no data or scripts are exported.

The exported application workbook can be imported to Essbase. See:

- [Create a Cube from an Application Workbook](#)
- [Create a Cube from a Local Application Workbook in Cube Designer](#)

Connect to a Cube in Smart View

In Smart View, you can create a private connection using the quick connection method, if you know the URL. The private connection URL is your Essbase login URL with the string `/smartview` appended to it.

1. From the Smart View ribbon, click **Panel**.
2. From the Smart View panel, click **Home**  and then select **Private Connections**.
3. In the text box, enter the login URL ending with `/essbase/smartview`; for example, `https://192.0.2.1:443/essbase/smartview`.
4. Click the connect arrow  .
5. On the Login dialog box, enter your Essbase user name and password, then click **Sign In**.

6

Design and Manage Cubes from Tabular Data

You can create a cube from tabular data by extracting fact tables from a relational database into an Excel file and then deploying the cube. You can also export a cube to tabular data.

Topics:

- [Transform Tabular Data to Cubes](#)
- [Create and Update a Cube from Tabular Data](#)

Transform Tabular Data to Cubes

You can create a cube from tabular data by extracting fact tables from a relational database into an Excel file and then deploying the cube.

Patterns in the relationships between column headers and data are detected to deploy a multidimensional cube. The process for transforming tabular data into a structure that can be used in a multidimensional cube include these concepts:

- Correlations between columns
- Correlations between column types (such as date, number, and text)
- Header text analysis for common prefixes and business intelligence-related terms (such as cost, price, account)
- Report structure (such as merged cells and empty cells)
- (Optional) Forced-designation headers that are used to explicitly define the shape of a cube and can include formulas to create measures dimensions.
- Measures hierarchies (which can also be generated in Transform Data in Cube Designer).

Sample tabular data Excel files are provided to demonstrate the concepts of intrinsic and forced-designation headers.

When working with tabular data, you should analyze the data before you create a cube from it. Then, after the cube is created, you should determine if the cube outline is the way you want it.

You can create a cube from tabular data in the Essbase instance or in Cube Designer. See [Create and Update a Cube from Tabular Data](#).

Use Intrinsic Headers to Transform Tabular Data to Cubes

Intrinsic headers use table.column format, which is demonstrated in the `Sample_Table.xlsx` file. In this sample file, the column headers have names such as Units, Discounts, Time.Month, Regions.Region, and Product.Brand.

The transformation process creates this hierarchy:

```

Units
Discounts
Fixed Costs
Variable Costs
Revenue
Time
    Month
    Quarter
Years
Regions
    Region
    Area
    Country
Channel
Product
    Brand
...
    
```

Use Forced Designation Headers to Transform Tabular Data Into Cubes

With forced-designation headers (hints), you can specify how tabular data should be handled during the transformation process.

For example, you can force a column to be treated as a measures or an attributes dimension. Most forced-designation headers require a keyword in brackets []. Forced-designation headers are demonstrated in the templates `Unstr_Hints.xlsx` and `Sample_Table.xlsx` templates (available in the gallery).

Supported forced-designation header formats:

Table 6-1 Forced-designation Header Formats

Designation	Header Format	Example
Dimension generation	ParentGeneration.CurrentGeneration	Category.Product
Alias	ReferenceGeneration.Generation[alias]	Year.ShortYearForm[alias]
Attribute	ReferenceGeneration.AttributeDimName[attr]	Product.Discounted[attr]
Measures	MeasureName[measure]	Price[measure]
Measure generation	Parent.child[measure] Top-most parent, if unique, is the account dimension name. If not unique, this member is auto-generated in the account dimension.	Measures.profit[measure] profit.cost[measure] cost.price[measure]

Table 6-1 (Cont.) Forced-designation Header Formats

Designation	Header Format	Example
Measures formula	MeasureName[= <i>formula_syntax</i>];	profit[="price"- <i>cost</i>]; profit[="D1"- <i>E1</i>]; price[=IF ("S1" == #MISSING) "R1"; ELSE "S1"; ENDIF;]
Measures consolidation	MeasureName[+] : add to parent MeasureName[-] : subtract from parent MeasureName[~] : no consolidation (equivalent to [measure]) The default is no consolidation.	price.shipment[+] Consolidation can be defined only for measure dim
Formula consolidation	FormulaName[+=<formula>] : add to parent FormulaName[-=<formula>] : subtract from parent	profit[+=price-cost] cost.external[+=ExternalWork +ExternalParts]
UDA	ReferenceGeneration[uda]	Product[uda]
Skip The column is not read.	ColumnName[skip]	column[skip]
Recur The last column cell value is used for empty cells Recur can be combined with other forced designations; include a comma separated list of forced designations within a bracket, ColumnName[designationA,recur].	ColumnName[recur]	Product[recur] Product[uda,recur]

You can specify columns to be measures dimensions and you can use formulas to create measures dimensions with calculated data during the transformation process. The measures and measures formula forced-designation headers are specified with the name for the measures dimension, followed by a keyword or formula that is enclosed in square brackets and appended to the measures dimension name.

You can also consolidate measures and formulas by adding them to, or subtracting from, the parent.

To specify a column to be a measures dimension, in the column header, you enter the name of the measures dimension and then append the keyword [measure]. For example, you can specify the Units and Fixed Costs columns as measures dimensions by using this syntax: Units[measure] and Fixed Costs[measure].

The transformation process creates this hierarchy, with Units, Discounts, Fixed Costs, Variable Costs, and Revenue as measures:

```

Time
  Year
    Quarter
    Month
Regions
  Region
    Area
    Country
...
Product
  Brand
...
Units
Discounts
Fixed Costs
Variable Costs
Revenue

```

You can create a measure generation hierarchy (parent.child[measure] hierarchy), in a similar way that you create regular dimension generations.

For example, to create a measure hierarchy, you enter Measures.profit[measure], profit.cost[measure] and cost.price[measure], which produces the following hierarchy:

```

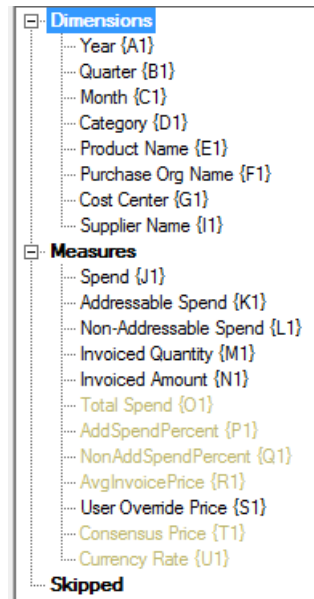
Measures
  profit
    cost
      price

```

To create measures dimensions from formulas, in the column header, you enter the name of the measures dimension and then append the formula syntax in brackets []. Within the brackets, start the formula with an equal sign (=) and end the formula with a semicolon (;). The arguments in the formula correspond to column names or cell coordinates, which must be enclosed in quotes. You can use Essbase calculation functions and commands in the formula.

Assume that you have an Excel file named Spend_Formulas.xlsx with tabular data on the SpendHistory worksheet, which has many columns. For example, there are dimensions named Year (column A) and Quarter (column B), and measures dimensions named Spend (column J) and Addressable Spend (column K). These columns have data. Then there are column headers that use formulas to create a measures dimensions. These columns do not have data. For example, to create the Total Spend dimension, the header in column O uses this Essbase formula: Measure.Total Spend[="Addressable Spend" + "Non-Addressable Spend"];. To create the AddSpendPercent dimension, the header in column P uses this Essbase formula: Measure.AddSpendPercent[="Addressable Spend"/"Total Spend";].

The transformation process creates this hierarchy:



The transformation process can also identify measures dimensions when a dimension name is duplicated. Assume that you have a column header that uses this formula, `Meas.profit[="a1-"b1"];`, which creates the Meas dimension. If, in another column header, you use the Meas dimension name as the top parent, such as `Meas.Sales`, the Sales dimension is also considered a measures dimension.

Create and Update a Cube from Tabular Data

In this workflow, you're using the sample tabular data Excel file named `Sample_Table.xlsx`, which uses intrinsic column headers. See [Transform Tabular Data to Cubes](#).

1. In the Essbase web interface, click **Files**.
2. On the Files page, click **Gallery**, then **Technical, Table Format**, and then **Sample table**.
3. From the Actions menu, next to `Sample_Table.xlsx` click **Download**.
4. Save the file to a local drive.
5. To **create** a cube: On the Applications page, click **Import**.
 - a. On the **Import** dialog box, click **File Browser** and browse to `Sample_Table.xlsx`.
 - b. On the Import Cube - Excel File dialog box, browse to `Sample_Table.xlsx`

The application and cube names are pre-populated. The application name is based on the source file name without the extension (in this example, `Sample_Table`) and the cube name is based on the worksheet name (in this example, `Sales`).

- (Optional) You can change the application and cube names on this dialog box.
- (Required) If an existing application matches the name of the application that you're importing, then you must ensure that the cube name is unique.

For example, if there is already an application named Sample_Table with a cube named Sales, then you're prompted to rename the cube.

- c. (Optional) Click **Advanced Options** to modify the cube type and the type of dimensions to be created.

You can perform these actions:

- Change the cube type. By default, cubes are set to **BSO** (block storage) with the **Hybrid BSO** option. You can keep the block storage type but remove the hybrid block storage option, or you can select the **ASO** (aggregate storage).
- Select **Enable Sandboxing**, if applicable.
- Click **Show Transformations** and, on the **Transformations** pane in the Import dialog box, enter names for the dimensions you want to rename.
- Change the dimension types.

If you make any changes, then click **OK** before proceeding.

The application and cube are listed on the Applications home page.

- d. (Optional) To view the cube outline, expand the application. From the Actions menu, to the right of the cube name, launch the outline editor.
6. To **update** a cube with new members or additional data (as an incremental load), from an Excel file: on the Applications page, click **Import**.

The tabular data must have forced designation headers, and the Excel properties must have two custom properties selected: database name and application name. Otherwise, it will use the Excel name as the application name, and sheet name as the cube name.

- a. To do the incremental load, select the file with the incremental data and load it to the cube in the application, which are specified in the Import dialog. On the Import dialog box, click **File Browser**, select the file to add, and click **Open**. A message reminds you that the cube already exists in the application.
- b. Click **Advanced Options**. For **Build Option**, select any update cube option, or keep the default, Update Cube — Retain All Data. Click **OK**.

The cube and corresponding tabular data are updated.

You can't add shared members from tabular data.

Export a Cube to Tabular Data

If you have at least database update application permission, you can export a cube from the Essbase web interface into Excel, in tabular format. This exported tabular data is organized into columns with headers that Essbase can use to deploy a new multidimensional cube.

The exported tabular data differs from data exported into an application workbook. Exported tabular data consists of data and metadata, whereas application workbooks are highly structured and contain more information about the cube, such as cube settings and dimensional hierarchies.

The cube you export must meet the following conditions:

- It must not be a scenario enabled cube.

- It must have a measures dimension, and the measures dimension must be dense.

If you export a cube containing shared members, those members are not added to the exported file.

To export a cube in tabular format:

1. In the Essbase web interface, expand the application that contains the cube that you want to export.
2. From the Actions menu, to the right of the cube name, select **Export to Table Format**.
3. Select whether to export dynamic blocks and click **OK**.

The column headers on the exported sheet are of the forced designation headers (hints) type.

You can import the tabular data file to create a new cube. See [Transform Tabular Data to Cubes](#) and [Use Forced Designation Headers to Transform Tabular Data Into Cubes](#).

7

Create and Manage Cube Outlines Using the Web Interface

An outline defines the structure of the cube through dimensions, members, attributes, and their properties. The outline structure, along with consolidation operators and formulas, determines how data is stored and calculated.

- [About Cube Outlines](#)
- [View and Edit Outline Properties for a Newly Created Cube](#)
- [Create a Sample Cube to Explore Outline Properties](#)
- [Add Dimensions and Members to Outlines](#)
- [Work with Attributes](#)
- [About Duplicate Member Names](#)
- [Set Dimension and Member Properties](#)
- [Select the Member Properties to Display in the Outline](#)
- [Name Generations and Levels](#)
- [Generate Aggregate Views Automatically](#)
- [Set Advanced Cube Properties](#)
- [Unlock Objects](#)
- [Remove Data Locks](#)

About Cube Outlines

Dimensions and members represent data hierarchies. In an outline, each dimension consists of one or more members. The members, in turn, may have child members. This ancestral rollup is called a hierarchy. Unary operators (such as +, -, *, /), assigned to each member in a hierarchy define how a child member consolidates to its parent.

View and Edit Outline Properties for a Newly Created Cube

Outline properties, in part, control the functionality available in an Essbase cube, but they also control member naming and member formatting for attribute dimensions, alias tables and text measures.

1. Log into the web interface as a power user.
2. On the Applications page, click **Create** to create a new application.
3. Give the application a unique name.
4. Name the cube.
5. (Optional) Click **Advanced Options** to select a database type, allow duplicate member names, or enable scenarios.

6. Click **OK**.
7. On the Applications page, expand the new application.
8. From the Actions menu, to the right of the cube name, select **Outline**.
9. Click **Edit**.
10. Click **Outline Properties**.

Work with General and Attribute-related Outline Properties

Outline properties-General tab shows what outline features are enabled for your cube and how they are formatted. Some fields on this tab can be changed and others cannot be changed and are for your information.

Table 7-1 General Outline Properties

Field	Description	View or Edit
Allow Duplicate Member Names	<p>Enabling a cube for duplicate member names is an option when a new application is created.</p> <p>If you migrate an on-premises Essbase application with a unique member outline to an Essbase instance, you cannot change the outline to allow duplicate members. To allow duplicate member names in your Essbase instance, convert the on-premises unique member outline to a duplicate member outline before migrating the application.</p>	This field cannot be changed and is for your information.
Typed Measures Enabled	All Essbase applications are enabled for typed measures by default.	If typed measures is disabled and you want to enable it, select True. If typed measures is enabled, you cannot change the setting and this field is for your information.
Date Format	You can change the date format if you plan to use typed measures that are dates.	Use the dropdown list to select the date format that will be displayed when you query text measures that are dates.

Table 7-2 Attribute Settings – Prefix and Suffix Format

Field	Description	View or Edit
Value	A prefix or suffix may be required for your attribute member names to support member name uniqueness. Prefix or suffix values display when attribute dimension members are included in a query.	To enable prefix or suffix values for your cube, make a selection in the Value drop-down menu. The default value of None disables all prefix or suffix options.
Format	You can define unique names by attaching a prefix or suffix to member names in Boolean, date, and numeric attribute dimensions in the outline.	After selecting a prefix or suffix Value, such as Parent, you can select the format.
Separator	Select a separator (to place between the prefix or suffix and the original name).	Options are underscore (_), pipe (), or caret (^).

Table 7-3 Boolean, Date and Numeric

Field	Description	View or Edit
True Member Name	Although your cube can contain more than one Boolean attribute dimension, all Boolean attribute dimensions will share the same value for True Member Name and False Member Name. By default, Essbase assigns member names of True and False. If you want to change these names, you must change them before you add the first Boolean attribute to your cube. Once the first Boolean attribute dimension is created, you cannot change these names.	This field can only be changed before you add the first Boolean attribute dimension to your cube.

Table 7-3 (Cont.) Boolean, Date and Numeric

Field	Description	View or Edit
False Member Name	Although your cube can contain more than one Boolean attribute dimension, all Boolean attribute dimensions will share the same value for True Member Name and False Member Name. By default, Essbase assigns member names of True and False. If you want to change these names, you must change them before you add the first Boolean attribute to your cube. Once the first Boolean attribute dimension is created, you cannot change these names.	This field can only be changed before you add the first Boolean attribute dimension to your cube.
Date Member Names	You can change the format of members of date attribute dimensions.	Select Month First or Day First formatting convention for Date Member Names.
Numeric Range	Members of numeric attribute dimensions can be defined in dimension build rules to represent date ranges. Here, you can define these ranges to be Top or Bottom of Ranges. All numeric attribute dimensions built using ranges will have the same numeric range setting.	Options are Tops of Ranges and Bottoms of Ranges.

Table 7-4 Calculation Dimension Names

Field	Description	View or Edit
Name	Every Essbase cube containing attribute dimensions contains a dimension containing standard math functions that can be applied to attribute queries. You can edit the name of this dimension, and the name of each standard math function. You cannot change which math functions are automatically calculated.	Type a name for the attribute calculations dimension, if you want to change it.
Sum Member	This is a member of the attribute calculations dimension. The name to use when requesting sum data.	Type a name for the Sum member in the attribute calculations dimension, if you want to change it.

Table 7-4 (Cont.) Calculation Dimension Names

Field	Description	View or Edit
Count Member	This is a member of the attribute calculations dimension. The name to use when requesting count data.	Type a name for the Count member in the attribute calculations dimension, if you want to change it.
Minimum Member	This is a member of the attribute calculations dimension. The name to use when requesting minimum data.	Type a name for the Minimum member in the attribute calculations dimension, if you want to change it.
Maximum Member	This is a member of the attribute calculations dimension. The name to use when requesting maximum data.	Type a name for the Maximum member in the attribute calculations dimension, if you want to change it.
Average Member	This is a member of the attribute calculations dimension. The name to use when requesting average data.	Type a name for the Average member in the attribute calculations dimension, if you want to change it.

Understand and Create Alias Tables

Aliases are stored in one or more tables as part of a database outline. An alias table maps a specific, named set of alias names to member names.

To create an alias table:

1. On the Applications page, expand the application.
2. Click the **Actions** menu to the right of the cube name and click **Outline**.
3. Click **Edit**.
4. Click **Outline Properties**.
5. Select the **Aliases** tab.
6. Enter the name of the alias table you want to create and click **Add**.
You can have a maximum of 56 alias tables.
7. Click **Apply and Close**.

See [Create Aliases](#) and Setting Aliases.

You cannot delete or rename the default alias table.

Understand and Work With Dynamic Time Series Outline Properties

To dynamically calculate period-to-date values, you can enable dynamic-time-series members for an outline. You must also associate the dynamic time series member with a generation member.

You use the Dynamic Time Series tab in the Outline Properties dialog box to enable and disable dynamic time series members, to associate dynamic time series members with generations, and to specify aliases for dynamic time series members.

The **Series** column lists the eight system-defined dynamic time series members. See Using Dynamic Time Series Members Using Dynamic Time Series Members:

- H-T-D (history-to-date)
- Y-T-D (year-to-date)
- S-T-D (season-to-date)
- P-T-D (period-to-date)
- Q-T-D (quarter-to-date)
- M-T-D (month-to-date)
- W-T-D (week-to-date)
- D-T-D (day-to-date)

To enable dynamic time series members:

1. On the Applications page, expand the application.
2. Click the **Actions** menu to the right of the cube name and click **Outline**.
3. Click **Edit**.
To view outline properties, simply click **Outline Properties**. You don't need to click **Edit** first.
4. Click **Outline Properties**.
5. Click **Dynamic Time Series**.
6. Select or clear items in the **Enabled** column to enable or disable the member associated with the option.
7. In the **Generation** column, select a generation number.
You cannot associate dynamic time series members with level 0 members of the time dimension, and you should not assign a generation number to multiple members.
8. (Optional) In the **Default** column, in the member row, enter one or more aliases (one each from one or more alias tables).

Understand and Create Textual Measures Outline Properties

Text measures serve as string masks for numeric values stored in Essbase.

Since all data stored in Essbase must be numeric, text measures provide the ability for users to select text strings as input to Essbase.

For example, assume that a user is to provide an input indicating risk assessment. It might be preferable to select from a list of strings: low, medium, high. To accomplish this in Essbase, you would create a textual measure list and assign the appropriate strings to numeric values stored in the database.

To add text measures:

1. On the Applications page, expand the application.
2. Click the **Actions** menu to the right of the cube name and click **Outline**.
3. Click **Edit**.
To view outline properties, simply click **Outline Properties**. You don't need to click **Edit** first.

4. Click **Outline Properties**.
5. Click **Textual Measures**.
6. Type a text measure name and click **Add**.
7. (Optional) Select **Auto Generate IDs** to automatically generate numeric IDs for the strings.
8. (Optional) Rename the #Missing or #OutOfRange Names.
9. Click **+**.
10. Assign a numeric ID (unless you auto-generated the IDs) and a string name.
11. Repeat steps 4 & 5 until all strings are identified.
12. Click **Apply and Close**.
13. Click **Save**.

Once you have created a textual measure list, you can create a measure member on your outline and assign to it type "text."

To select a text measure for a member, select the member and then select a text measure from the **Type** drop-down menu in the properties panel to the right of the outline. All text measures are shown with the prefix, "Text."

Create a Sample Cube to Explore Outline Properties

Throughout this chapter, you will work with a copy of the Sample.Basic gallery template that you create on your server. You must be a power user to create the application.

If you aren't a power user, ask one to create an application for you and provision you as Database Manager for the application.

1. Log into the web interface as a power user.
2. On the Applications page, click **Import**.
3. Select **Catalog**.
4. Double-click **Gallery**.
5. Double-click **Applications**.
6. Double-click **Demo Samples**.
7. Double-click **Block Storage**.
8. Highlight **Sample_Basic.xlsx** and click **Select**.
9. Type a unique Application Name and click **OK**.

If the application name you choose isn't unique, you will receive an error message asking you to change the name.

For the remainder of the chapter, when we refer to *<yourapplication>*, you should use the application you just created.

Set Outline Properties in your Sample Cube

You can set outline properties in *<yourapplication>*.

1. On the Applications home page, expand *<yourapplication>*.
2. From the **Actions** menu, to the right of the cube name select **Outline**.
3. Click **Edit**.
4. Select **Outline Properties**.

Add Dimensions and Members to Outlines

The top level members of any hierarchy in an outline are called dimension names or dimensions. There are two types of dimensions: standard dimensions and attribute dimensions.

You can add dimensions and members to a cube using any of the following methods:

- Add dimensions and members manually with the outline in edit mode.
- Import an Excel file containing dimension definitions (either tabular data or an application workbook).
- Build dimensions using a datasource and rule file.

In this chapter, we focus on manual outline updates.

Add Dimensions to Outlines Manually

In block storage or partial hybrid mode cubes (which have one or more stored dimensions), if you add, delete, or move members in dimensions and then save the outline, then the cube is restructured.

After restructuring is complete, recalculate the data. Aggregate storage and fully hybrid mode cubes do not need to be recalculated because they are dynamic (upper level data is not stored).

If you add a dimension that is virtual (dynamic calc or label only), then any data existing in the cube is stored with the first level-0 stored member in the new dimension. There must be at least one stored member in the hierarchy.

Dimension names must always be unique in the outline, even if the outline allows duplicate member names. To add a dimension to an outline:

1. On the Applications page, expand *<yourapplication>*.
2. Click the **Actions** to the right of the cube name and then choose **Outline**.
3. Click **Unlock**. This is only needed if the outline is locked. Otherwise, proceed to step 4.
4. Click **Edit** and then select a dimension.
5. On the outline toolbar, under **Actions**, select **Add a sibling below the selected member**.
6. Enter a name for the new dimension and press Tab.
Use no more than 1024 characters when naming dimensions, members, or aliases.
7. On the outline toolbar, under **Actions**, select **Display member properties panel on the right side** to open the properties pane, and select the properties that you want for the new dimension.

8. Click **Save**.

Add Members to Outlines Manually

Unless the cube is enabled for duplicate member names, each member has a unique name.

1. On the Applications page, expand *<yourapplication>*.
2. From the **Actions** menu, to the right of the cube name, select **Outline**.
3. Click **Edit**.
4. To view and select lower-level members in a dimension, drill down in the dimension by expanding the dimension name and subsequent member names.
5. When you reach the member to which you want to add a child or sibling member, select it.
6. From the outline toolbar, under **Actions**, select **Add a sibling above the selected member**, **Add a sibling below the selected member** or **Add a child to the selected member**.
7. Enter the name for the new member and press Tab.
Use no more than 1024 characters when naming dimensions, members, or aliases.
8. On the outline toolbar, under **Actions**, select **Display member properties panel on the right side** to open the properties pane, and select the properties that you want for the new member.
9. Click **Save**.

Work with Attributes

Attributes describe characteristics of data, such as the size and color of products. You can use attributes to group and analyze members of dimensions based on their characteristics. For example, you can analyze product profitability based on size or packaging, and you can make more effective conclusions by incorporating market attributes, such as the population size of each market region, into your analysis.

When manually working with attributes, use the outline editor and the Attributes tab in the outline inspector.

Workflow for manually building attribute dimensions:

1. Create attribute dimensions.
2. Tag the dimensions as attribute dimensions and set the attribute dimension type (text, numeric, Boolean, or date).
Use the outline inspector, general tab to set the dimension as an attribute dimension, and to set the attribute dimension type.
3. Add members to attribute dimensions.
4. Associate a standard dimension with an attribute dimension, thereby defining the base dimension of the attribute dimension. Use the **Attributes** tab in the outline inspector to associate an attribute dimension to a base dimension.

When creating an attribute dimension, by default, a base dimension is associated with the newly created attribute dimension. The associated base dimension is either a newly created last sparse dimension or the last existing sparse dimension.

For example, if you create two sparse dimensions, dim1 and dim2, and then create an attribute dimension attr1, attr1 is associated with dim2 (the last sparse dimension that was created). If no sparse dimension was created recently, attr1 is associated with the last sparse dimension.

See *Working with Attributes*.

About Duplicate Member Names

When you create a cube, you can specify that duplicate (non-unique) member names and aliases are allowed in a cube outline, with some restrictions.

1. From the web interface, log in as a power user, and click **Create**.
2. Enter a unique application name and any cube name.
3. Expand **Advanced Options** and select **Allow Duplicate Member Names**.
4. Click **OK**.

A duplicate member outline might, for example have a Market dimension and require two members named New York: one as a child member of the dimension parent member, Market, and one as a child of the member, New York. The member names are displayed as New York. The qualified member names are:

- [Market].[New York]
- [Market].[New York].[New York]

To add a duplicate member name, enter the duplicate member in the outline. There are no additional requirements for adding a duplicate member.

1. On the Applications page, expand the application you just created.
2. Click the **Actions** menu for that cube and then choose **Outline**.
3. Click **Edit**.
4. Type **Market** and press **Tab**.
5. From the outline toolbar, under **Actions**, select **Add child to the selected member**.
6. Type **New York** and press **Tab**.
7. From the outline toolbar, under **Actions**, select **Add child to the selected member**.
8. Type **New York** and press **Tab**.
9. Highlight the last member you created and, on the outline toolbar, under **Actions**, select **Display member properties panel on the right side** to open the properties pane.
10. Expand **Name** and look at the **Name** and **Path**.
Notice that the fully qualified member name of [Market].[New York].[New York] is displayed, but the outline member is called New York.

Duplicate Naming Restrictions:

- If the outline is not enabled for duplicate members, then an error is returned when a duplicate member name is entered.
- Dimension names, generation names, and level names must always be unique, and sibling members under a parent member must always be unique.
- You must enable duplicate member names at the time you create the application. You cannot convert a unique member outline to a duplicate member outline.
- Duplicate member names applies to the entire outline and cannot be assigned only to a single dimension, for example.
- After you migrate an on-premises cube with a unique member outline to a cloud service instance, you cannot change the outline to allow duplicate members. If you want your on-premises cube to allow duplicate members in your cloud service instance, you must convert the unique member outline to a duplicate member outline in your on-premises installation before migrating the application to a cloud service instance.

Set Dimension and Member Properties

To set dimension and member properties, open the outline in edit mode.

Once in edit mode, you can set dimension and member properties:

- Inline, by double-clicking on a member name or in a column next to a member name in the outline.
- In the properties panel, by highlighting a member and, on the outline toolbar, under **Actions**, selecting **Display member properties panel on the right side**.
- On the outline toolbar, by highlighting a member and selecting the options that you want on the toolbar.

Open the Outline in Edit Mode

Before you can change or set member properties, you need to open the outline in Edit mode.

1. From the Applications page, expand *<yourapplication>*.
2. Click the **Actions** menu to the right of the cube name and select **Outline**.
3. Click **Edit**.

Set Member Properties while in Edit Mode

With the outline in Edit mode, you can set properties for individual members. Using the keyboard or the member inspector, you can make changes quickly and efficiently.

To enable inline editing, double click on a member or in one of the columns to the right of the member name in the outline. For example, if you click along a row for a member you want to edit in the Data Storage Type column, you can use a menu to select a storage type for the highlighted member. If you double-click in the formula column, you can type a member formula.

With inline editing enabled you can:

- Type member names, or rename existing members.

- Use the Tab key to move from left to right between columns.
- Use the Enter key to move down in the outline tree.
- Use the space bar to expand menus, and use the up and down arrows to navigate the menu items.

You can also select multiple rows and change member properties in all selected rows at one time. For example, you can select several rows and change the member consolidation to + by clicking the + sign on the toolbar.

Set Properties in the Member Inspector

You can view and set member properties in the member inspector.

To open the Member Inspector:

1. On the Applications page, expand the application.
2. Click the **Actions** menu to the right of the cube name and select **Outline**.
3. Click **Edit**.
4. Drill into the outline to find the member you want to update and select it.
5. Right click and select **Inspect**.
6. In the Member Inspector, choose a tab on which to make your modifications:
 - **General**
 - **Aliases**
 - **Formula**
 - **Attributes**
 - **User-defined Attributes**

See Setting Dimension and Member Properties.

Set General Properties

On the General tab, you can view or modify basic dimension or member information (such as consolidation properties, storage properties, and comments).

The options available on the tab vary, depending on the outline type, and the dimension and member type. For example, the items available vary depending on whether the cube is block storage or aggregate storage, or whether you selected a dimension name or a member within a dimension.

The following is a partial list of properties. For background information on the various properties, see *Designing and Maintaining Essbase Cubes*.

Table 7-5 Dimension and Member General Properties

Field Name	Description	Applies to...
Name	Enter a dimension or member name. Use no more than 1024 bytes when naming dimensions, members, or aliases.	<ul style="list-style-type: none"> Aggregate storage dimensions and members Block storage dimensions and members
Comment	Enter a comment. Comments can contain up to 255 characters.	<ul style="list-style-type: none"> Aggregate storage dimensions and members Block storage dimensions and members
Dimension type	For a dimension within an aggregate storage outline, select: <ul style="list-style-type: none"> None Accounts Time Attribute 	<ul style="list-style-type: none"> Aggregate storage dimensions Block storage dimensions
Consolidation	For a member that is not a dimension or an attribute, select a consolidation operator: <ul style="list-style-type: none"> + (addition) - (subtraction) * (multiplication) / (division) % (percentage) ~ (ignore) ^ (nonconsolidating) Addition (+) is the default. The ^ (nonconsolidating) operator applies only to block storage cubes.	<ul style="list-style-type: none"> Aggregate storage members Block storage members
Two-Pass	Select the Two-Pass calc check box to calculate the member during a second pass through the outline.	<ul style="list-style-type: none"> Block storage stored members For dynamic members, set solve order instead
Data Storage	Select an option to determine how data values for the current dimension or member are stored: <ul style="list-style-type: none"> Store data Dynamic calc (This option does not apply to aggregate storage cubes.) Never share Label only Shared member 	<ul style="list-style-type: none"> Aggregate storage dimensions and members Block storage dimensions and members

Table 7-5 (Cont.) Dimension and Member General Properties

Field Name	Description	Applies to...
Member solve order	Specify a solve order between 0 and 127 to indicate the priority in which the member is calculated.	<ul style="list-style-type: none"> Aggregate storage members Dynamic block storage members
Hierarchy	<p>Specify Stored (the default) or Dynamic or, for a dimension within an aggregate storage outline, select the Multiple hierarchy enabled option (which equates to selecting both Stored and Dynamic).</p> <p>The storage option that you select is applied to the hierarchy headed by the dimension or generation 2 member.</p>	<ul style="list-style-type: none"> Aggregate storage dimensions Generation 2 aggregate storage members
Level Usage for Aggregation	<p>Select one of these options to provide a way for an administrator to influence both default and query-based view selection:</p> <ul style="list-style-type: none"> Default: Internal mechanisms decide how to create aggregations. No aggregation: Aggregation is not performed along this hierarchy. All views selected are at the input level. Top level only: (Applies to primary hierarchies.) Queries are answered directly from input data. No intermediate levels: (Applies to primary hierarchies.) This selects top and bottom levels only. 	Aggregate storage dimensions

Table 7-5 (Cont.) Dimension and Member General Properties

Field Name	Description	Applies to...
Variance reporting expense	<p>Members from the dimension tagged as type Accounts can have an Expense property value of True or False. When @VAR or @VARPER formulas are evaluated, Account members whose expense property is False will have opposite sign to those whose expense property is True.</p> <p>Example: Scenario dimension member Variance with formula @VAR(Actual, Budget). For Account dimension member Sales [with Expense property False], Variance member will be calculated as Actual-Budget. For Account dimension member COGS [with Expense property True], Variance member will be calculated as Budget-Actual.</p>	Block storage accounts dimension and members

Table 7-5 (Cont.) Dimension and Member General Properties

Field Name	Description	Applies to...
Account information	<p>Time Balance: To use time balance properties, you must have a dimension tagged as Accounts and a dimension tagged as Time.</p> <ul style="list-style-type: none"> • None: Apply no time balance property. Member values are calculated in the default manner. • Average: A parent value represents the average value of a time period. • First: A parent value represents the value at the beginning of a time period. • Last: A parent value represents the value at the end of a time period. <p>Skip option: Select an option (None or Missing) to determine what values are ignored during time balance calculations. If you select None, then no values are ignored, and, if you select Missing, then #MISSING values are ignored. You can specify skip settings only if the time balance property is set as first, last, or average.</p> <ul style="list-style-type: none"> • None • Missing <p>You can set these properties for any members except Label Only members.</p>	Block storage Accounts dimension only

Create Aliases

On the Aliases tab, you can assign alternate names, or aliases, to a dimension, member, or shared member. For example, in the <yourapplication>.Basic cube outline, members of the Product dimension are identified by product codes, such as 100, and by descriptive aliases, such as Cola.

1. On the Applications page, expand the application.
2. From the **Actions** menu, to the right of the cube name, select **Outline**.
3. Click **Edit**.
4. Drill into the outline to find the member you want to update and select it.
5. Right click and select **Inspect**.

6. Click **Aliases**.
7. In the field for the alias table you want to use, enter the value of the alias.
8. Click **Apply and Close**.
9. Click **Save**.

See [Understand and Create Alias Tables](#) and Setting Aliases.

Create Member Formulas

On the Formula tab of the Member Inspector, you can create and edit member formulas for both block storage and aggregate storage cubes. These formulas are calculated through default cube calculations and calculation-script calculations.

You can construct block storage member formulas from operators, functions, dimension names, member names, substitution variables, and numeric constants. To write formulas for block storage outlines, a set of calculation functions and operators is provided. For syntax and examples, see Calculation Functions.

Aggregate storage member formulas cannot be created using Calculator language. Instead, create them using Multidimensional Expression Language (MDX).

Let's create an example member formula. Suppose you have a dynamic calc member called "Watchlist Products" and you want it to be the sum of products "100-10", "200-10" and "300-10."

1. On the Applications page, expand *<yourapplication>* and select the Basic cube.
2. Click the **Actions** menu and select **Outline**.
3. Click **Edit**.
4. Select the Product dimension, add a child called Watchlist_Products, and press the Tab key.
5. Right click on Watchlist_Products and select **Inspect**.
6. Select the **Formula** tab.
7. In the member tree, in the left panel of the Formula Editor, drill into Product to find the first product member to add to your formula, "100-10." Right click the member name and click **Insert Name** to insert it into your formula.
8. Place the cursor after "100-10" and press the + key.
9. Use the member tree to pick the next product member to insert, 200-10. Right click the member name and click **Insert Name** to insert it into your formula.
10. Repeat for the last product member, 300-10 and put a semi-colon (;) at the end of the formula.
The formula should look like this: "100-10"+"200-10"+"300-10";
11. Click **Verify** and fix any errors.
12. Click **Apply and Close**.
13. In the Data Storage Type column for Watchlist_Products, select **Dynamic Calculation**.
14. Click **Save** to save the outline.

Member formulas like the one you just created can also include Essbase functions. When using Essbase functions in member formulas, use the **Function Name** menu on

the right side of the formula editor to find and add calculation functions the script. See the Function description under the menu to read descriptions of each function.

See Developing Formulas for Block Storage Databases.

To write formulas for block storage outlines, a set of calculation functions and operators, known as the Calculator, or Calc, language, is provided. For descriptions of calculation commands and functions, see Calculation Commands and Calculation Functions.

Aggregate storage member formulas cannot be created using Calculator language. Instead, create them using Multidimensional Expression Language (MDX). See Aggregate Storage and MDX Outline Formulas and Developing Formulas on Aggregate Storage Outlines.

Set Attribute Associations

When manually working with attributes, use the outline editor and the Attributes tab in the member inspector. First you associate attribute dimensions with base dimensions and then you associate attribute members with members of the base dimension.

Attributes are associated with Base dimensions; base dimensions are sparse standard dimensions containing members with which you would like to associate attributes.

Associate an Attribute Dimension with a Base Dimension

To associate an attribute dimension in *<yourapplication>* with a base dimension:

1. On the Applications page, expand *<yourapplication>*.
2. From the **Actions** menu to the right of the cube name, select **Outline**.
3. Select a base dimension to which you want to associate an attribute dimension. For this exercise, choose Market.
4. If you are not already in Edit mode, click **Edit**.
5. Right click on Market and select **Inspect**.
6. Click **Attributes**.
7. Select an attribute dimension, Intro Date from the **Attribute Name** column.
8. Click the right arrow next to **Associated Attributes** to associate the selected attribute to the regular dimension you selected in step 4.
9. Click **Apply**.
10. Click **Close**.
11. Click **Save** to save the outline.

After you associate an attribute dimension with a base dimension, you must associate members of the attribute dimension with members of the base dimension; these members must all be from the same level in the base dimension.

Associate an Attribute Member with a Member of the Base Dimension

To associate an attribute member in *<yourapplication>* with a member of a base dimension:

1. With the *<yourapplication>* outline still open, click **Edit**.

2. Expand Market, then East and select New York.
New York is the base member to which we'll associate an attribute.
3. Right click on New York and select **Inspect**.
4. Select **Attributes**.
5. From the member tree, expand **Population** and select the attribute member you want to associate to New York.
6. Click **Apply and Close**.
7. Click **Save** to save the outline.

See Working with Attributes.

Create User-Defined Attributes

On the User-defined Attributes tab, you can create, assign, and unassign user-defined attributes (UDAs). A UDA is a word or phrase that describes the member. For example, you might create a UDA called Major Market and assign it to all members in the outline that are part of a major market, as defined by your organization.

Like attributes, UDAs are used to filter data retrievals. Unlike attributes, UDAs have no built-in calculation functionality. However, UDAs can be assigned to dense and sparse dimensions, whereas attributes can be assigned to only sparse dimensions. Also, a UDA can be assigned to any level or generation in a dimension.

1. On the applications page, expand *<yourapplication>*.
2. Click the **Actions** menu to the right of the cube name and select **Outline**.
3. Click **Edit**.
4. Highlight a member to which you would like to assign a UDA.
5. Right click the member and select **Inspect**.
6. Click the User-defined Attributes tab.
7. In the **User-defined Attributes** field, enter a UDA name and press the Enter key.
8. Click **Apply and Close** to create the UDA for the dimension and assign the new UDA to the member.
9. Click **Save** to save the outline.

Select the Member Properties to Display in the Outline

You can customize which member properties to display in the outline.

1. On the Applications page, expand the application.
2. From the **Actions** menu, to the right of the cube name, select **Outline**.
3. On the outline toolbar, under **Inspect**, select **Display selected columns in the table**.
4. In the **Select the member properties to display** dialog box, clear the check box next to **Property name** to deselect all the properties.
5. Select the properties that you want to display in the outline.
6. Press **Apply and Close**.

Only the selected properties are displayed in the outline.

Name Generations and Levels

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

Use generation and level names in calculation scripts wherever you need to specify either a list of member names or a list of generation or level numbers. For example, you can limit a calculation in a calculation script to the members of a specific generation.

Data Visualization displays generation names, while in Smart View, you use dimension names for browsing.

1. On the Applications page, expand *<yourapplication>*.
2. From the **Actions** menu, to the right of the cube name, click **Inspect**.
3. In the inspector, select the **Dimensions** tab.
4. On the **Dimensions** tab, select the dimension in which you want to name generations or levels.
5. Click a generation or level name to enable editing of that field.
6. Enter a generation or level name.

<yourapplication> already has generation and level names, but you can change them if you want to.

7. Click **Save**.

Generate Aggregate Views Automatically

You can create aggregate views to be generated automatically, according to the dimension structure in an outline for an aggregate storage cube.

If you add the application configuration setting `DEFAULTVIEWBUILD` and set it to `TRUE`, aggregation views will be generated automatically after each data load to an empty cube, after changes to metadata that require redesign or rebuild of aggregate views, or on-demand (when the execute aggregate selection `MaxL` statement is run).

When you use automatically generated aggregation views, query performance can improve. It may impact data load time and increase the amount of disk space used by data.

To control the size of the resulting aggregate views, add the additional application configuration setting `DEFAULTVIEWBUILDSIZE`, and set its value to the desired total size ratio. For example, `DEFAULTVIEWBUILDSIZE AsoSamp 1.2` limits the resulting growth of the aggregated cube to no more than 20% of its size prior to the aggregation.

Set Advanced Cube Properties


If the current cube is a block storage cube, then you can select whether to enable the following options:

- **Aggregate missing values:** If you never load data at parent levels, selecting this option may improve calculation performance. If this option is selected and you load data at the parent level, then the parent-level values are replaced by the results of the cube consolidation, even if the results are #MISSING values.
 - **Create blocks on equations:** If this option is selected, then when you assign a non-constant value to a member combination for which no data block exists, a data block is created. Selecting this option can produce a very large cube.
 - **Two-Pass calculation:** If this option is selected, then after a default calculation, members that are tagged as two-pass are recalculated.
1. On the Applications page, expand the application.
 2. From the **Actions** menu, to the right of the cube name, click **Inspect**.
 3. Select the Settings tab.
 4. Select **Calculation**.
 5. Select the options that you want.
 6. Click **Save**.

Unlock Objects

Essbase uses a checkout facility for cube objects (such as calculation scripts and rules files). Objects are locked automatically when they are in use and the locks are deleted when they are no longer in use.

You can view and unlock objects, according to your security role. Users with the Service Admin role can unlock any object. Users without the Service Admin role can unlock only those objects that they locked.


1. On the Applications page, expand the application.
2. From the **Actions** menu, to the right of the cube name, click **Inspect**.
3. Select **Locks**.
4. From the Display menu, select **Objects**.
5. Select the object you want to unlock and click **Unlock**  .

Remove Data Locks

Data locks apply to block storage cubes only.

Occasionally, you may need to release a lock that you created in the cube, generally from a Smart View Submit Data action. For example, if you're calculating a cube that has active locks on data, and the calculation encounters a lock, then the calculation must wait. If you release the lock, the calculation can resume.

You can always unlock data that you locked. To remove another user's data locks, you must have the Application Manager or Database Manager role.

1. On the Applications home page, expand the application.
2. From the **Actions** menu, to the right of the cube name, click **Inspect**.
3. Select the **Locks** tab.
4. From the Display menu, select **Blocks**.
5. Select the lock and click **Unlock** .

8

Use Connections and Datasources

Using saved connections and Datasources, you can set up cubes to interact easily with a variety of source data.

For example, you can set up a partition between a cube and RDBMS tables, share data between a cube and Oracle Database, develop security filters using variables to fetch members or user names from outside source data, and load data from REST API endpoints.

Many cube operations require connection information, such as login details, to access remote source data or hosts. You can define these as connections and Datasources once, and reuse them in various operations, so that you do not have to specify the details each time you perform a task.

You can implement saved connections and Datasources to facilitate the following operations:

- Loading dimensions and data
- Importing cubes
- Defining variable security filters
- Connecting cubes using partitions, and accessing real-time data
- Drilling through to remote sources of data

Topics in this chapter:

- [About Connections and Datasources](#)
- [Create Connections and Datasources](#)

About Connections and Datasources

Many operations call for connecting to source data external to the cube. Connections and Datasources, which you create and save as reusable objects in Oracle Essbase, provide a way to do this efficiently.

If you have network connectivity between an external source of data and Essbase, you can define connections and Datasources in Essbase to easily "pull" data from the external source. If you have no network connectivity between Essbase and the external source of data, then you should stream data loads or dimension builds using the CLI tool, first creating a local connection, and then issuing the `dataload` or `dimbuild` command with the `stream` option; this "push" method is slower.

A **connection** stores information about an external server and the login credentials that are required to access it. By defining one connection that can be used by multiple processes and artifacts, you can simplify many aspects of your analytics. For example, when it's time to change a system password, you only need to update one connection.

A **Datasource** is another object that you can define once and reuse, to help you manage data flow into and out of your cubes. You can define a Datasource to

represent any external source of data, whether a relational system, a table, a file, or another cube.

You can define one connection and use it to access multiple Datasources. For example, consider an external Oracle Database server that has separate tables for products, resellers, and sales territories. You need only one connection to access Oracle Database, but you might want to create unique Datasources to access each of the tables.

One use case in which you might define multiple Datasources per connection is as follows: if you use separate load rules to build each dimension in a cube, each rules file can be set up to access the relevant table in Oracle Database. For example, assume your cube has a Market dimension, and you regularly build dimensions using a Dim_Market load rule to populate the Market dimension from a SALES_TERRITORIES table. Likewise, you use a Dim_Product load rule to populate the Product dimension from a PRODUCT table. Both load rules can use the same connection, but because they draw from separate tables, you defined two different Datasources.

Historically, you needed to hard code connection and source data details into Essbase artifacts such as rule files, location aliases, and partitions. While hard coded information is still supported in these artifacts, you can work more efficiently if you define connections and Datasources globally (or, at the application level).

Create Connections and Datasources

Before you can create connections to external source data from Essbase, you must get the connection details such as host names, user names, passwords, and any other service credentials from your system administrator.

Topics in this section:

- [Create a Connection and Datasource to Access Oracle Database](#)
- [Create a Connection and Datasource to Access Another Cube](#)
- [Create a Connection and Datasource to Access a Data File](#)


You can also create connections and Datasources for Spark, DB2, SQL Server, and MySQL.

Create a Connection and Datasource to Access Oracle Database

Define a connection and Datasource between Essbase and Oracle Database.

1. In Essbase, on the Sources page, click **Connections**.
To define the connection and Datasource at application level, instead of globally, start on the Applications page instead of the Sources page. From the Actions menu to the right of an application name, launch the inspector and click **Sources**.
2. Click **Create Connection** and select **Oracle Database**.
3. Enter a connection name, host, port number, user name, and password. Select **SID** (server ID) or **Service**, and enter server details.

Create Connection


Oracle Database

Autonomous(Beta)

* Name

* Host

* Port

* User

* Password

SID Service

Description

4. Click **Test** to validate the connection, and if successful, click **Create**.
5. Verify that the connection was created successfully and appears in the list of connections.

Next, you will create a Datasource for the Oracle Database connection.

6. Click **Datasources**, and click **Create Datasource**.
7. From the **Connection** drop-down box, select the name of the connection you just created; for example, custDBaaS. For application-level Datasources, select the application-level connection name, in the format *appName.connectionName*.
8. Provide a name for the Datasource; for example, OracleDB_DS.
9. Optionally enter a description of the Datasource; for example, Datasource on top of Oracle DB.
10. In the Query field, provide the appropriate SQL query that selects the Oracle Database data you want to make available in this Datasource.

Create Datasource

General Columns Parameters Preview

* Connection: custDBaaS

* Name: MyOracleDB_DS

Description: Data source for Oracle DB

* Query: SELECT * FROM SAMPLE_BASIC_TABLE

Create Cancel

11. Click **Next**. If the SQL statement was correct to query an Oracle Database area, you should see the queried columns populated.
12. Change any numeric columns to Double, and click **Next**.
13. Change any additional source-specific parameters, if applicable, and click **Next**.
14. Review the preview panel. You should see the results of the SQL query fetching columns of data from Oracle Database.
15. If the preview looks correct, click **Create** to finish creating the Datasource.

Create a Connection and Datasource for Oracle Autonomous Data Warehouse

Define a connection and Datasource between Essbase and Autonomous Data Warehouse.

To do this from Global Sources, you need to have the service administrator role. To do it from application level sources, you need to have the user role, plus application manager permission on the application.

1. In Essbase, on the Sources page, click **Connections**.
To define the connection and Datasource at application level, instead of globally, start on the Applications page instead of the Sources page. From the Actions menu to the right of an application name, launch the inspector and click **Sources**.
2. Click **Create Connection** and select **Oracle Database**.
3. Select **Autonomous** using the toggle switch.

Create Connection

Oracle Database

Autonomous

* Name

* Service Name

Wallet File

* User

* Password

Description

Test Create Cancel

4. Enter a connection name and a service name.
5. Drag and drop a wallet file, or click to upload.
Obtain a wallet file by selecting **Download Client Credentials (Wallet)** from your Autonomous Data Warehouse Administration page in Oracle Cloud Infrastructure.
6. Enter your Autonomous Data Warehouse username, password, and optionally, a description.
7. Click **Test** to validate the connection, and if successful, click **Create**.
8. Verify that the connection was created successfully and appears in the list of connections. Next, you will create a Datasource for the Autonomous Data Warehouse connection.
9. Click **Datasources**, and click **Create Datasource**.
10. From the Connection drop-down box, select the name of the connection you just created; for example, EssbaseADW. For application-level Datasources, select the application-level connection name, in the format *appName.connectionName*.
11. Provide a name for the Datasource; for example, ADW_DS.
12. Optionally enter a description of the Datasource; for example, Autonomous Data Warehouse Datasource.
13. In the **Query** field, provide the appropriate SQL query that selects the Autonomous Data Warehouse data you want to make available in this Datasource.
14. Click **Next**. If the SQL statement was correct to query an Autonomous Data Warehouse area, you should see the queried columns populated.
15. Change any additional source-specific parameters, if applicable, and click **Next**.


16. Review the preview panel. You should see the results of the SQL query fetching columns of data from Autonomous Data Warehouse.
17. If the preview looks correct, click **Create** to finish creating the Datasource.

Create a Connection and Datasource to Access Another Cube

Define a connection and Datasource between two Essbase cubes.

1. In Essbase, on the Sources page, click **Connections**.
To define the connection and Datasource at application level, instead of globally, start on the Applications page instead of the Sources page. From the Actions menu to the right of an application name, launch the inspector and click **Sources**.
2. Click **Create Connection** and select **Essbase**.
3. Enter a connection name; for example, `Essbase_FinanceCube_Conn`.
4. Check the box to **Use URL**, and enter the connection details to an Essbase instance. Connection information is provided by your Service Administrator.

Create Connection


Essbase

* Name

Use URL

* URL

Host

Port

* User

* Password

Description

Use the discovery URL. A discovery URL is the URL provided by your Service Administrator, with `/agent` appended to the end. For example:

```
https://192.0.2.1:443/essbase/agent
```

5. Click **Test** to validate the connection, and if successful, click **Create**.
6. Verify that the connection was created successfully and appears in the list of connections.

Next, you will create a Datasource for the Essbase connection.

7. Click **Datasources**, and click **Create Datasource**.
8. From the **Connection** drop-down box, select the name of the connection you just created.
9. Enter a name for the Datasource, and an optional description.
10. Select the application and database that will be used for this Datasource.
11. Provide a valid MDX query that selects the cube data you want to make available in this Datasource.

Create Datasource

< Back General Columns Parameters Preview Next >

* Connection: Essbase_FinanceCube_Conn

* Name: Essbase_FinanceCube_DS

Description: Connection to my other cube

* Application: Sample

* Database: Basic

* MDX Query: SELECT
 {([West].children)}
 ON COLUMNS,
 {([Diet].children)}
 ON ROWS
 FROM Sample.Basic

Create Cancel

12. Click **Next**. If the MDX syntax was correct to query the cube, you should see the queried columns populated.
13. Change any numeric columns to Double, and click **Next**.
14. Change any additional source-specific parameters, if applicable, and click **Next**.

15. Review the preview panel. You should see the results of the MDX query fetching columns of data from the other cube.
16. If the preview looks correct, click **Create** to finish creating the Datasource.

Create a Connection and Datasource to Access a Data File

Define a connection and Datasource between Essbase and a source data file.

1. Upload the source data file to the file catalog on Essbase.
If you need a sample source data file for this task flow, you can copy and paste `UserDetails.csv` from the gallery section of the file catalog to your application's file catalog. It represents a data repository of 22 users, with their associated countries, cost centers, currency, managers, company, business units, and offices.
2. In Essbase, on the Sources page, click **Connections**.
To define the connection and Datasource at application level, instead of globally, start on the Applications page instead of the Sources page. From the Actions menu to the right of an application name, launch the inspector and click **Sources**.
3. Click **Create Connection** and select **File**.
4. Enter a name for the connection; for example, `UserDetails_Conn`.
5. Provide the catalog path to the source data file.
6. Enter an optional description; for example, `Connection to user repository for filters`.
7. Click **Test** to validate the connection, and if successful, click **Create**.
8. Verify that the connection was created successfully and appears in the list of connections.
Next, you will create a Datasource for the file connection.
9. Click **Datasources**, and click **Create Datasource**.
10. From the **Connection** drop-down box, select the name of the connection you just created; for example, `UserDetails_Conn`.
11. Enter a name for the Datasource, and an optional description.

Create Datasource

< Back General Columns Parameters Preview Next >

* Connection: UserDetails_Conn

* Name: UserDetails_DS

Description: User details repository

Header Row:

Start Row: 1

End Row:

Delimiter: Comma

Create Cancel

12. Essbase detects and enters details about the source data; for example, whether it has a header row, and is comma-delimited. Click **Next**.
13. You should see the columns populated from the file source. Change any numeric columns to Double, and click **Next**.
14. If the preview looks correct, click **Create** to finish creating the Datasource.

If you update the source file metadata (for example, to add columns), you must recreate the Datasource.

9

Build Dimensions and Load Data

Building dimensions is the process of converting source data containing information about dimensions and members into a database outline, including hierarchies, using an Essbase data source and rule file. Loading data is the process of adding data values to a cube from any number of data sources.

A data source can contain data values, information about members (such as member names, member aliases, and formulas), generation and level names, data storage properties, attributes, and user-defined attributes (UDAs). Since data sources seldom are configured solely to support dimension build and data load processes, a rule file is generally used to create Essbase-compatible directives to be applied to the data source.

- [Typical Workflow for Building Dimensions and Loading Data](#)
- [About Dimension Builds](#)
- [About Data Loads](#)
- [Work with Rules](#)
- [Build Dimensions and Load Data Using a Rule File](#)
- [Upload Files to a Cube](#)
- [Build Dimensions and Load Data by Streaming from a Remote Database](#)
- [Build Dimensions and Load Data Using SQL](#)

Typical Workflow for Dimension Builds and Data Loads

The workflow for building dimensions and loading data into a cube includes the steps described here. The use of rules are necessary if the dimensions and data are not in Essbase-ready format.

1. For data sources other than flat files, set up a connection to the data source, and then select the application-specific data source.
2. Build dimensions using a rule, and then run the build dimension job in the Essbase web interface.
3. Load data using a rule, and then run the load data job in the Essbase web interface.

About Dimension Builds

Dimensions, and their associated hierarchies, can be built from various types of data sources, using a rule file.

Building dimensions is the process of adding dimensions and members to an Essbase database outline by using a data source and a rule file. You can also use Outline Editor to build dimensions and members manually.

For more information on dimensions and members, see [Add Dimensions and Members to Outlines](#).

You can build dimensions using one of the following methods:

- Building a flat file of dimensions using a rule file. See [Build Dimensions Using a Rule File](#).
- Using SQL. See [Build Dimensions using SQL](#).
- Using the CLI tool and the streaming option. [Build Dimensions and Load Data by Streaming from a Remote Database](#).
- Adding dimensions manually in the Outline Editor. See [Add Dimensions and Members to Outlines](#).
- In cube designer, by opening an application workbook and clicking **Build Cube** on the cube designer ribbon. You can also update cubes incrementally in cube designer. See [Update Cubes Incrementally in Cube Designer](#).

About Data Loads

Loading data is the process of adding data values to a cube from any number of data sources or SQL database. Since data sources seldom are configured solely to support Essbase dimension build and data load processes, a rule file is generally used to create Essbase-compatible directives to be applied to the data source.

You must have the Database Update role to load data into a cube, and Database Manager role to create the necessary artifacts.

For all files that you upload and import to Essbase, using Essbase or command line interfaces, their file name lengths are limited to 30 characters, including file extension characters. You must shorten file names accordingly before performing these operations. In addition, when you are building or changing cubes using application workbooks, the name of the workbooks must be no more than 30 characters.

You can load data to a cube using one of the following methods:

- Using a flat data file or database table, using a rule file. See [Load Data Using a Rules File](#).
- Using SQL. See [Load Data Using SQL](#).
- Streaming from a remote database. See [Build Dimensions and Load Data by Streaming from a Remote Database](#).
- Loading data values from an application workbook that you are using to build a cube. See [Create a Cube from an Application Workbook](#).
- Submitting data values in Smart View. See [About the Submit Data Options](#) in the *Working with Oracle Smart View for Office*.

Work with Rules

Using rules, you can define operations that Essbase performs on data values and dimensions and members, loaded from a data source. You also use rules, if necessary, to map data values to an Essbase cube, or dimensions and members to an Essbase outline.

Rules are stored in rule files. A rule file loads the rule that defines which build method to use, whether data values or members are sorted or in random order, and how to transform data values or members before loading them. You can create a separate rule for each dimension.

Essbase reads the data values or dimensions in the data source, loads them based on the rules. Essbase doesn't change the data source. You can reuse a rule with any data source that requires the same set of rules.

If you build a cube from an application workbook, Essbase creates the rules for you.

If a load data rule already exists, you must edit it when you add a new dimension, change a data source for your analysis, or change mappings or properties.

You can also use rules, while building dimensions and loading data, to do the following:

- Define operations that Essbase performs on data values or on dimensions and members when it processes a data source.
- Map data values to an Essbase database.
- Map dimensions and members to an Essbase outline.

You must use a rule in the following instances.

- You need a rule if you're loading data, and you need to define the mapping of the data source fields to database fields.
- If loading data from a SQL data source or a database, you need a rule to map the relational table information of the database columns to dimensions.
- When building dimensions, if you're adding or changing dimensions and members in the database, you need a rule.
- If you change fields in any way, including the mapping of data, and order of fields, you need to use a rule when loading the data.
- You need to prepare a separate rule file for each unique non-Essbase source, whether the source requires a dimension build or load data rule.

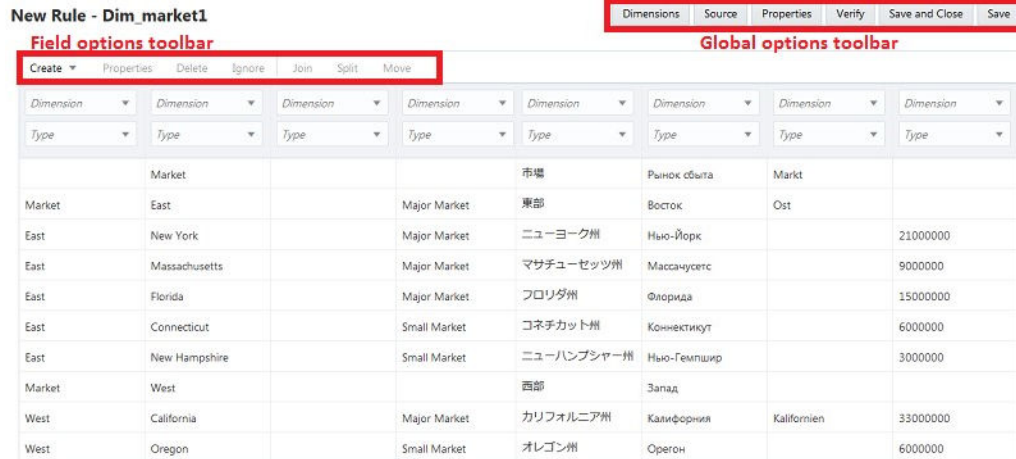
If you're using native format data files, you do not need to use a rule file.

Global and Field Options

Various options can be specified at the global and field level in the Rules editor, when creating and editing a rule.

The sequence of steps to open the Rules editor are as follows: On the home page, expand an application to see its cube, open the row's Actions menu, select Inspect, Scripts, and then Rules, and then create or edit a rule.

The options that are available, in the Global and Field option toolbars on the Rules editor, are described here.



For information on member property codes, see Using the Data Source to Work with Member Properties in *Designing and Maintaining Essbase Cubes*.

Global Options

The Global toolbar, on the upper right of the Rules editor page, allows you to edit the data source, general file options, and properties. In most cases, the default options do not need to be modified.

Table 9-1 Global Options in Rules Editor

Global Options Toolbar Tabs	Description
Dimensions (main tab)	(For dimension builds only.) This main tab allows you to enter a new dimension, select from existing ones, and edit their global options.
Dimensions, General	(For dimension builds only.) <ul style="list-style-type: none"> • Member Name: This can be left empty. It is the same value as the dimension name. • Type, Storage, Config, Unique, and Hierarchy: These options have the default value of “Existing”, when the dimension already exists, or you can select values from the menu. • Allow xxxx Changes: Allows you to make changes to mapped associations, properties, formula, and user-defined attributes.
Dimensions, Advanced	(For dimensions builds only.) <ul style="list-style-type: none"> • Update Option: Enables you to use Merge to do incremental updates, or Remove Unspecified to delete existing members and replace file contents. • Create Attributes: Allows you to add and associate attributes. • Moves: Allows you to move members between hierarchies. Generation-2 option allows only generation-2 members to be moved.
Dimensions, Measure Properties	(For dimension builds only.) Applies to dimensions of type measures.
Dimensions, Attribute Properties	(For dimension builds only.) Applies to attribute dimensions.
Source, General	Enables you to define data source options, specify header (for repeating header values), and specify token ignore or token join options.

Table 9-1 (Cont.) Global Options in Rules Editor

Global Options Toolbar Tabs	Description
Source, File Properties	Enables you to change file property options, including: set header, file type, and delimiter.
Source, SQL/Datasource Properties	Enables you to set properties and queries for SQL (such as Oracle SQL server) or datasource properties.
Properties	Includes load and sign flip options for data loads, and smart list options for dimension build text measures. Also enables you to clear multiple combinations of members by entering them in rows.
Verify	Enables you to verify the syntax of the rule. Errors are displayed.

Field Options

The Field Options toolbar, on the left of the Rules editor page, allows you to set field-level properties and options.

Table 9-2 Field Options in Rules Editor

Field Options Toolbar Tabs	Description
Create, Regular	Allows you to add a field (column).
Create, With Static Value	Allows you to add a field with a specified static value.
Create, With Join	(Not available for dimension build indexed based rules). Allows you to create a field using a join. First, you select multiple field columns, and then enter a value in the Join Position dialog box, for the field position (column) in which to place the joined fields.
Create, With Expression	(For dimension build index based rules only.) Allows you to create an expression within a rule.
Properties, General	Allows you to change a field name, handle a smart list (text measures), handle upper and lower case letters, set date format (for a Date dimension), and trim spaces (remove leading or trailing spaces).
Properties, Filters	Allows you to set filters including to select (include), reject (exclude), and replace (find and replace tool to fix mistakes in the data).
Expression	(For dimension build index based rules only.) Allows you to create or edit an expression within a rule.
Delete	Allows you to delete a field from the rule after highlighting the field column. You can't delete a field after performing a field operation on it, such as join or split.
Ignore	Toggle option that allows you to exclude a field from processing (the field display is grayed out) by highlighting a field column. Any fields that are not mapped can be set to Ignore.
Join	(Not available for dimension build index based rules.) Allows you to join (merge) two fields by highlighting the two fields' columns, and joining them into the first field's position, in Join Position .

Table 9-2 (Cont.) Field Options in Rules Editor

Field Options Toolbar Tabs	Description
Split	(Not available for dimension build index based rules.) Allows you to split a field's data into two fields. Highlight the two fields' columns, and then enter the Split Position (from where you want to split the field). For example, if the field's value is "NewYork" and the Split Position value is "3", then the new (split) fields are "New" and "York".
Move	(Not available for dimension build index based rules.) Allows you to move a field by highlighting a field column, and then moving the data to another field's column (with the target field number specified in the Move field to value).

Build Dimensions and Load Data Using a Rule File

Using a rule, you can build a dimension and load data from a text or other flat file.

Before you begin, you will need the following resources.

- Access to an Essbase instance.
- If you're not using a flat file as the source of data, you will need a connection and Datasource that have been set up in Essbase at the application level.
- Dimension metadata file (sample exercise file: `dim-market.txt`) downloaded to your computer.
- Data file (sample exercise file: `data-basic.txt`) downloaded to your computer.

Using the listed resources, you can now perform the tasks of building dimensions and loading data using a rule.

Build Dimensions Using a Rule File

You can edit and map dimensions to an Essbase outline using a rule, rather than manually building empty dimensions in the Essbase Outline editor. In this section, we address and illustrate building dimensions from a flat file, using a rule.

When you build using a rule, you define the hierarchical structure of dimensions and member metadata. You can create one or more dimensions using a single rule file, or use one rule file per dimension.

You can build a dimension to add or modify dimensions, but you can't use it to delete an existing dimension.

Here, we illustrate an example of building dimensions, from a flat file, using rules. The process of loading data using SQL, or by streaming, is described in other topics.

1. Open the downloaded dimension metadata file, `dim-market.txt`, in a formatted text editor. Notice that the file doesn't have a header row and that the file delimiter is a comma.
2. Sign into the Essbase web interface.
3. On the home page, expand the Sample application, and select the Basic cube.
4. Now you create the rule file.

- a. From the **Actions** menu to the right of the cube, launch the inspector.
- b. Click **Scripts**, and then **Rules**. The Rules editor is displayed, showing the currently defined rules.
- c. Click **Create**, and select **Dimension Build (Indexed Based)** to define the build dimension rule. An index-based build dimension rule removes dependency of fields to each other and allows the fields to appear in any order.
- d. In the New Rule dialog box, enter `Dim_market1` as the name of the rule file.
- e. Under Preview Data, select **File** for the flat file input option.
- f. Click the browse icon and locate the file `dim-market.txt` that you downloaded, and click **Open** to select it.
- g. As you saw earlier, the first row of the flat file doesn't contain header values. Deselect the **Header Row** check box if it is selected.
- h. Specify the **Delimiter** value as Comma, based on the file format.
- i. Click **Proceed**.

You can now preview the dimension structure in the Rules editor, with the columns displayed based on the input flat file.

The top-right toolbar in the Rules editor shows the Global options for a rule. You can change the properties or data source here and view the results. The left toolbar of the Rules editor shows the Field options for the rule.

5. On the Rules editor page, you can now set up and edit the rule.
 - a. On the Preview page for the new rule, in the first field (column), click **Dimension**, and select **Market** as the dimension name. The Market dimension is now assigned to all fields.
 - b. Under Market, in the first field, it, click **Type**, and select the dimension type, **Parent**.

The source file for this rule is in parent-child format. If you had a generation-based source file, you could set the first field to Generation. In that case, the Generation Number is set to 2, as by default, the Generation 1 is the dimension itself.

- c. Set up the other fields:
 - Set Field 2 Type to **Child**.
 - Set Field 3 Type to **Property**, and third row Parent/Child box to **Child**.
 - For Field 4 and 5, set Type to **UDA**, and third row Parent/Child boxes to **Child**.
 - For Field 6-9, set Type to **Alias**, third row Alias boxes to **ChineseNames**, **JapaneseNames**, **RussianNames**, and **GermanNames** respectively; and fourth row boxes to **Child**.
 - Set Field 10 Type to **Attribute Member**, third row box to **Population**, and fourth row box to **Child**.

The Dimension field is most often set to Generation, Parent or Child. If the Dimension name you want isn't in the menu, click **Dimensions** (on the Global toolbar), add the dimension name, and click **Add** and **OK**.
- d. Now check the field properties for a field. Select the last field column, **Population**. On the Field options toolbar, open the **Properties** tab and verify

that the Case option is set to **No Operation** > This means that uppercase and lowercase text aren't handled differently here than they were in the source text file.

- e. In the Global toolbar, click the **Source** tab, if you want to change the data source file. On the File Properties tab, verify that the Delimiter is set to **Comma**.
- f. When you have finished defining the rule, click **Verify** in the Global toolbar, to validate the rule syntax.
- g. Click **Save and Close**.
- h. Click **Refresh**. See that your created rule is now listed in the Rules pane of the Scripts tab. You can edit your rule by clicking on the rule name and then clicking **Proceed**.

From the **Actions** menu for a listed rule, you can optionally copy, rename, copy or export the build (into a json file to be used for troubleshooting purposes). Click **Close** to return to the home page.

- 6. Next, you create and run a job to build the dimension using the rule.
 - a. On the home page, select, **Jobs**, and then **New Job**.
 - b. Select **Build Dimension**.
 - c. In the Build Dimension dialog box, from the **Application** list, select the **Sample** application.
 - d. In the **Database** list, select the **Basic** cube. It might take a few moments to load.
 - e. In the **Script** list, select the build dimension rule that you created, `Dim_market1.rul`.
 - f. For **Load Type**, select **File**.
 - g. In the **Data File** list, select the `Dim_Market` as the data dimension data file. This file is located in the Sample, Basic folder.
 - h. From the **Restructure Options** list, select the **Preserve Input Data** option for the data you want to preserve.

To disconnect other users who are connected to the Sample, Basic cube, so that you can immediately build the dimension, you could select **Force to Build Dimension**.

For leaf level data, only level-0 values are preserved. Use this option if all data required for the calculation resides in level-0 members. For input data, only blocks that contain data being loaded are preserved. Neither option applies to aggregate storage databases.

- i. Click **OK**. The build dimension job is executed.
- j. On the Jobs page, click **Refresh** to monitor the job status.
- k. When the job completes, click the **Actions** menu for the executed job, and select **Job Details** to verify the status of your build job.
- l. On the Applications home page, to the right of the Basic cube in the Sample application, open **Actions**, and then **Outline** to verify the dimension hierarchy. In **Actions, Database, Inspect**, you can also view the created generation names under the dimension tab. When done, exit the view.

You have now completed building a dimension using a rule.

Load Data Using a Rule File

You can use rules to extract, transform and load data values to an Essbase cube. The source data values can contain the following:

- Data values
- Member names, aliases and formulas
- Generation and level names
- Data storage properties
- Attributes and user—defined attributes

When you build an Essbase cube, data files and load data rule files are created in the cube directory. You can also use data and rules from a supported on-premises version of Essbase.

Both pivot data and row set flat file data format are supported.

When you load data, SUM, MIN, MAX, AVG, and COUNT operations are supported in data columns across rows. This supports big-data use cases in which Essbase cubes are created with upper-level members. You can drill through, from Essbase, to view the data at a more granular level.

Here, we illustrate an example of loading data from a flat file, using rules. The process of loading data using SQL, or by streaming, is described in other topics.

1. Open the downloaded data file, `data-basic.txt`, in a formatted text editor. Notice that there's no header row and that the file delimiter is a comma.
2. Sign in to the Essbase web interface.
3. On the home page, expand the Sample application, and select the Basic cube.
4. Now create the load rule.
 - a. From the **Actions** menu to the right of the Basic cube, launch the inspector.
 - b. Select **Scripts** tab, and then **Rules**. The Rules editor is displayed, showing currently defined rules.
 - c. Click **Create**, and select **Data Load** to define the load data rule.
 - d. In the New Rule dialog box, enter **Data_basic1** as the name of the rule.
 - e. Enter **Measures** as the data dimension.
 - f. Under Preview Data, select **File** for flat file input.
 - g. Click the browse icon to locate the file `data-basic.txt` that you downloaded, and click **Open** to select it.
 - h. As you saw earlier, the first row of the flat file doesn't contain header values. Deselect the **Header Row** check box if it is selected. When the header row is present, the columns are mapped automatically.
 - i. Select **Comma** as the **Delimiter** value, based on the file format.
 - j. Click **Proceed**.

You can now see the preview of the data in the Rules editor, based on the input flat file.

The Global options toolbar, on the top right of the Rules editor allows you to modify file properties or the data source and to see the results in the Rules editor. The Field options toolbar on the left side of the Rules editor allows you map fields in the rule.

Because there were no headers in the input file, you need to map each column to the appropriate dimensions and members.

5. In the Rules editor, you can now set up the rule fields.
 - a. Click **Create** drop-down menu, and start setting the field names.
 - Set Field (column) 1 to **Product**.
 - Set Field 2 to **Market**.
 - Set Field 3 to **Year**.
 - Set Field 4 to **Scenario**.
 - Set Field 5 to **Sales**.
 - Set Field 6 to **COGS**.
 - Set Field 7 to **Marketing**.
 - Set Field 8 to **Payroll**.
 - Set Field 9 to **Misc**.
 - Set Field 10 to **Opening Inventory**.
 - Set Field 11 to **Additions**.

All dimensions must be represented in the load data rule before any data can be loaded.
 - b. When you are finished defining the rule, with global and field options, click **Verify** on the Global toolbar to validate the syntax and click **Close**.
 - c. After syntax is validated, click **Save and Close**.
 - d. Click **Refresh**. See that your created rule is now listed in the Rules pane of the Scripts tab. You can edit your rule by clicking the rule name and then clicking **Proceed**.
 - e. Click **Close** to return to the Applications home page.

Next, create a job to load the data using the rule.

6. On the home page, select **Jobs**, and then **New Job**.
 - a. Select **Load Data**.
 - b. In the Load Data dialog box, from the **Application** menu, select the **Sample** application.
 - c. In the **Database** list, select the **Basic** cube.
 - d. In the **Script** list, select the load data rule that you created, `Data_market1.rul`.
 - e. For **Load Type**, select **File**.
 - f. Select the file `Data_Basic1` from the **Data File** list. This file is located in the `Sample > Basic` folder.
 - g. Optional: select the **Abort on error** check box if you want the load to stop if an error occurs.

Build Dimensions and Load Data by Streaming from a Remote Database

If the data or dimensions you want to load to a cube are in a remote database, you can use the stream option in the Oracle Command Line Interface (CLI) utility, to push the data or members to your cube, using a rule file.

When you use the **stream** option for the CLI [Dataload: Load Data to a Cube](#) or [Dimbuild: Load Dimensions to a Cube](#) command, you must also reference a saved JDBC connection that reflects your driver and connection strings.

Before you Begin

1. The rule file must exist in the Files section for the relevant database.
2. The database query used to load data or build dimensions must have the same dimensionality as the columns in the rule file. (For example, see [Build Dimensions Using SQL](#), where the order of dimensions in the rule file must match the order of dimensions in the SQL query).

Limits

- Substitution variables are not supported in SQL statements used in load rules.
- Only use SQL functions that are supported by JDBC. ODBC scalar functions are not supported in CLI.

Workflow for Streaming Dimension Builds and Data

1. Create a saved JDBC connection string that reflects your data source's driver and connection strings, using the CLI [Createlocalconnection: Save a JDBC Connection](#) command.
2. (Not required for Oracle database) Set an environment variable `EXTERNAL_CLASSPATH` to point to the .jar file for your database driver. See the *Examples of EXTERNAL_CLASSPATH Environmental Variables* section in this topic.
3. Run the CLI [Dataload: Load Data to a Cube](#) or [Dimbuild: Load Dimensions to a Cube](#) command with the streaming option, providing the saved connection name.

You can optionally specify the database query in the dataload or dimbuild command. Otherwise, you can specify it in the load rules, in the **Select** section of the **Data Source** tab. For examples, see [Build Dimensions Using SQL](#) and [Load Data Using SQL](#).

Examples of EXTERNAL_CLASSPATH Environmental Variables

You must set the `EXTERNAL_CLASSPATH` environment variable before you can stream from any data source other than the Oracle database. Set the variable to point to the location of relevant database driver .jar file.

DB2

Set the external classpath variable to point to the location of the DB2 driver jar file.

C Shell Example

```
setenv EXTERNAL_CLASSPATH /scratch/db/jars/db2jcc.jar
```

Korn or Bash Shell Example

```
export EXTERNAL_CLASSPATH=/scratch/db/jars/db2jcc.jar
```

Windows Example

```
set EXTERNAL_CLASSPATH=C:\db\jars\db2jcc.jar
```

MySQL

Set the external classpath variable to point to the location of the MySQL driver jar file.

C Shell Example

```
setenv EXTERNAL_CLASSPATH /scratch/db/jars/mysql-connector-java-5.1.43-bin.jar
```

Korn or Bash Shell Example

```
export EXTERNAL_CLASSPATH=/scratch/db/jars/mysql-connector-java-5.1.43-bin.jar
```

Windows Example

```
set EXTERNAL_CLASSPATH=C:\db\jars\mysql-connector-java-5.1.43-bin.jar
```

Microsoft SQL Server

Set the external classpath variable to point to the location of the SQL Server driver jar file.

C Shell Example

```
setenv EXTERNAL_CLASSPATH /scratch/db/jars/sqljdbc4-3.0.jar
```

Korn or Bash Shell Example

```
export EXTERNAL_CLASSPATH=/scratch/db/jars/sqljdbc4-3.0.jar
```

Windows Example

```
set EXTERNAL_CLASSPATH=C:\db\jars\sqljdbc4-3.0.jar
```

Teradata

Set the external classpath variable to point to the location of both Teradata driver jar files.

C Shell Example

```
setenv EXTERNAL_CLASSPATH /scratch/db/jars/tdgssconfig.jar:/scratch/db/
jars/terajdbc4.jar
```

Korn or Bash Shell Example

```
export EXTERNAL_CLASSPATH=/scratch/db/jars/tdgssconfig.jar:/scratch/db/
jars/terajdbc4.jar
```

Windows Example

```
set EXTERNAL_CLASSPATH=C:\db\jars\tdgssconfig.jar;C:\db\jars\terajdbc4.jar
```

Build Dimensions and Load Data Using SQL

Using SQL, you can import a table to an RDBMS server, create dimension build and data load rules, connect to the RDBMS, and load dimensions and data to a cube.

Before you begin, you will need the following resources.

- Access to an Essbase instance
- Access to an RDBMS server
- Oracle SQL Developer, or another SQL integrated development environment
- A tabular data file. For example, in this exercise, consider a `sample_basic_table.csv` file that contains columns of data, as in the following abbreviated representation (many rows are omitted):

Figure 9-1 Tabular Data File to Import to Relational Database

PRODUCT	MARKET	YEAR	SCENARI	SALES	STATENAM	COGS	MARKETIN	PAYROLL	MISC	BEGINV	ADDITIONS
100-10	Central	Sep	Actual	107	Ohio	43	13	22	0		102
100-10	Central	Sep	Budget	110	Ohio	40	10	20			90
100-10	Central	Oct	Actual	107	Ohio	43	13	22	1		112
100-10	Central	Oct	Budget	100	Ohio	30	0	10			90
100-10	Central	Nov	Actual	114	Ohio	46	14	22	0		125
100-10	Central	Nov	Budget	110	Ohio	40	10	20			120
100-10	Central	Dec	Actual	101	Ohio	41	13	22	1		96
100-10	Central	Dec	Budget	90	Ohio	40	10	20			90
100-10	Central	Jan	Actual	190	Wisconsin	79	72	29	1	551	180
...
400-10	Central	Aug	Budget	90	Colorado	40	10	30			100
400-10	Central	Sep	Actual	102	Colorado	45	17	33	0		97
400-10	Central	Sep	Budget	80	Colorado	30	10	30			90
400-10	Central	Oct	Actual	120	Colorado	54	20	33	0		126
400-10	Central	Oct	Budget	80	Colorado	30	0	20			110

- A valid connection string, as listed below. For both connections, you do not have to edit `odbc.ini`. Essbase makes the connection using the ODBC driver.

Connection Strings

The available OCI and DSN-less connection string types are listed, with syntax and examples.

Oracle Call Interface (OCI)

Syntax: `$Keyword$DatabaseServerName:PortNumber/SID`

Example: `OCIMyOracleServer:1521/ORCL`

Oracle Database (DSN-less)

Syntax (SID): `oracle://HostName:PortNumber/SID`

Example (SID): `oracle://somedb99:1234/ORCL`

Syntax (ServiceName): `ORACLESERVICE:oracle://HostName:PortNumber/ServiceName`

Example (ServiceName): `ORACLESERVICE:oracle://somedb99:1234/esscs.host1.oraclecloud.com`

Microsoft SQL Server (DSN-less)

Syntax: `sqlserver://HostName:1433:DBName`

Example: `sqlserver://myMSSQLHost:1433:myDbName`

DB2 (DSN-less)

Syntax: `db2://HostName:PortNumber:DBName`

Example: `db2://myDB2Host:1234:myDbName`

MySQL (DSN-less)

Syntax: `mysql://HostName:3306:DBName`

Example: `mysql://someHostName:3306:myDbName`

Once you have all the prerequisite information listed in this topic, you can perform the tasks of building dimensions and loading data using SQL.

- [Build Dimensions Using SQL](#)
- [Load Data Using SQL](#)

No members from a CellProperties dimension should be included in tabular data or in the headers of the SQL-based load rule files.

Build Dimensions Using SQL

This task flow demonstrates how to import a table to a RDBMS server, create dimension build rules, connect to the RDBMS, and build dimensions using SQL.

Before you begin, complete the prerequisites and obtain a valid connection string. See [Build Dimensions and Load Data Using SQL](#)

1. In Oracle SQL Developer (or your choice of SQL tool), import a table from a flat file (for example, `sample_basic_table.csv`), to your SQL database server connection.

An example of the imported table, `SAMPLE_BASIC_TABLE`, is shown here.

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	PRODUCT	VARCHAR2 (256 BYTE)	Yes	(null)	1	(null)
2	MARKET	VARCHAR2 (256 BYTE)	Yes	(null)	2	(null)
3	YEAR	VARCHAR2 (256 BYTE)	Yes	(null)	3	(null)
4	SCENARIO	VARCHAR2 (256 BYTE)	Yes	(null)	4	(null)
5	SALES	NUMBER (25,0)	Yes	(null)	5	(null)
6	STATENAME	VARCHAR2 (256 BYTE)	Yes	(null)	6	(null)
7	COGS	NUMBER (25,0)	Yes	(null)	7	(null)
8	MARKETING	NUMBER (25,0)	Yes	(null)	8	(null)
9	PAYROLL	NUMBER (24,0)	Yes	(null)	9	(null)
10	MISC	NUMBER (23,0)	Yes	(null)	10	(null)
11	BEGINV	NUMBER (25,0)	Yes	(null)	11	(null)
12	ADDITIONS	NUMBER (25,0)	Yes	(null)	12	(null)

Next, you will delete some members from Sample Basic, and then create a load rule to rebuild the Market dimension from the SQL table.

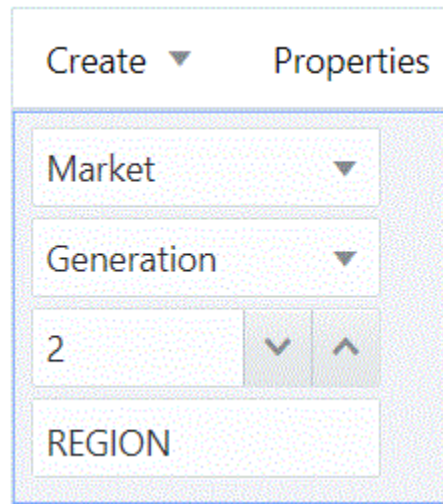
- In the Essbase web interface, on the Applications page, expand the Sample application, and select the cube, Basic.
- From the **Actions** menu to the right of Basic, select **Outline**.
- Click the Market dimension, and then click member East.
- Click **Edit** to lock the outline for editing.
- Delete some of the states from the East market. For example, delete Connecticut, New Hampshire, and Massachusetts.
- Click **Save**, and then verify that East now contains only the states Florida and New York.

Next, you will create dimension build rules and repopulate the Market dimension, from the SQL table, with the states you have removed.

- Close the Outline browser tab.
- On the Applications page, from the **Actions** menu to the right of Basic, launch the inspector, click **Scripts**, then choose the **Rules** tab.
- Click **Create > Dimension Build (Regular)** to begin defining new dimension build rules.
- In the **Name** field, enter the name of the rules file as MarketSQLDimbuild. Leave the other options as-is, and click **Proceed**.
- Click the **Dimensions** button.
- Click the field containing the text **Select existing dimension**, select Market, and click **Add**, then **OK**.
- On the **New Rule - MarketSQLDimbuild** page, click the **Dimension** drop-down field and select Market.
- Click the **Type** drop-down field and select **Generation**. Increment the generation number to 2.

16. Click the **Generation Name** field and type REGION.

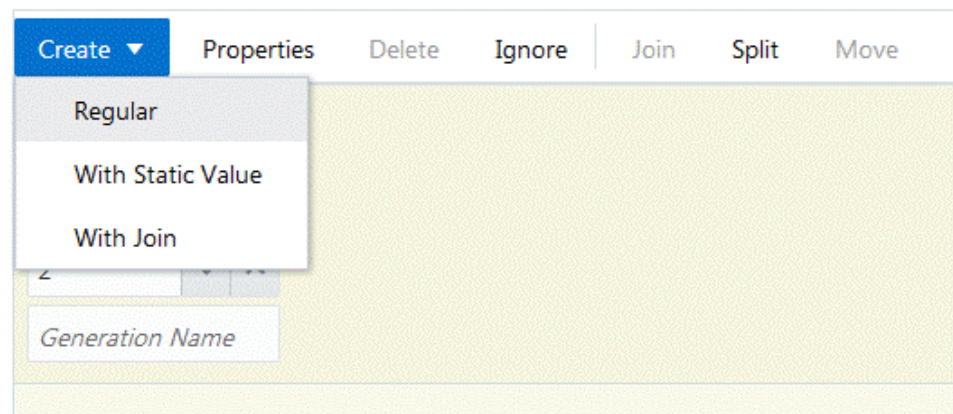
The Market dimension is generation 1, and you added a child named Region.



The screenshot shows a 'Create' dialog box with a 'Properties' tab. It features a 'Market' dropdown menu, a 'Generation' dropdown menu, a numeric input field containing '2' with up and down arrow buttons, and a text input field containing 'REGION'.

17. Click **Create > Regular** to create a second dimension build rule field.

New Rule - MarketSQLdimbuild



The screenshot shows a 'New Rule' dialog box with a 'Create' dropdown menu open. The menu options are 'Regular', 'With Static Value', and 'With Join'. Below the menu, the 'Generation Name' field is visible.

18. Name the field STATE and associate it with dimension Market, at generation 3.

19. Click the **Source** button to begin associating a data source with the dimension build rules.
20. In the **General** tab, enter the valid connection string.
 - a. For Oracle Call Interface connections: In the **Name** field of the **General** group, enter the valid OCI connection string.

Edit Source

- b. For DSN-less connections, such as Oracle DB, Microsoft SQL Server, and DB2: You must leave the **Name** field of the **General** group empty. Instead, enter the connection string in the **Server** field of the **SQL/Datasource Properties** group. The format is *oracle://host:port/sid* for an Oracle database.
21. In Oracle SQL Developer (or your alternate SQL tool of choice), write and test a SELECT statement selecting some columns from the table SAMPLE_BASIC_TABLE:


```
Select distinct market,statename from SAMPLE_BASIC_TABLE
```
22. If the SQL query is valid, it should return the requested table columns, Market and Statename, from the database to which your SQL tool is connected:

	MARKET	STATENAME
1	Central	Wisconsin
2	South	Louisiana
3	East	Massachusetts
4	East	Connecticut
5	Central	Colorado
6	East	Florida
7	South	Oklahoma
8	West	Oregon
9	West	Washington
10	West	Nevada

23. Copy the SELECT statement to your clipboard. The results of this query are the dimensions you will load into the Sample Basic cube.
24. Back in the **Edit Source** dialog for your dimension build rule, paste the SQL statement into the **Query** field of the **SQL/Datasource Properties** group.

Edit Source

General File Properties **SQL/Datasource Properties**

Properties SQL Datasource

Server

Application

Database

Dictionary

Query

25. Click **OK**, then **Verify, Save and Close**. to save and close the MarketSQLDimbuild rule.
26. Refresh the list of rules in the Scripts list to ensure that MarketSQLDimbuild has been added to the list of rule files for the cube Sample Basic.
27. Click **Close**.
Next, you will use this rule file to load the members back into the Market dimension.
28. Click **Jobs**, and click **New Job > Build Dimension**.
29. Enter Sample as the application name, and Basic as the database name.
30. For the script name, select the name of the dimension build rule file you created, MarketSQLDimbuild.

31. Select **SQL** as the load type.
32. Leave **Connection** blank, unless you already have a saved SQL connection you wish to use.
33. Enter the user name and password of one of your SQL database schema users.
34. Leave **Data File** blank.
35. From the **Restructure Options** drop-down list, select **Preserve All Data**.
36. Click **OK** to begin the job.

The dimension build begins. Click the Refresh symbol to watch the status, and when it completes, click **Job Details** from the Actions menu.

37. Inspect the outline to verify that your dimensions were built (verify that Connecticut, New Hampshire, and Massachusetts exist as children under East).

Load Data Using SQL

This task flow demonstrates how to clear data from a cube, create data load rules, load data (using SQL) from an RDBMS server, and verify in Smart View that the data was loaded.

Before beginning this task flow, complete prerequisites and obtain a valid connection string. See [Build Dimensions and Load Data Using SQL](#) for details.

1. After building the dimensions, you will clear data from the cube, and then load the data again from a table. In Essbase, click **Jobs**, and click **New Job**.
2. Select **Clear Data** as the job type. Select application Sample and database Basic, and click OK.
3. Click OK to confirm that you want to clear data. The job begins. Click the Refresh symbol to watch the status, and when it completes, click **Job Details** from the Actions menu.
4. Connect to the Sample Basic cube from Smart View, and do an ad hoc analysis.
5. Notice that data was cleared. For example:

				Sales
Actual	Connecticut	Cola	Jan	#Missing
Budget	Connecticut	Cola	Jan	#Missing
Variance	Connecticut	Cola	Jan	#Missing
Variance	Connecticut	Cola	Jan	#Missing
Scenario	Connecticut	Cola	Jan	#Missing

Keep the worksheet open. Next, you will create load rules that use SQL to repopulate the Sales data from the table.

6. On the Applications page, expand the Sample application, and select the cube, Basic.
7. From the **Actions** menu to the right of Basic, launch the inspector, click **Scripts**, then choose the **Rules** tab.
8. Click **Create > Data Load** to begin defining new load rules.
9. In the **Name** field, enter the name of the rule file as SalesSQLDataLoad.

10. In the **Data Dimension** drop-down box, select the Measures dimension.
11. Leave the other options as-is, and click **Proceed**.
12. In Oracle SQL Developer (or your alternate SQL tool of choice), write and test a SELECT statement selecting some columns from the table
SAMPLE_BASIC_TABLE: `Select Product,Year,Scenario,Statename,Sales
from SAMPLE_BASIC_TABLE`
13. Ensure that the SQL query is valid and returns a result in your SQL tool. If the SQL query is valid, it should return the requested table columns, PRODUCT, YEAR, SCENARIO, STATENAME, and SALES, from the database to which your SQL tool is connected:

	PRODUCT	YEAR	SCENARIO	STATENAME	SALES
1	100-10	Sep	Actual	Ohio	107
2	100-10	Sep	Budget	Ohio	110
3	100-10	Oct	Actual	Ohio	107
4	100-10	Oct	Budget	Ohio	100
5	100-10	Nov	Actual	Ohio	114
6	100-10	Nov	Budget	Ohio	110
7	100-10	Dec	Actual	Ohio	101
8	100-10	Dec	Budget	Ohio	90
9	100-10	Jan	Actual	Wisconsin	190
10	100-10	Jan	Budget	Wisconsin	190

14. Copy the SQL query to a text file or your clipboard. You will need to use this in an upcoming step. The results of this query are the data you will load into the Sample Basic cube.
15. Note the order of dimensions in your SQL query. The dimensions of the load rule fields must appear in the same order. This means that when you add fields, you should first add the last dimension listed in the SQL query (Sales). Each time you add a new field, it appears in front of the previous one, so when you are finished adding all fields, the dimensional order will match that of the SQL query.
16. In Essbase, in the **New Rule** browser tab for your SalesSQLDataLoad rule, select Sales from the **Select** drop-down box.
17. Click **Create > Regular** to create a second load rule field. From the **Select** drop-down box, select Market (which maps to Statename in your SQL query).
18. Click **Create > Regular** to continue adding fields, in this order: Scenario, Year, and Product.

Your load rule fields should now be arranged like this:

**New Rule -
SalesSQLDataLoad**

Dimensions Source Properties Verify Save and Close Save

Create ▾ Properties Delete Ignore Join Split Move

Product ▾	Year ▾	Scenario ▾	Market ▾	Sales ▾
-----------	--------	------------	----------	---------

19. Click the **Source** button to begin associating a data source with the load rules.
20. In the **General** tab, enter the valid connection string.
 - a. For Oracle Call Interface (OCI) connections: In the **Name** field of the **General** group, enter the valid connection string.

Edit Source

General | File Properties | SQL/Datasource Properties

Name \$OCI\$mydsn01:1521/ORCL

- b. For DSN-less connections, such as Oracle Database, Microsoft SQL Server, and DB2: You must leave the **Name** field of the **General** group empty. Instead, enter the connection string in the **Server** field of the **SQL/Datasource Properties** group.

Edit Source

General | File Properties | SQL/Datasource Properties

Properties SQL Datasource

Server \$OCI\$mydsn01:1521/ORCL

21. Click **OK**.
22. Verify, save, and close the SalesSQLDataLoad rule.
23. Refresh the list of rules in the Scripts list to ensure that SalesSQLDataLoad has been added to the list of rule files for the cube Sample Basic, and then close the database inspector.

Next, you will load the data from Jobs.

24. Click **Jobs**, and click **New Job > Load Data**.
25. Enter Sample as the application name, and Basic as the database name.
26. For the script name, select the name of the dimension build rule file you created, SalesSQLDataLoad.
27. Select **SQL** as the load type.

28. Leave **Connection** blank, unless you already have a saved SQL connection you wish to use.
29. Enter the user name and password of one of your SQL database schema users.
30. Leave **Data File** blank.
31. Click OK to begin the job.

The data load begins. Click the Refresh symbol to watch the status, and when it completes, click **Job Details** from the Actions menu.

32. Go back to the worksheet in Smart View, and refresh it to verify that the data was loaded from the table.

				Sales
Actual	Connecticut	Cola	Jan	310
Budget	Connecticut	Cola	Jan	290
Variance	Connecticut	Cola	Jan	20
Variance	Connecticut	Cola	Jan	6.896552
Scenario	Connecticut	Cola	Jan	310

10

Calculate Cubes

A cube contains two types of values: values that you enter, called input data, and values that are calculated from input data.

A cube can be calculated using one of two methods. Outline calculation, which is the simplest calculation method, bases the calculation of a cube on the relationships between members in the cube outline and on any formulas that are associated with members in the outline.

Calculation script calculation lets you procedurally calculate a cube; for example, you can calculate one part of a cube before another, or copy data values between members.

The topics in this section are about calculation script calculation:

- [Access to Calculations](#)
- [Create Calculation Scripts](#)
- [Execute Calculations](#)
- [Use Substitution Variables](#)
- [Set Two-Pass Calculation Properties](#)
- [Tracing Calculations](#)
- [Calculate Selected Tuples](#)

Access to Calculations

If you have the Database Update user role, you have access to run the default calculation on the cube (from Smart View), and to run specific calculation scripts provisioned to you. If you have the Application Manager or Database Manager role, you have Calc privileges and rights to execute all calculations, and to provision access to execute specific calculation scripts.

When creating or editing a calculation script in the Essbase web interface, you can use the Permissions page within the script editor to provision users to execute the script.

Create Calculation Scripts

Calculation scripts specify how block storage cubes are calculated and, therefore, override outline-defined cube calculations. For example, you can calculate cube subsets or copy data values between members.

You create calculation scripts using a script editor in the Essbase web interface.

Calculation scripts do not apply to aggregate storage applications.

1. On the Application page, expand the application.
2. From the Actions menu, to the right of the cube name, launch the inspector.

3. Select the **Scripts** tab, and then select the **Calculation Scripts** tab.
4. Click Add **+** to create a new calculation script.
5. If member names are required in your calculation script, drill into the **Member Tree** to find the members you want to add.

Right-click dimension or member names to insert them into the script.

6. If function names are required in your calculation script, use the **Function Name** menu to find calculation functions and add them to the script.

See the **Function description** under the menu to read descriptions of each function.

7. Click **Validate** before saving your script.

Validating a script verifies the script syntax. For example, incorrectly spelled function names and omitted end-of-line semicolons are identified. Validation also verifies dimension names and member names.

8. Correct any validation errors.

Calculation scripts can contain runtime substitution variables designed to derive the calculation scope from the point of view (POV) in a Smart View grid. These types of calculation scripts will not pass validation on the server, because the point of view can only be known from a Smart View grid.

9. Click **Save**.

To learn about calculation script logic, see *Developing Calculation Scripts for Block Storage Databases*.

To learn about calculation functions and commands, see *Calculation Functions and Calculation Commands*.

Execute Calculations

After creating and saving calculation scripts, you use the Jobs page to execute them and perform the calculations on data loaded in your cube.

1. Create your calculation script, or upload an existing calculation script.
2. In Essbase, click **Jobs**.
3. On the **Jobs** page, click **New Job** and select **Run Calculation**.
4. On the **Run Calculation** dialog box, select the application and cube you want to calculate.
5. Select the script you want to use.
6. Click **OK** to start the calculation.
7. Click **Refresh** to see the status of your calculation.

Calculation scripts can contain runtime substitution variables designed to derive the calculation scope from the point of view (POV) in a Smart View grid. These types of calculation scripts will not execute from the server, because the point of view can only be known from a Smart View grid.

You can also execute calculation scripts from Smart View (whether or not they contain point-of-view based substitution variables).

Assign access to execute specific calculation scripts:

1. Log into the Essbase web interface as a service administrator or power user.
2. On the Applications page, expand an application, and select a cube.
3. From the **Actions** menu, to the right of the cube name, launch the inspector.
4. Select the **Scripts** tab, and then select the **Calculation Scripts** tab.
5. Select a script and select the **Permissions** tab.
6. Add the users or groups to assign them access and save your changes. The users or groups are given permission to execute the specific calculation script.

See also: [Create Calculation Scripts](#).

See also: [Upload Files to a Cube](#).

Use Substitution Variables

Use **substitution variables** in calculation scripts to store values that might change. Use **runtime substitution variables** when you need different users to specify different values for the same script.

For example, if a variety of your calculation scripts, formulas, filters, report scripts, and MDX scripts all need to refer to the current month, you would not want to search and replace the month approximately every 30 days throughout your library of cube artifacts. Instead, you can define a substitution variable named CurrMonth, and change its assigned value each month to the appropriate month. All of the cube artifacts that reference the variable will then reference the appropriate month.

Here is an example of a simple substitution variable to represent the current month:

Variable name: CurrMonth

Value: Jan

Substitution variable values apply to all users who run a calculation script containing the variable. For example, if CurrMonth has the value Jan, then all scripts containing &CurrMonth will execute for Jan. The scope of a substitution variable can be:

- global (for all applications and cubes on the server)
- application (for all cubes in the application)
- cube (for a single cube)

To define a substitution variable for a specific cube,

1. In the Essbase web interface, on the Applications page, expand the application to show the cube you want to modify.
2. From the **Actions** menu to the right of the cube, launch the inspector.
3. Select the **Variables** tab, and click Add **+**.
4. Enter the variable name and value, click **Save**, and click **Close**.

To define a substitution variable for a specific application,

1. On the Applications page, from the Actions menu to the right of the application, launch the inspector.
2. Select the **Variables** tab, and click Add **+**.

3. Enter the variable name and value, click **Save**, and click **Close**.

To define a substitution variable globally,

1. In Essbase, click **Console**.
2. Click the **Variables** tab, and click **Add**.
3. Enter the variable name and value, and click **Save**.

Once your substitution variable is defined, you can use it in calculation scripts, formulas, filters, MDX scripts, load rules, and reports. To reference the variable, prefix it with the & symbol.

Here is an example of a calculation script that references a substitution variable:

```
FIX(&CurrMonth)
  CALC DIM (Measures, Product);
ENDFIX
```

Here is an example of a formula that references a substitution variable:

```
@ISMBR(&CurrMonth)
```

Runtime substitution variables enable you to declare variables and their values in the context of a runtime action, such as a calculation script, MaxL script, or MDX query. Runtime substitution variables can be assigned to have numeric values or refer to member names. A default value can be assigned in case a user does not change an input value. Also, for calculation scripts, the variable value can be populated at runtime from the members of a dimension presented on a Smart View grid. For calculation scripts with variable values that populate at runtime, you must launch the calculation script from Smart View, as the variable has no definition outside the context of the grid.

Runtime substitution variables may be defined in the calculation script using key-value pairs:

```
SET RUNTIMESUBVARS
{
  myMarket = "New York";
  salesNum = 100;
  pointD = "Actual"->"Final";
}
```

Or, to define runtime substitution variables with values that change dynamically depending on the POV, assign the definition to POV, and use XML syntax to enable Smart View contextual prompts.

For more information, see

- Using Substitution Variables
- Using Runtime Substitution Variables in Calculation Scripts Run in Essbase and Using Runtime Substitution Variables in Calculation Scripts Run in Smart View
- The SET RUNTIMESUBVARS calculation command
- The gallery template Sample_Basic_RTSV, which you can find in Files > Gallery > Technical > Calc.

Set Two-Pass Calculation Properties

The Two-Pass Calculation property can be applied to members in non hybrid mode, block storage cubes to indicate members that need to be calculated twice to produce the desired value. To obtain the correct values for two-pass members, the outline is calculated, and then members that are dependent on the calculated values of other members are recalculated.

Even though two-pass calculation is a property that you can give to any non-attribute dimension member, it works only on members of the Accounts dimension and Dynamic Calc members. If two-pass calculation is assigned to any other member, it is ignored.

Two-pass calculations are supported only on block storage cubes. Hybrid mode and aggregate storage cubes use member solve order, instead of two-pass calculation, to control when members are calculated. In hybrid mode, members tagged as two pass are assigned a [solve order](#) of 100 by default.

1. On the Applications page, expand the application.
2. From the Actions menu, to the right of the cube name, select **Outline**.
3. Click **Edit**.
4. In the outline editor, find and select the member you want to modify.
5. In the **Properties** pane, expand the **Two-pass Calculation** menu, and select **True**.

See Setting Two-Pass Calculations.

Trace Calculations

You can use calculation tracing to analyze member formula processing, and refine your calculation scripts.

Calculation tracing enables you to access logged information about a calculation, after the calculation script successfully executes against a cube.

Tracing a calculation does not change anything about calculation behavior. If a calculation is launched in Smart View, and the connected server has calculation tracing enabled by an administrator, Smart View displays a pop-up dialog box containing details, after the calculation runs. The calculation tracing information can be pasted from the pop-up dialog into a text editor. Or, you can find the same information in `calc_trace.txt`, located in the database files directory in Essbase.

The calculation tracing information can help you debug calculation script execution, in case the results of the calculation are not what you expected.

Calculation tracing is not supported on applications with scenario management enabled.

To enable calculation tracing, the administrator must first turn on the `CALCTRACE` application configuration parameter. After calculation tracing is enabled for your application, there are two ways to take advantage of it:

- In Smart View, you can use context-sensitive tracing for a single cell value.

1. In Smart View, connect a query sheet to the application for which you enabled calculation tracing.
 2. Highlight a data cell whose calculated value you would like to trace.
 3. In the Data panel of the Essbase tab, click the **Calculate** button and select a calculation script to execute. You will see the point-of-view from your highlighted data cell in the trace member runtime prompts.
 4. Click **Launch** to execute the calculation script.
The full scope of the calculation as contained in the script will be calculated, but only the highlighted data cell context will be traced during the calculation.
 5. At the end of the calculation script, examine the **Calculation Result** dialog box, which shows the pre- and post-calculation results for your highlighted data cell.
If the highlighted data cell was not modified during the calculation, you will see a message indicating that the cell was not modified.
- In calculation scripts, you can use the SET TRACE calculation command to select data intersections to trace. SET TRACE enables you to trace multiple data cells. Additionally, you can trace sections of calculation scripts by using a combination of SET TRACE *mbrList* (to turn calculation tracing on over a member list) and SET TRACE OFF (to disable calculation tracing until a new SET TRACE is encountered in the script. To use SET TRACE command, you must execute the calculation script outside of Smart View, using Cube Designer or the Jobs page in the Essbase web interface.

```
SET TRACE ("100-10", "California", "Jan", "Sales", "Budget");
FIX("California", "Budget")
  "Sales" (
    "100-10" = @MEMBER(@CONCATENATE(@NAME(@PARENT("Product")), "-20")) /
  10;
  );
ENDFIX;
```

Sample Basic has two sparse dimensions: Product and Market. The member formula is on Sales, a member of Measures, which is a dense dimension. The FIX statement's member list only contains one sparse member, California, which belongs to the Market dimension.

The number of existing blocks in the FIX statement determines the number of times the traced cell is calculated. In this example, the calculation cycles through all existing sparse member combinations of California. Each of these combinations represents a block.

After the calculation completes, the following tracing information is logged and displayed:

```
Tracing cell: [100-10][California][Jan][Sales][Budget] (Cell update
count: 1)
Previous value: 840.00
Dependent values:
  [100-20][California][Jan][Sales][Budget] = 140.00
New value: [100-10][California][Jan][Sales][Budget] = 14.00

Computed in lines: [91 - 93] using:
```



```

"Sales"(
"100-10"=@MEMBER(@CONCATENATE(@NAME(@PARENT("Product")), "-20"))/10;
)

Tracing cell: [100-10][California][Jan][Sales][Budget] (Cell update
count: 2)
Block from FIX scope: [100-30][California]
Actual block used in calculation: [100-10][California]
Previous value: 14.00
Dependent values:
  [100-20][California][Jan][Sales][Budget] = 140.00
New value: [100-10][California][Jan][Sales][Budget] = 14.00
Computed in lines: [91 - 93] using:
"Sales"(
"100-10"=@MEMBER(@CONCATENATE(@NAME(@PARENT("Product")), "-20"))/10;
)

Tracing cell: [100-10][California][Jan][Sales][Budget] (Cell update
count: 3)
Block from FIX scope: [200-10][California]
Actual block used in calculation: [100-10][California]
Previous value: 14.00
Dependent values:
  [200-20][California][Jan][Sales][Budget] = 520.00
New value: [100-10][California][Jan][Sales][Budget] = 52.00
Computed in lines: [91 - 93] using:
"Sales"(
"100-10"=@MEMBER(@CONCATENATE(@NAME(@PARENT("Product")), "-20"))/10;
)

[...calc iterations 4-7 are omitted from example...]

Tracing cell: [100-10][California][Jan][Sales][Budget] (Cell update
count: 8)
Block from FIX scope: [400-30][California]
Actual block used in calculation: [100-10][California]
Previous value: 9.00
Dependent values:
  [400-20][California][Jan][Sales][Budget] = 90.00
New value: [100-10][California][Jan][Sales][Budget] = 9.00
Computed in lines: [91 - 93] using:
"Sales"(
"100-10"=@MEMBER(@CONCATENATE(@NAME(@PARENT("Product")), "-20"))/10;
)

```

The calculation tracing log provides the following insights about how the calculation worked, on the cell that was traced:

- The traced cell was calculated several times, and the cell value was overwritten each time with the new value (the reported cell update count stops at 8).
- The value of the cell, before calculation, was 840.00.
- For each calculation occurrence, dependent values and new values are shown. Dependent values come from the member formula in the FIX statement.

- The final value of the traced cell, after all calculation completes, is 9, but it represents the value of product "400-20" -> California divided by 10.
- Lines 91-93 of the calculation script, containing a member formula on Sales, are responsible for the updated values.

For each of the blocks cycled through, Sales is calculated using the formula:

```
"100-10"=@MEMBER(@CONCATENATE(@NAME(@PARENT("Product")), "-20"))/10
```

The formula contains a sparse member on the left hand side, which could cause the actual calculation block to be different than the initial FIX block. For example, when the calculation cycles through "California"->"100-20", the calculations are actually done in "California"->"100-10".

The trace log entries entitled `Block from FIX scope` and `Actual block used in calculation` are only printed if there is a discrepancy between the blocks in the FIX statement and the block that is represented in the member formula. These log entries can provide indications as to why there are duplicate calculations, helping you to debug your calculation scripts.

Calculate Selected Tuples

By selecting tuples, you can focus your calculations in the active Smart View grid, limiting their scope to specific slices of data in your cube. Tuple selection helps you optimize asymmetric grid calculations across dimensions, avoiding over-calculation.

Essbase calculation tuples differ from tuples used in MDX queries. Calculation performance and cube size are mainly driven by the number of blocks in the database (given a specific block size). For this reason, calculation tuples are specified only for sparse member combinations. In addition, for ease of calculation scripting, multiple members from a single sparse dimension can be included in a calculation tuple specification. For example, if you specify ("New York", "California", "Actual", "Cola") as a calculation tuple, then you calculate the following cell intersections:

```
"New York"->"Actual"->"Cola"  
"California"->"Actual"->"Cola"
```

Consider the following symmetric grid. It is symmetrical because each product has the same markets and scenario (Actual) represented in the grid.

		Profit	Inventory	Ratios
		Actual	Actual	Actual
		Jan	Jan	Jan
Cola	New York	█	█	█
	Massachus	█	█	█
	Florida	█	█	█
	Connectic	█	█	█
	New Hamp	█	█	█
Diet Cola	New York	█	█	█
	Massachus	█	█	█
	Florida	█	█	█
	Connectic	█	█	█
	New Hamp	█	█	█

The following grid is asymmetric, because the Diet Cola product has fewer markets in the grid than the Cola product has.

		Profit	Inventory	Ratios
		Actual	Actual	Actual
		Jan	Jan	Jan
Cola	New York	█	█	█
	Massachus	█	█	█
	Florida	█	█	█
	Connectic	█	█	█
	New Hamp	█	█	█
Diet Cola	New York	█	█	█
	Florida	█	█	█

The default calculation scope, when more than one dimension is in a FIX statement or a Smart View grid point of view (POV), is to calculate the cross product (all possible combinations) of the members in the FIX or grid. In other words, a POV-driven calculation in which product and market combinations are taken from the grid calculates all of these row-member combinations:

```
Cola->"New York"
Cola->"Massachusetts"
Cola->"Florida"
Cola->"Connecticut"
Cola->"New Hampshire"
"Diet Cola"->"New York"
"Diet Cola"->"Massachusetts"
"Diet Cola"->"Florida"
"Diet Cola"->"Connecticut"
"Diet Cola"->"New Hampshire"
```

This may be more calculation activity than you need. If you want to calculate *only* the combinations shown on the grid, you can specify which tuples to calculate, and limit the calculation to a smaller slice. Calculating tuples can also lower calculation time and cube size.

```
Cola->"New York"
Cola->"Massachusetts"
```

```
Cola->"Florida"  
Cola->"Connecticut"  
Cola->"New Hampshire"  
"Diet Cola"->"New York"  
"Diet Cola"->"Florida"
```

Tuple-Based Calculation

A calculation **tuple** is a way to represent a data slice of members, from two or more sparse dimensions, to be used in a calculation.

Examples of valid calculation tuples:

- ("Diet Cola", "New York")
- ("Diet Cola", "Cola", Florida)
- (Cola, "New Hampshire")

If you write MDX expressions, you might be aware of these tuple restrictions that apply to MDX:

- Only a single member from each dimension can be included in an MDX tuple
- All tuples in an MDX set must have the same dimensions represented, in the same order

However, when you select tuples in calculation scripts, these requirements are relaxed for convenience. You may freely write tuple expressions, and the tuples may describe member lists, as the following tuple does: (@Children(East), Cola).

Select Tuples for Point of View Calculation

An easy way to select tuples is to insert them explicitly into a calculation script, as a list inside the FIX statement.

Recall that the format of a FIX statement is as follows:

```
FIX (fixMbrs)  
COMMANDS ;  
ENDFIX
```

In the FIX statement below, two tuples are specified before the command block begins. The tuples are enclosed within the curly braces { } that delimit a **set**, which is a collection of tuples.

```
FIX({  
    (@Children(East), Cola),  
    ("New York", Florida, "Diet Cola")  
})  
Sales (Sales = Sales + 10);  
ENDFIX
```

Another way to select tuples is contextually, based on whichever members are present in a Smart View grid POV at calculation run time. You do this by providing the @GRIDTUPLES function as an argument to FIX, in your calculation script.

```
FIX ({@GRIDTUPLES(Product, Market)})
  Sales (Sales = Sales + 10);
ENDFIX
```

If you execute this calculation script from Smart View against the grid below, then only the displayed combinations of products and markets are calculated. For example, "Diet Cola"->Massachusetts is not calculated, as it is not shown explicitly on the grid. Note that all scenarios (the third sparse dimension in this sample cube) are calculated, even though only Actual is shown on the grid. This is because the Scenario dimension is not part of the GRIDTUPLES statement in the calculation script.

		Profit	Inventory	Ratios
		Actual	Actual	Actual
		Jan	Jan	Jan
Cola	New York			
	Massachusetts			
	Florida			
	Connecticut			
	New Hampshire			
Diet Cola	New York			
	Florida			

Tuple selection, whether done using explicit lists of tuples or by using the @GRIDTUPLES function, is applicable only in the context of the FIX...ENDFIX calculation command. The syntax of the FIX statement is expanded to enable tuple selection:

```
FIX ([{ tupleList | @GRIDTUPLES(dimensionList) },] fixMbrs)
COMMANDS ;
ENDFIX
```

- *tupleList* - comma-separated set of tuples.
- *dimensionList* - at least two sparse dimensions whose members from the active Smart View grid are used to define the calculation regions. (In calculation scripts, you can use only sparse dimensions to define tuples.)
- *fixMbrs* - a member or list of members.

Examples of Tuple Selection to Reduce Calculation Scope

Using a Smart View grid and a calculation script FIX statement, you can calculate selected member tuples based on the grid point of view (POV). Alternatively, you can explicitly type the tuple combinations in your FIX statement, removing the dependency on a particular Smart View grid to define the calculation scope.

Calculating selected tuples helps you efficiently work with asymmetric regions in both calculation scripts and Smart View grids.

Consider the following examples:

- **No Tuple Selection** - Calculates in the default manner, based on current Smart View grid point-of-view (POV). The calculation is not limited to any specific tuples.
- **Selection of Named Sparse Dimensions** - Calculates tuples from two or more sparse dimensions named in a calculation script. The calculation is limited to members from the tuple dimensions that are present in the Smart View grid.
- **Selection of Contextual Sparse Dimensions** - Calculates tuples from sparse dimensions selected at run-time. The calculation is limited to members from the tuple dimensions present in the Smart View grid.

To try the examples, download the `CalcTuple_Tuple.xlsx` workbook template from the Technical > Calc section of the **gallery** folder in the **Files** area of the Essbase web interface. Refer to the README worksheet in the workbook for instructions.

No Tuple Selection

Demonstrating the default calculation behavior that occurs when you do not select tuples, the following calculation script calculates the entire cross-product of Product and Market dimension members from a Smart View grid.

With the help of two runtime substitution variables (RTSV) defined in the SET RUNTIMESUBVARS block, calculation is limited to whichever Product and Market points of view are present in the grid when the calculation is run from Smart View.

```
SET RUNTIMESUBVARS
{
ProductGridMembers = POV
<RTSV_HINT><svLaunch>
<description>All Product's members on the grid</description>
<type>member</type>
<dimension>Product</dimension><choice>multiple</choice>
</svLaunch></RTSV_HINT>;
MarketGridMembers = POV
<RTSV_HINT><svLaunch>
<description>All Market's members on the grid</description>
<type>member</type> <dimension>Market</dimension><choice>multiple</choice>
</svLaunch></RTSV_HINT>;
};
FIX (
&ProductGridMembers, &MarketGridMembers
)
Marketing(
    Marketing = Marketing +1;
);
ENDFIX
```

Selection of Named Sparse Dimensions

Using the `@GRIDTUPLES` function to select the tuple of Product and Market dimensions, this calculation script calculates tuples for only those two dimensions,

limiting its scope to those members present in a Smart View grid at the time the calculation is executed from Smart View.

```
FIX (
  {@GRIDTUPLES(Product, Market)}
)
Marketing(
  Marketing = Marketing + 1;
);
ENDFIX
```

By fixing on only the sparse dimensions named in the tuple, the calculation encompasses a much smaller number of blocks than a default calculation would. However, all members from dimensions not mentioned in the fix (Year, Scenario) are calculated by this calculation script.

Selection of Contextual Sparse Dimensions

Using the @GRIDTUPLES function and a runtime substitution variable, this calculation script calculates only selected tuples from the grid, based on the sparse dimension selections in the RTSV prompt.

The runtime substitution variable *&DimSelections*, which is defined in the SET RUNTIMESUBVARS block, limits the calculation scope to only the sparse dimensions of the cube, excluding Scenario. The @GRIDTUPLES function used in the FIX statement calls this variable, limiting how many intersections are calculated.

```
SET RUNTIMESUBVARS
  {
    DimSelections = "Version", "Site", "Entity", "Product",
    "Market"
    <RTSV_HINT><svLaunch>
    <description>List two or more sparse dimensions used for
forming calculation tuples:</description>
    <type>string</type>
    </svLaunch></RTSV_HINT>;
  };
FIX (
  {@GRIDTUPLES(&DimSelections)}
)
Marketing(
  Marketing = Marketing + 1;
);
ENDFIX
```

The calculation encompasses an even smaller number of blocks than the previous example, because in this case, the tuple definition extends to more sparse dimensions beyond Product->Market.

To try the examples, download the CalcTuple_Tuple.xlsx workbook template from the Technical > Calc section of the **gallery** folder in the **Files** area of the Essbase web interface. Refer to the README worksheet in the workbook for instructions.

11

Run and Manage Jobs Using the Web Interface

The Jobs page in the Essbase web interface is a centralized place from which to execute operations over the entire Essbase platform.

Essbase administrators or users with execute permissions on certain applications can use the Jobs page to quickly execute jobs such as clearing and loading data, importing and exporting applications, running calculations and much more.

The Jobs page is convenient for one-time execution of administrative tasks, but it is not a replacement for scripted administration of Essbase platform jobs. MaxL, CLI, REST, and API programs are the most efficient way to schedule jobs for production activities and life cycle maintenance.

View Job Status and Details

Users have access to job listings based on their assigned user role. For example, if you have the Service Administrator role, you can see all jobs; if you have the User role, you can see only the jobs you ran.

Because Essbase jobs run in the background, you must refresh the Jobs page to view their status.

The job listing shows all the jobs for all the applications provisioned to the logged in user. You can scroll down to see the history of all the jobs that you ran.

1. On the Applications page, click **Jobs**.
2. Click **Refresh** to refresh once, or click **Auto Refresh** to refresh the jobs every few seconds. In Cube Designer, job status refreshes automatically.

You can also view details for an individual job. To view job details, click the **Actions** menu to the right of the job listing, and select **Job Details** to see input and output details for a job.

Execute Jobs

You can execute numerous types of jobs from the Jobs page. For each, you choose an option from the **New Job** drop-down list, and then provide the necessary information.

Aggregate storage:

- [Build Aggregations](#)
- [Clear Aggregations](#)

Block storage:

- [Export to Table Format](#)

- [Run Calculation](#)

Aggregate storage and block storage:

- [Build Dimension](#)
- [Clear Data](#)
- [Export Data](#)
- [Export Excel](#)
- [Export LCM](#)
- [Import LCM](#)
- [Load Data](#)
- [Run MDX](#)

Build Aggregations

Building aggregations requires Database Access permission.

Aggregations apply to aggregate storage cubes. Aggregations are intermediate stored consolidations called Aggregate Views. Aggregate views store upper-level intersections, which support query performance by avoiding dynamic aggregations on the most commonly queried intersections in the cube. The term aggregation is used to refer to the aggregation process and the set of values stored as a result of the process.

When you build an aggregation, Essbase selects aggregate views to be rolled up, aggregates them based on the outline hierarchy, and stores the cell values in the selected views. If an aggregation includes aggregate cells dependent on level 0 values that are changed through a data load, the higher-level values are automatically updated at the end of the data load process.

Build Aggregations

* Application

* Database

* Ratio To Stop

Based On Query Data

Enable Alternate Rollups

To build aggregations:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Build Aggregations**.
3. For **Application**, choose an application.
4. For **Database**, choose a cube.

5. Optionally, enter a non-zero value for **Ratio To Stop**.
Leaving **Ratio To Stop** at zero (the default) means there is no stopping ratio set.

Consider this option if there is no known common type of query executed by your cube's users, and you want to improve performance by limiting the cube's growth. Essbase aggregates the selected views, with the exception that the maximum growth of the aggregated cube must not exceed the given ratio. For example, if the size of a cube is 1 GB, specifying the total size as 1.2 means that the size of the resulting data cannot exceed 20% of 1 GB, for a total size of 1.2 GB.
6. Check or clear the box for **Based on Query Data**.
If you check the box for **Based on Query Data**, Essbase aggregates a selection of views that is defined based on analysis of user querying patterns. This is a good approach if similar types of queries are typically executed by the users of your cube.

This check box has no effect unless you have first enabled query tracking. For general information about query tracking, see [Selecting Views Based on Usage](#).

After you've enabled query tracking, allow sufficient time to collect user data-retrieval patterns before running this job. A good approach is to prepare a set of your most important and long running queries, enable query tracking, run the prepared set of queries, and then run this job to create an aggregate view based on the query tracking.

While query tracking is enabled, the cost of retrieving cells is recorded for every level combination. This recording continues until the application is shut down or until you turn off query tracking (using the MaxL statement `alter database <db-name> disable query_tracking`).
7. Select whether to enable alternate rollups.
Consider checking this box if your cube implements alternate hierarchies for shared members or attributes, and you want to include them in the aggregation.

See [Aggregating an Aggregate Storage Database](#).

Clear Aggregations

Clears aggregations. Requires Database Update permission.

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Clear Aggregations**.
3. For **Application**, choose an application.
4. For **Database**, choose a cube.
5. Click **OK**.

See [Build Aggregations](#) and [Aggregating an Aggregate Storage Database](#).

Export to Table Format

If you have at least Database Update application permission, you can export a cube from the Essbase web interface into Excel, in tabular format.

This exported tabular data is organized into columns with headers that Essbase can use to deploy a new multidimensional cube. See [Export a Cube to Tabular Data](#).

To export a cube in tabular format:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Export to Table Format**.
3. For **Application**, select an application.
4. Choose whether to export dynamic blocks.
If you choose **Export to Dynamic Blocks**, cells for dynamic members in the dense dimensions are exported.

Run Calculation

Requires at least Database Update permission, as well as provisioned access to the calculation script.

Prerequisite: upload the script, as a .csc file, to the cube directory. See [Upload Files to a Cube](#).

To run a calculation:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Run Calculation**.
3. For **Application**, choose an application.
4. For **Database**, choose a cube.
5. Select a calculation script.
6. Click **OK**.

See [Calculate Cubes](#).

Build Dimension

Building dimensions is the process of loading dimensions and members to a cube outline using a data source and a rule file. To run a dimension build job, you must have at least Database Manager permission.

Build Dimension

* Application

* Database

* Script

Load Type SQL Datasource File

Use Connection

* Connection

* User name

* Password

* Data File

Restructure Options

Force to Build Dimension

This procedure covers how to build dimensions using the **File** load type. For information on the available load types see:

- [SQL - Build Dimensions Using SQL](#)
- [Datasource - Use Connections and Datasources](#)
- [File - Build Dimensions Using a Rule File](#)

To build a dimension:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Build Dimension**.
3. For **Application**, choose an application.
4. For **Database**, choose a cube.
5. Click the Actions menu to the right of the **Script** field and select a rule file.
6. Select the **File** load type.
7. Disregard **Connection**, as it is not relevant to the **File** load type.
8. Click the actions menu to the right of the **Data File** field to select a data file.
9. Choose a restructure option.
 - **Preserve All Data:** Preserves all existing data.
 - **Preserve No Data:** Discards existing data (valid for block storage and aggregate storage cubes).
 - **Preserve Leaf Level Data:** Preserves data in existing level 0 blocks (block storage only). If you select this option, all upper-level blocks are deleted before the cube is restructured. After restructure, only data in level 0 blocks remains.

- **Preserve Input Data:** Preserves existing input-level blocks (block storage only).

Clear Data

Changes the values of all cells containing data to #Missing. Requires at least Database Update permission.

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Clear Data**.
3. For **Application**, choose an application.
4. For **Database**, choose a cube.
5. Click **OK**.

Export Data

Exports data to a text file. Requires at least Database Manager permission.

Export Data

* Application ▼

* Database ▼

* Data Level ▼

Column Format


Compress

OK Cancel

To export data:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Export Data**.
3. For **Application**, choose an application.
4. For **Database**, choose a cube.
5. For **Data Level**, select a data level.
You can choose from **All Data**, **Level 0 Data**, or **Input Data**.

To download the exported data file:

1. On the applications page, click **Jobs**.
2. Select the Actions menu to the right of the export job.
3. Select **Job Details**.
4. To view the data file, you can click the **Output Path** link, or to download the file, select download .

The exported data file is stored in the database folder in the catalog.

Export Excel

Exports a cube to an Excel application workbook. Requires at least Database Manager permission.

Export Excel

* Application

* Database

* Export Build Method

Export Data

Export Scripts

To export to Excel:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Export Excel**.
3. For **Application**, choose an application.
4. For **Database**, choose a cube.
5. Select a build method.
See Understanding Build Methods.
6. Choose whether to export data. This option adds a data worksheet to the application workbook.
7. Choose whether to export scripts. This option adds Calc and MDX sheets to the application workbook if calculation scripts and MDX scripts exist in the cube.
8. Click **OK**.

Export LCM

Backs up cube artifacts to a Lifecycle Management (LCM) ZIP file. Requires at least Application Manager permission.

Export LCM

* Application

* Zip File

Skip Data

To back up cube artifacts to a Lifecycle Management (LCM) ZIP file:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Export LCM**.
3. For **Application**, select an application.
4. Enter a name for the ZIP file.
5. Choose whether to skip data. This option excludes data from the backup.

The ZIP file is stored in the user directory of the user who exported it.

See also: [LcmExport: Back Up Cube Files](#).

Import LCM

You must be the Power User who created the application, or a Service Administrator to execute the Import LCM job.

Restores cube artifacts from a Lifecycle Management (LCM) ZIP file that was created using the [Export LCM](#) job (or the [LcmExport: Back Up Cube Files](#) CLI command).

Import LCM

* Zip File

Application Name

Server Artifacts

Overwrite

Verbose

To restore cube artifacts from a Lifecycle Management (LCM) ZIP file:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Import LCM**.
3. Select the LCM export ZIP file.
4. Enter the target application name.
5. Select or clear **Server Artifacts**.
If server-level artifacts were included in the LCM export, you can check this box to also include server-level artifacts on LCM import.
6. Select or clear **Overwrite**.
Choosing to overwrite causes Essbase to recreate the target application. If overwrite is selected, the import operation deletes and recreates the entire application, importing only the artifacts present in the ZIP file. If overwrite is not selected, and the specified application name is the same as an existing application, the Import LCM job will fail.
7. Select whether to use verbose descriptions.
Choosing **Verbose** enables extended descriptions.

See also: [LcmImport: Restore Cube Files](#).

Load Data

This procedure covers how to load data using the file load type. For information on the available load types see:

- **SQL** – [Load Data Using SQL](#)
- **Datasource** - [Use Connections and Datasources](#)
- **File** – [Load Data Using a Rule File](#)

Load Data

* Application *Select Application*

* Database *Select Database*

Script *...*

Load Type SQL Datasource File

Use Connection

* Connection *Select Connection*

* User name *Enter user name*

* Password *Enter password*

* Data File *...*

Abort on Error

OK Cancel

To load data:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Load Data**.
3. Click the Actions menu to the right of the **Script** field to select a rule file if required.
4. For the **Load Type**, select **File**.
Connection is not relevant to the file load type.

If you are using a saved connection and datasource, no user name and password are required. If you are not using a saved connection, and the rule file connects to an RDBMS, specify the user name and password to connect to the RDBMS.

5. Click the Actions menu to the right of the **Data File** field to select a data file.
6. Select **Abort on Error** if you want to end the data load if it encounters an error.

Run MDX

Requires at least Database Access permission.

To run an MDX script:

1. On the Applications page, click **Jobs**.
2. From the **New Job** menu, select **Run MDX**.
3. For **Application**, choose an application.
4. For **Database**, choose a cube.
5. Select an MDX script.
6. Click **OK**.

See [Run MDX Scripts](#).

12

Adopt Hybrid Mode for Fast Analytic Processing

The Oracle Essbase calculation and query processor enables you to perform real-time analytics using procedural calculations and read-and-write modeling capabilities.

If you have worked with Essbase 11g on-premises, then you likely are familiar with one or more of these cube design modes, tailored for different purposes:

- **Block storage:** best used when there are large, sparse dimensions. Cubes in this mode are stored and pre-aggregated to achieve good query performance. Includes a rich set of calculation functions for analysis.
- **Aggregate storage:** best used for cubes having a large number of dimensions, and many upper-level aggregations.
- **Hybrid mode:** block storage mode enhanced with the benefits of aggregate storage.

Hybrid mode is the default query engine for aggregating block storage cubes in Essbase 19c and Oracle Analytics Cloud - Essbase. Hybrid mode provides robust dependency analysis and fast aggregation. It is excellent at handling the complexity of stored-member dependencies on dynamic members.

In your analytic applications, Oracle recommends the use of dynamic dependencies, including sparse aggregations; you are not limited to implementing dynamic calc selectively on sparse dimensions, as was the case in Essbase 11g on-premises. In particular, sparse dynamic aggregations are possible and recommended, subject to performance tuning guidelines and testing.

Though hybrid mode is the default for aggregating block storage cubes in in Essbase 19c and Oracle Analytics Cloud - Essbase, it is not the default engine for executing calculation scripts. If your calculation scripts contain many sparse dependencies, Oracle recommends you enable hybrid mode for calculation scripts as well. The way to do this is to turn on the `HYBRIDBSOINCALCSCRIPT` configuration setting in your application configuration properties (or use the `SET HYBRIDBSOINCALCSCRIPT` calculation command to control it on a per-calculation basis).

Most Essbase calculation functions will operate in hybrid mode. To see a list and syntax for all hybrid mode supported calculation functions, as well as the few exceptions, see [Functions Supported in Hybrid Mode](#). Parallel calculation using `FIXPARALLEL` is supported in hybrid mode, but not parallel calculation using `CALCPARALLEL`.

See `ASODYNAMICAGGINBSO` for the syntax to configure hybrid mode beyond the default settings, or to turn it off.

Topics in this section:

- [Benefits of Hybrid Mode](#)
- [Comparison of Hybrid Mode, Block Storage, and Aggregate Storage](#)
- [Get Started with Hybrid Mode](#)

- [Optimize the Database for Hybrid Mode](#)
- [Limitations and Exceptions to Hybrid Mode](#)
- [Solve Order in Hybrid Mode](#)

Benefits of Hybrid Mode

Hybrid mode combines block storage procedural calculation and write back functionality with aggregate storage aggregation performance. Hybrid mode offers the benefit of fast performance by eliminating the need to store sparse aggregations. This, in turn, reduces database size and memory footprint, and speeds up batch calculation times. The deployment considerations are simplified, as you no longer have to consider using block storage for heavy use of level 0 calculations, versus aggregate storage for many upper-level aggregations, versus designing partitioned models in which the cube is split along dimensional lines to facilitate calculation performance.

The following are some scenarios where hybrid mode is likely to improve calculation performance:

- A block storage database has sparse members that are not level 0, and are calculated according to hierarchy (rather than by calculation scripts).
- A sparse, dynamic calc parent member has more than 100 children.
- You are using a transparent partition between an empty aggregate storage target and a block storage source. If the formulas on the aggregate storage target are simple and translatable to block storage formula language, you can achieve fast results on block storage using hybrid mode.
- You are using a transparent partition between two block storage databases, and calculation performance is a concern.

Another benefit of hybrid mode is that it enables you to use [scenario management](#).

Comparison of Hybrid Mode, Block Storage, and Aggregate Storage

Without hybrid mode, large, sparse dimensions in block storage databases must be stored; making them dynamic would result in too much block I/O at query or calculation time, affecting performance. Very large stored sparse dimensions can lead to lengthy batch aggregation times, as well as large database sizes that grow in relation to the number and size of the sparse dimensions. Even with such drawbacks, block storage is widely used for its powerful functionality.

Aggregate storage is designed specifically to enable large databases with more and larger dimensions. Unlike block storage, it does not require large sparse dimensions to be pre-aggregated to achieve good query performance. The key lies in the aggregate storage database kernel, which facilitates rapid dynamic aggregation across large dimensionality.

For all the benefits that aggregate storage offers, however, there are many applications that are simply better suited to block storage, such as the need to do allocations or currency conversion. In such cases, hybrid mode might be the solution. Hybrid mode is a combination of the best features of block storage and aggregate storage. In hybrid mode, Essbase

- Enables full procedural calculation flexibility, even when the calculations depend on sparse, dynamic aggregations.
- Uses the hybrid engine for queries accessing dynamic sparse members. For the small percentage of queries that cannot be processed this way, Essbase employs the block storage calculation flow to satisfy the request.
- Offers these benefits, if you mark sparse members as dynamic:
 - Eliminates the need for pre-aggregation
 - Improves restructure performance
 - Improves backup performance
 - Reduces disk space requirements
- Because hybrid mode involves dynamic calculations, you can sequence the calculations by using [solve order](#).

 **Note:**

Hybrid calculations, whether driven by queries or calculation scripts, are performed in temporary memory space, utilizing a formula cache and the aggregate storage cache.

Get Started with Hybrid Mode

To get started with hybrid mode, follow these guidelines:

- Set up a development environment, and migrate existing block storage applications to it. Hybrid mode is enabled by default for block storage cubes.
- Where possible, make larger sparse dimensions dynamic.
- Run test queries and examine the application log, both before and after enabling hybrid mode. This activity can reveal the extent to which the aggregate storage query processor was used, and the benefits of hybrid mode that were gained. For each query, the application log states `Hybrid aggregation mode enabled` or `Hybrid aggregation mode disabled`.
- If too many queries are logged with hybrid mode disabled, contact Oracle Support.

Optimize the Database for Hybrid Mode

To use hybrid mode most effectively:

- If there are non-level-0 stored members that are batch calculated based solely on their hierarchy, Oracle recommends that you convert them to Dynamic Calc members.
- Avoid using two-pass calculation in hybrid mode. Use [solve order](#) instead.
- If the conversion to Dynamic Calc members affects solve order for dependent formulas, you may need to adjust the outline's order of dimensions to align the solve order with the previous batch calculation order and two-pass calc settings.

- You may need to adjust the dimensions' dense or sparse configurations (applies only to block-storage engine utilization in cases where the hybrid engine cannot be used).
- Minimize the size of blocks, if possible.

The default solve order for hybrid mode cubes is similar to the calculation order of block storage cubes, with some enhancements. If you wish to use a non-default solve order, you can set a custom solve order for dimensions and members.

Limitations and Exceptions to Hybrid Mode

In some cases, a query would not execute optimally in hybrid mode. Essbase detects when these conditions are present, and aggregates them in block storage mode. If a query mixes supported and unsupported hybrid mode calculation types, Essbase defaults to block storage calculation execution.

If enabled, hybrid mode is in effect for member formulas using supported functions. For a list of supported and unsupported functions, see [Functions Supported in Hybrid Mode](#).

The following types of queries are not executed in hybrid mode:

- Dynamic Calc members with formulas that are a target of transparent partitions
- Queries where the shared member is *outside* the target partition definition and its referenced member is *inside*, or the reverse
- XOLAP
- Text measures/text lists

Attribute calculations will execute in hybrid mode, for Sum only.

If dependent members have a higher [solve order](#) than the formula member, the following warning appears:

```
Solve order conflict - dependent member member_name with higher solve order will not contribute value for formula of member_name
```

Solve Order in Hybrid Mode

The concept of solve order applies to dynamic calculation execution, whether initiated by a dynamic member formula or a dynamic dependency in a calculation script. When a cell is evaluated in a multidimensional query, the order in which the calculations should be resolved may be ambiguous, unless solve order is specified to indicate the required calculation priority.

You can set solve order for dimensions or members, or you can use the default Essbase solve order. The minimum solve order you can set is 0, and the maximum is 127. A higher solve order means the member is calculated later; for example, a member with a solve order of 1 is solved before a member with a solve order of 2.

When hybrid mode is enabled, the default solve order (also known as calculation order) closely matches that of block storage databases:

Dimension/Member Type	Default Solve Order Value
Stored members	0
Sparse dimensions	10
Dense dimension - Account	30
Dense dimension - Time	40
Dense dimension	50
Attribute dimension	90
Two pass dynamic members	100

In summary, the default solve order in hybrid mode dictates that stored members are calculated before dynamic calc members, and sparse dimensions are calculated before dense dimensions, in the order in which they appear in the outline (top to bottom).

Dynamic members (with or without formulas) that do not have a specified solve order inherit the solve order of their dimension, unless they are tagged as two pass.

Two-pass calculation is a setting you can apply, in block storage mode, to members with formulas that must be calculated twice to produce the correct value. Two pass is not applicable in hybrid mode, and any members tagged as two pass are calculated last, after attributes. In hybrid mode, you should implement a custom solve order, instead of two pass, if the default solve order does not meet your requirements.

The default solve order in hybrid mode is optimized for these scenarios:

- Forward references, in which a dynamic member formula references a member that comes later in the outline order. There is no outline order dependency in hybrid mode.
- Aggregation of child values based on outline order more closely matches aggregation using equivalent formulas.
- Dynamic dense members as dependencies inside sparse formulas. In hybrid mode, if a sparse formula references a dense dynamic member, the reference is ignored, because sparse dimensions are calculated first. To change this, assign a solve order to the sparse dimension that is higher than (calculated later than) the dense dimension's solve order.

If you need to use a non-default solve order, you can set a custom solve order for dimensions and members in hybrid mode.

To change the solve order, use the outline editor in the Essbase web interface, or use Smart View (see [Changing the Solve Order of a Selected POV](#)).

If you implement a custom solve order, it overrides the default solve order. If members or dimensions have equal solve order, the order in which they appear in the outline (top to bottom) resolves the conflict.

To explore use cases for solve order, see the Solve Order templates in the Technical section of the gallery of application workbooks, which you can find in the files catalog in Essbase.

13

Model Data in Private Scenarios

Using scenario management, scenario participants can perform what-if analysis to model data in their own private work areas. These scenarios can optionally be subject to an approval workflow which includes a scenario owner and one or more approvers. In the workflow, scenario owners merge scenario data with the final cube data only after it is approved.

- [Understand Scenarios](#)
- [Scenario Workflow](#)
- [Enable Scenario Modeling](#)
- [Work with Scenarios](#)

Understand Scenarios

Scenarios are private work areas in which users can model different assumptions within the data and see the effect on aggregated results, without affecting the existing data.

Each scenario is a virtual slice of a cube in which one or more users can model data and then commit or discard the changes.

Scenario-enabled cubes have a special dimension called Sandbox. The sandbox dimension is flat, with one member called Base and up to 1000 other members, commonly referred to as sandbox members. All members in the sandbox dimension are level-0. Sandbox members are named sb0, sb1, and so on. Each sandbox is a separate work area, whereas the Base holds the data currently contained in the cube. A specific scenario is associated with exactly one sandbox member.

```
Sandbox
  Base
  sb0
  sb1
  sb2
```

Base data is the starting point before you use the sandbox to model possible changes. Sandbox data (also known as scenario data) is not committed unless the scenario owner applies it, at which point it overwrites the Base data.

When first created, sandbox member intersections are all virtual and have no physical storage. The physical data from the cube is stored in the Base member slice. Querying new sandbox members dynamically reflects the values stored in the Base.

Only after you update any of the values in a sandbox are your changes stored physically in the sandbox. After you update some values in a sandbox member, queries against the sandbox reflect a mixture of stored sandbox values and values inherited dynamically from the Base.

Changes made in a sandbox are not committed to the Base until you do so explicitly, generally after an approval workflow. See [Understand Scenario User Roles and Workflow](#).

After you're finished with the sandbox, you can put the sandbox through the approval workflow, or you can skip the workflow and commit the updated values to the Base, or reject and discard the sandbox changes.

You must enable hybrid mode for scenario management to work. For queries, it is enabled by default. Do not disable it. For calculations, you also need to enable the HYBRIDBSOINCALCSCRIPT application configuration. See HYBRIDBSOINCALCSCRIPT (or use the SET HYBRIDBSOINCALCSCRIPT calculation command to control it on a per-calculation basis).

Security and filters apply to the Sandbox dimension.

Scenario enabled cubes have a CellProperties dimension that you should ignore, as it is for internal processes. You do not need to modify it nor account for it in calculations, queries, or load rules, and it shouldn't be included in any calculations or other operations.

View and Work with Scenario Data

There are two entry points for viewing and working with scenario data in Smart View.

You can use the Essbase web interface to launch a scenario in Smart View, or you can use a Smart View private connection and work with the scenario data that way.


To analyze data in a scenario, you must have all of the following permissions:

- Be a user provisioned to the application.
- Have a minimum of database access permission for the application (and have a write filter if you want to change data in the sandbox).
- Be a participant in the scenario (created by a user with higher privilege).

View and Work With Scenario Data From the Essbase Web Interface

You can launch Smart View from a scenario in the web interface.

When you do this, because you enter from the scenario, you can only work in Smart View in the sandbox member associated with the scenario from which you entered. The sandbox member is implicit. You will not see it in the Smart View grid.

1. In Essbase, click **Scenarios**.
2. Click the Excel icon  next to the scenario you want to view.
3. Select to open the file.
4. This launches Excel with a Smart View connection to the scenario.

When you do this, the slice of data for that specific scenario is in the worksheet. You can query data only in that scenario. If you have minimum database update permission on the application, you can submit data to the scenario. (When you submit data to a scenario, you are submitting data to one sandbox member).

You can launch a scenario in Smart View from the web interface only on Windows using Firefox, Internet Explorer, or Chrome browsers.

View and Work With Scenario Data From a Smart View Private Connection

You can open Excel and make a private connection to your cube, without starting from the web interface.

When you do this, the sandbox dimension will be in the worksheet, so you can submit data to any sandbox member to which you have access. This is helpful when you are a participant in more than one scenario, but you must explicitly know which sandbox you want to work in.

To see which sandbox member is associated with a scenario, go to the web interface, click on **Scenarios**, click on the scenario name, and view the **General Information** tab.

1. Open Excel.
2. Make a private connection to your scenario-enabled cube.
3. Do an ad hoc analysis.
4. Drill into the Sandbox dimension to view the sandbox members.

Examples

This is a Smart View grid including the Base member and a sandbox member. Sandbox values have not been updated, so they reflect the Base values. Those values are stored only in the Base, not in the sandbox members:

					Base	sb10
Cola	New York	Actual	Jan	Sales	678	678
Cola	New York	Actual	Jan	COGS	271	271

The changed sandbox value below, 500, is stored in a sandbox member. The remaining sandbox value, 271 that was not updated is stored only in the Base:

					Base	sb10
Cola	New York	Actual	Jan	Sales	678	500
Cola	New York	Actual	Jan	COGS	271	271

Below is a grid with multiple sandbox members. If you have the Database Access user role and the appropriate write filter, you can submit data within multiple scenarios simultaneously:

					Base	sb0	sb1
Actual	Jan	Sales	New York	Cola	678	500	600
Actual	Jan	COGS	New York	Cola	271	271	271

About Scenario Calculations

By default, Essbase calculates all members from a dimension unless a fix statement is used to limit the scope of the calculation to a specific member or group of members from the dimension.

The sandbox dimension is an exception to this behavior; if members from the sandbox dimension are not included in the fix for a calculation, only the base member from the sandbox dimension is calculated by default. To calculate non-base members from the sandbox dimension, include them in the fix statement, optionally along with the base member.

When you specify non-base sandbox members in a fix statement, base is excluded from the calculation unless explicitly added into the fix.

This behavior is different from calculations on non-sandbox dimensions excluded from the fix; if you exclude a dimension from your fix statement, Essbase calculates all members from the implied dimension. Sandbox dimensions are calculated differently, as the intent is usually to calculate either Base or specific sandboxes at a given time. Essbase calculates the Base member values, rather than the working sandbox values, except:

- When the calculation fixes on particular sandbox members.
- When the calculation is executed from a sheet launched from a scenario in the web interface (this is called a scenario-launched sheet). See [View and Work With Scenario Data From the Essbase Web Interface](#).
- When a sandbox cell value is selected in a private connection Smart View sheet and a calculation script is launched.

If you execute a calculation script from a scenario-launched sheet, the calculation runs in the sandbox associated with the scenario as long as no sandbox is explicitly mentioned in the script.

If you're in a sheet opened using a Smart View private connection and you're displaying sandbox and base values, if you highlight any data cell from the sandbox and launch a calculation script without explicit sandbox fix, the sandbox will implicitly be calculated and Smart View will indicate that the sandbox was calculated. If you highlight a cell from the base member (or highlight no cell), then the base will be calculated when you launch your calc script and Smart View will indicate that the base was calculated.

You can calculate sandbox members using your pre-existing MAXL scripts by using the reserved runtime substitution variable name: `ess_sandbox_mbr`.

This statement can be implemented (for your sandbox) in any MAXL script without creating any substitution variable on the server or application.

About Data Loads to Scenario-enabled Cubes

You can load scenario-enabled cubes using data exports taken before enabling the cube for scenarios. The data will load to the base sandbox member.

If you didn't use column export, then you can't have outline member changes that would invalidate your data load. If you used column export but your outline has changed, you may need a `.rul` file to load the data.

About Data Exports from Scenario-enabled Cubes

Scenario-enabled cubes have a CellProperties dimension that is for internal purposes, nonetheless this dimension is included in data exports and must be considered when loading exported data. Also, it is important to understand the behavior of the sandbox dimension when working with exported data.

The following are considerations when exporting data from scenario-enabled cubes:

- If you use the web interface **Jobs** page to export data from a scenario-enabled cube, the resulting data file contains all three members from the CellProperties dimension (EssValue, EssStatus, and EssTID). Do not eliminate any of these columns.
- The data file from the export includes data physically stored in the cube, based on the selection you make: level zero data, all data, or input data.
- If values have been changed in sandboxes, then sandbox values will be in your export.
- In order to load exported data into sandboxes, values for all three CellProperties members (EssValue, EssStatus, and EssTID) must be in the data file.

About Transparent and Replicated Partitions in Scenario-enabled Cubes

Transparent and Replicated partitions connect slices from two Essbase cubes together. This is the case when neither, one, or both cubes are scenario-enabled.

Sandboxes come into use when scenarios are created. However, there is no guarantee that scenarios on partitioned cubes will map to the same sandbox number. The same user may not be a participant in sandboxes in multiple cubes. Introducing scenarios imposes the following limitations:

- If source of a transparent partition is scenario-enabled, target queries will always pull data from the source base sandbox member.
- Write-back between scenario-enabled source and target cubes is only allowed between base members in the cubes, target cube base to source cube base. Example: Write-back to source, which is normally enabled from transparent partition target cubes, is disabled for non-base sandbox members of scenario-enabled target cubes. It is a violation of permissions to allow a remote sandbox user to write directly into the base of the source cube.
- For replicated partitions, replication is only possible between source cube base and target cube base.

See [Understand Transparent and Replicated Partitions](#).

About XREF/XWRITE in Scenario-enabled Cubes

In scenario-enabled cubes, you can use XREF and XWRITE to reference or write to data in another cube.

XREF queries a remote cube from a local cube (the cube containing the XREF statement). If the remote cube is scenario-enabled, XREF only pulls base data from the remote cube.

XWRITE updates a remote cube from a local cube (the cube containing the XWRITE statement). Because XWRITE writes data into the remote cube, the scope of the XWRITE statement matters.

For different combinations of scenario-enabled and non-scenario-enabled cubes, XWRITE behaves in the following ways:

When a scenario-enabled local cube references a non-scenario-enabled remote cube,

- A Fix on the base member in the local cube with an XWRITE to the remote cube writes the local cube base into the remote cube.
- No Fix on any sandbox member in the local cube with XWRITE to the remote cube writes the local cube base into the remote cube. If you don't include a sandbox member in the Fix, base is included automatically.
- A Fix on sandbox in the local cube with an XWRITE to the remote cube returns an error. Writing from a non-base sandbox member into a remote cube is not supported.

When a scenario-enabled local cube references a scenario-enabled remote cube,

- A Fix on the base member in the local cube with an XWRITE to the remote cube writes the local cube base into the remote cube base.
- No Fix on any sandbox member in the local cube with an XWRITE to the remote cube writes the local cube base into the remote cube base. If you don't include a sandbox member, base is included automatically.
- A Fix on sandbox in the local cube with an XWRITE to the remote cube returns an error. Writing from a non-base sandbox member into a remote cube is not supported.

When a non-scenario-enabled local cube references a scenario-enabled remote cube, XWRITE always updates the remote cube base member.

See [Understand XREF/XWRITE](#).

About Audit Trail in Scenario-enabled Cubes

Data audit trail tracks updates made to data in a cube. To work with audit trail in scenario-enabled cubes you should understand what defines "old" and "new" data values, and the two different entry points for working with sandbox data in Smart View.

This topic assumes you are familiar with the different entry points for viewing scenario data. See:

- [View and Work With Scenario Data From a Smart View Private Connection](#)
- [View and Work With Scenario Data From the Essbase Web Interface](#)

If you consider the latest data update committed to a cell to be "new" data, and all prior data values for that cell to be "old," it can help you understand how audit trail works in scenario-enabled cubes.

A new or unused sandbox in a scenario-enabled cube contains no stored values. The values shown to users, such as the values displayed in a spreadsheet, reflect the values stored in the base.

If you use data audit trail on a new scenario-enabled cube, the base values that display in the spreadsheet for the sandbox are considered the "old" values.

When you update values in a sandbox, those values are stored in the sandbox (not in the base). For the purposes of data audit trail, these values are the “new” values.

If you later update these “new” values, audit trail will track the latest changes. It will treat the previous values as “old” and the updated values as “new.”

In summary,

- Old values are the base values reflected in a new sandbox.
- Initially, new values are the updated, stored values in the sandbox.
- Subsequently, updated values are new, and the values they replace are old.

There are two possible entry points for working with data in Smart View:

- Where you open Excel and make a private connection to your cube, without starting from the Essbase web interface.
- Where you launch Smart View from a scenario in the web interface.

When you start by opening Excel and making a private connection to your cube, audit trail works as you would expect with any other data set.

When you launch Smart View from a scenario in the Essbase web interface, audit trail works differently.

- When you export logs to a sheet, the sheet does not show the implicit sandbox member.
- When you launch a new sheet using the **Ad hoc** button below the **Audit Trail** pane, the new sheet does not show the implicit sandbox member, and any changes in that sheet affect the data values for that sandbox member.

About Scenario Limitations

These limitations apply to scenarios and sandbox dimensions.

- Scenarios are not supported on aggregate storage cubes.
- The DATAEXPORT calculation command isn't supported on sandbox members. It is only supported on the Base member.
- When you connect to a scenario from a scenario-launched sheet, MDX queries, MDX inserts, and MDX exports will work with the base instead of working with the sandbox for that scenario.
- Runtime substitution variables with the svLaunch parameter are not supported when you launch the scenario in Smart View from the Essbase web interface. See [View and Work With Scenario Data From the Essbase Web Interface](#).

Runtime substitution variables with the svLaunch parameter work correctly when you connect to the scenario directly from a private connection. This is because the sandbox member is included in the sheet.

There are a limited number of functions that are not supported in hybrid mode, which is used with scenario-enabled cubes. See [Functions Supported in Hybrid Mode](#).

Scenario Workflow

You can review a scenario using an optional approval workflow. Alternatively, when working with a scenario, you can change data values in the scenario and commit data changes to the cube (or reject them), without going through an approval process.

Scenario status changes and workflow are affected by the number of participants and approvers for a given scenario. With participants, but no approvers, participants do not have the option to submit the scenario for approval, and there is no option to approve or reject the scenario. With no participants and no approvers, the scenario owner makes the changes and applies them. Again, there is no approval process.

- Scenario with participants but no approvers:
 1. Scenario owner creates the scenario (Status = In Progress)
 2. Scenario owner and participants make changes in Smart View or the web interface
 3. Scenario owner applies changes to base (Status = Applied)
- Scenario with no approvers and no participants
 1. Scenario owner creates the scenario (Status = In Progress)
 2. Scenario owner makes changes in Smart View or the web interface
 3. Scenario owner applies changes to base (Status = Applied)
- Scenario with participants and approvers
 1. Scenario is created by owner (Status = In Progress)
 2. Scenario owner, participants and approvers can make changes in Smart View or the web interface
 3. Scenario owner submits the scenario for approval (Status = Submitted)
 4. Scenario is either approved by all approvers or rejected by one or more approvers (Status = Approved or Status=Rejected)
Rejected status is the same as In Progress status, in that all participants can make changes to reach approved status.
 5. After the scenario reaches approved status (all approvers have approved the scenario), then the scenario owner applies the changes to the base (Status=Applied).
- [Enable Email Notifications for Scenario Status Changes](#)
- [Create a Scenario](#)
- [Model Data](#)
- [Submit a Scenario for Approval](#)
- [Approve or Reject Scenario Changes](#)
- [Apply or Discard Data Changes](#)
- [Copy a Scenario](#)
- [Delete the Scenario](#)
- [Understand Scenario User Roles and Workflow](#)

Enable Email Notifications for Scenario Status Changes

If the system administrator has enabled outgoing emails from Essbase, then the appropriate scenario participants receive email notifications for scenario changes.

To set up SMTP email notifications:

1. Log in to Essbase as a system administrator.
2. Click **Console**.
3. Select **Email Configuration**.
4. Select the SMTP Configuration tab.
SMTP controls outgoing email.
5. Enter your company's SMTP host and port.
6. Enter your company email address and password, of the sender of the notification email.
7. Click **Save**.

When SMTP mail is set up, scenario participants begin receiving emails when their scenarios change status, ownership, priority, or due date.

When users are added to the system, email is an optional field. If it has not been filled out, then that user cannot receive emails even if they participate in scenarios.


Scenario State	Email to	Email Cc	Email Subject
Create scenario	Participant, approver	Owner	You are invited to participate in scenario <scenario name>
Submit	Approver	Owner, participant	Scenario <scenario name> is submitted for approval
Approve	Owner	Participant, approver	Scenario <scenario name> is approved
Reject	Owner	Participant, approver	Scenario <scenario name> is rejected by <user>
Apply	Participant	Owner, approver	Scenario <scenario name> is updated
Delete	Participant, approver, owner	Deleting user	Scenario <scenario name> is deleted
<i>Update action</i> Can be a change in ownership, priority, or due date.	Participant, approver	Owner	Scenario <scenario name> is updated

An existing scenario can be updated (see *Update action* in the table) to change the owner, the priority, or the due date. If, for example, the scenario's due date is changed, then the participants will receive an email indicating the new due date. The old due date will appear in strike through text, so that it is clear what information about the scenario was updated.

Create a Scenario

To create a scenario, you specify general information about your scenario, including creating a scenario name, selecting a due date, selecting an application and cube, and choosing whether to use calculated values. Then you add users and define whether each user is a participant or an approver.


To create a scenario you must:

- Be a user provisioned to the application or be the application's owner.
 - Have database update permission.
1. In Essbase, login as a user with database update (or higher) permission to at least one application.
 2. Click **Scenarios**.
 3. Click **Create Scenario**.
 4. On the **General Information** tab, enter a scenario name and select a **Priority** (optional), **Due Date**, **Application**, and **Database** (cube). You will only see applications for which you have minimum database update permission.
 5. Turn on **Use Calculated Values** if you want to merge calculated values to base values when running calculation scripts on scenarios.
 6. (Optional) Enter a description.
 7. On the **Users** tab, click **Add**  for a list of users.
 8. Add the users that you want.
 9. Close the **Add Users** dialog box.
 10. For each user, keep the default (**Participant**), or select **Approver**.
Scenario user roles determine the workflow for the scenario.
 11. Save your changes.

See also: [Understand Scenario User Roles and Workflow](#).

Model Data

As a scenario user, you can model data slices in your own scenario.

1. In Essbase, click **Scenario**.
2. On the Scenarios page, locate the scenario in which you want to model data.
 - You can search for the scenario by name in the **Search** field.
 - You can select your application from the **All Applications** drop-down list and search within that application.
 - After selecting the application, you can further narrow your search by selecting the database (cube) from the **All Databases** drop-down list and searching within that specific cube.
3. Launch Smart View by clicking the **Excel**  icon before the scenario name.
4. Make data changes and perform your what-if analysis in Smart View.

If you change and submit values and decide you want to go back to the base values, you can revert to the base by typing #Revert in the changed cells and choosing **Submit Data** on the Smart View Essbase ribbon.

If a cell in the base has a value, and you want the corresponding cell in the scenario to be #Missing, you can send #Missing to the scenario or you can delete the value in Smart View and select **Submit Data** on the Smart View Essbase ribbon.

5. Continue this process until you're ready to submit data for approval.

If a calculation has been run on a sandbox and the changes are not acceptable, request from your application designer a calc script to revert the changes, or request a new sandbox.

Submit a Scenario for Approval



After you submit a scenario for approval, no one will be able to write to that scenario.

1. In the Essbase web interface, log in as the application owner or the scenario owner.
2. Click **Scenarios**.
3. Click the **Submit** → arrow under **Actions**.
4. (Optional) Enter a comment.
5. Click **OK**.

After a scenario is submitted for approval, the scenario approver can approve or reject the data changes.

Approve or Reject Scenario Changes

After the owner of the scenario submits for approval, the approver has the option to approve or reject scenario changes, and the scenario owner is notified of the action. You must be logged in as an approver to have the options to approve or reject a scenario.

1. In the Essbase web interface, click **Scenarios**.
2. Next to the submitted scenario, under **Actions**, click **Approve**  or **Reject** .
3. Enter a comment on the Approve or Reject dialog box.

After a scenario is approved, the scenario owner can apply the changes to the cube.

Apply or Discard Data Changes

When you apply data changes, the changes stored within the scenario overwrite the base data.

You can apply or discard changes from the Scenario page.

1. In the Essbase web interface, click **Scenarios**.
2. Next to the approved scenario, under **Actions**, click **Apply** or **Discard**.
3. When prompted, confirm your selection.

- You can also apply data changes using the DATAMERGE calculation command.
- After a scenario is applied or discarded, you can delete the scenario to reuse the sandbox for that scenario.
- Database managers and higher can execute a calculation script to perform a DATAMERGE. They do not need to be designated as scenario approvers in order to do so.
- After a scenario is applied, it can be re-applied, but it cannot be changed.

Copy a Scenario

If you have the service administrator role, or if you are a scenario user (participant, approver, or owner), you can copy a scenario. You can copy scenarios at any point in the scenario workflow, prior to Delete Scenario. The approval state of the copied scenario is reset to In Progress.

1. In the Essbase web interface, click **Scenarios**.
2. Click the **Actions** menu for the scenario you want to copy, and click **Copy**.
3. Enter the scenario name and select what scenario components to copy from **Approvers, Participants, Comments, and Data**.
4. Click **OK**.

Delete the Scenario

Since there are a fixed number of available sandboxes in a cube, you may need to free up sandboxes from inactive scenarios. After the associated scenario is deleted, the sandbox is empty and is automatically returned to the pool of available sandboxes.

To reuse a sandbox associated with a scenario, you need to delete the scenario.

1. In the Essbase web interface, click **Scenarios**.
2. Click the **Actions** menu for the scenario you want to delete, and click **Delete**.

Understand Scenario User Roles and Workflow

You can review a scenario using an optional approval workflow.

Scenario user role assignments determine the workflow for scenarios. You must have at least one approver to enable the scenario workflow. Without an approver, participants do not have the option to submit the scenario for approval, for example, and there is no option to approve or reject the scenario.

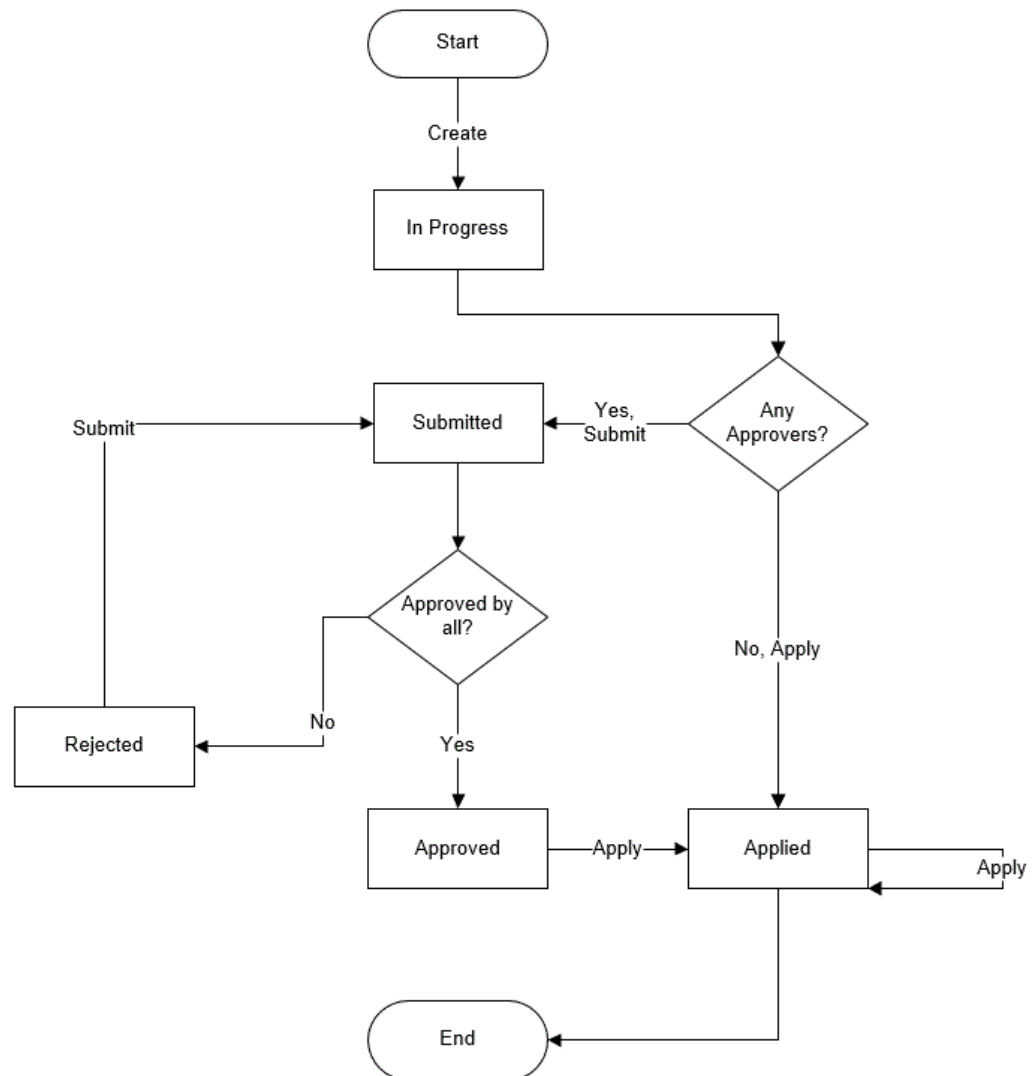
The only action for scenarios without at least one approver is Apply. Without an approver, the scenario owner can still change data values in the scenario and apply data changes to the cube (or reject them), without going through an approval process.

Participants can participate in a what-if analysis. They must have Database Update or Database Access user role. Adding participants is not mandatory.

Approvers monitor the process, and approve or reject scenarios. They must have Database Access or higher role. Scenarios can have multiple approvers, in which case each one must approve the scenario before it can be submitted.

Participants and approvers with the Database Access user role cannot write to a scenario until they are granted write access through a filter.

Participants and approvers are not mandatory. The scenario owner can change data values in the scenario and commit data changes to the cube (or reject them) without designating participants or approvers.



Enable Scenario Modeling

Enabling scenario modeling as part of the cube creation process is as easy as selecting a check box in the user interface or populating the right fields in an application workbook.

You can create or enable a cube for scenario modeling using one of the following methods:

- [Create a Scenario-Enabled Cube](#)
- [Create a Scenario-Enabled Sample Cube](#)

- [Enable an Existing Cube for Scenario Management](#)
- [Create Additional Sandbox Members](#)

Data Audit Trail is not supported on scenario-enabled cubes.

Create a Scenario-Enabled Cube

Scenario-enabled cubes have specialized dimensions required to use scenario management. These include the Sandbox dimension and the CellProperties dimension. CellProperties is considered a hidden dimension in that you do not need to interact with it in any way when performing Essbase tasks such as building cubes, loading data, or calculating cubes.

1. On the Applications home page, click **Create Application**.
2. On the Create Application dialog box, enter an **Application Name** and a **Database Name** (cube name) and expand **Advanced Options**.
3. Ensure that in **Database Type, Block Storage (BSO)** is selected.
4. Select **Enable Scenarios**.
5. Click **OK**.

Create a Scenario-Enabled Sample Cube

You can create a scenario-enabled cube by importing the scenario-enabled sample application workbook.

1. In the Essbase web interface, click **Import**.
2. Click **Catalog**.
3. Drill down into the **Gallery, Cubes**, and **General** folders.
4. Select **Sample_Basic_Scenario.xlsx** and click **Select**.
5. Provide a unique name and click **OK**.

Enable an Existing Cube for Scenario Management

If you have the application manager role, you can enable an existing cube to use scenario modeling. It is best to do so on a copy of the original cube. Existing scripts, rules, and queries will work as before, on the base member. If you need to run them on a sandbox member, you can run them from a scenario-launched sheet.

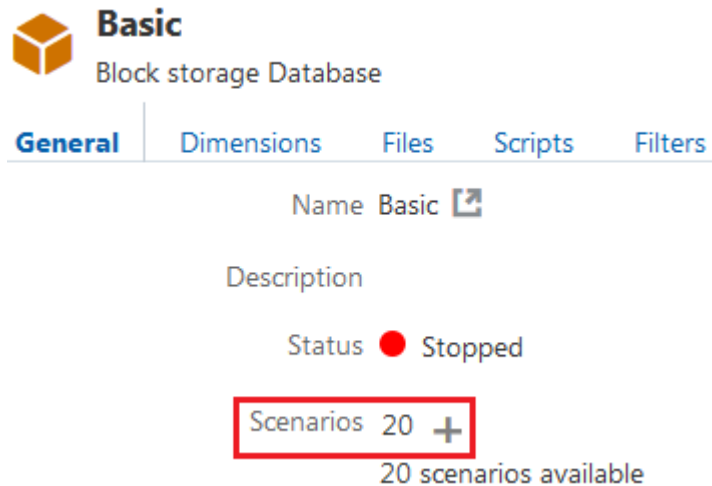
A scenario launched sheet is an Excel sheet launched from a scenario in the web interface. See [View and Work With Scenario Data From the Essbase Web Interface](#) .

1. On the Applications page, expand the application.
2. Click the **Actions** menu to the right of the cube name and select **Inspect**.
3. On the **General** tab, for **Scenarios**, click **Not Enabled**.
4. Adjust the number of scenarios members (non-base sandbox members) you want to create and click **Ok**.

Create Additional Sandbox Members

By default, a new scenario-enabled cube has 100 sandbox members. You can create additional sandbox members (up to 1000).

1. On the Applications page, expand the application.
2. Click the **Actions** menu to the right of the cube name, and select **Inspect**.
3. On the **General** tab, click the plus sign next to **Scenarios**.



4. Enter the number of sandbox members you want to create.
5. Click **OK**.

Work with Scenarios

After you create a scenario-enabled cube, you can create scenarios and follow a workflow that includes modeling data, approving or rejecting changes, applying or discarding changes, and submitting the scenario for approval.

- [View Base Member Data](#)
- [Compare Scenario Values to Base Values](#)
- [Set Scenario Cells to #Missing](#)
- [Revert Scenario Values Back to Base Values](#)
- [Understand When to Aggregate Sandbox Dimensions](#)

View Base Member Data

From the web user interface, you can launch an Excel sheet showing base data for a scenario.

1. In Essbase, click **Scenarios**.
2. Click the **Actions** menu for the scenario you want to view, and click **Show Base Data**.

- Click on the downloaded link to launch Smart View.

The Excel sheet that is launched shows base data for the cube. It does not show sandbox data.

Compare Scenario Values to Base Values

If you are the owner, approver or participant for a given scenario, you can view scenario and base values in a spreadsheet or in the web user interface to compare models.

Compare Values in Excel

- In the Essbase web interface, click **Scenarios**.
- From the **Actions** menu, select **Show Changes in Excel**.
- Click on the downloaded link to open the Smart View link.
- You can view values for both the scenario and base members in the spreadsheet.

	A	B	C	D	E	F	G
1						Base	sb10
2	Cola	New Yo	Actual	Jan	Sales	678	700
3	Cola	Massac	Actual	Jan	Sales	494	500
4	Cola	Florida	Actual	Jan	Sales	210	250
5	Cola	Connec	Actual	Jan	Sales	310	350
6	Cola	New Ha	Actual	Jan	Sales	120	150
7	Cola	East	Actual	Jan	Sales	1812	1950

- In column G, sb10 is the scenario (or sandbox) member.
- In column F, Base shows the base values.
- In the scenario, values for sb10 on rows 2 through 6 have been changed, and you can see the aggregated result in row 7.

Compare Values in the Web User Interface

- In the Essbase web interface, click **Scenarios**.
- From the **Actions** menu, select **Show Changes**.

The **Data Changes** dialog box is empty if no data changes have been made.

Compare the scenario to the base in order to determine your next steps. For example, you might choose to change the status of the scenario to approved based on this information.

Set Scenario Cells to #Missing

You can set scenario cells to #Missing even if the corresponding base cells have values.

To set a scenario cell to #Missing:

- Type #Missing in the cell or delete the cell contents.

2. Select **Submit Data** on the Smart View ribbon.

Example

1. Initially, the value in sb1 is an exact mirror of the value in the base.

				Base	sb1
				Jan	Jan
100-10	New York	Sales	Actual	678	678

2. Enter #Missing in sb1 (or delete the cell contents) and submit data.

				Base	sb1
				Jan	Jan
100-10	New York	Sales	Actual	678	#Missing

3. Refresh the sheet. See that sb1 is #Missing.

				Base	sb1
				Jan	Jan
100-10	New York	Sales	Actual	678	#Missing

Revert Scenario Values Back to Base Values

Initially, scenario values are not stored and they are an exact mirror of the base values. After you change the scenario values in Excel and submit the changes to the cube, the scenario values are stored, and they are different from the base. You can revert the scenario values back to the base by typing #Revert in the changed cells and clicking **Submit Data** on the Smart View ribbon.

To revert scenario values back to the base:

1. In Excel, type #Revert in the scenario cells you want to revert to the base.
2. Click **Submit Data** on the Smart View ribbon.

The selected scenario values are updated to the base values.

Example

1. Initially, the value in sb1 is an exact mirror of the value in base.

				Base	sb1
				Jan	Jan
100-10	New York	Sales	Actual	678	678

2. Submit a new value, 100, to sb1.

				Base	sb1
				Jan	Jan
100-10	New York	Sales	Actual	678	100

- Submit #Revert to sb1.

				Base	sb1
				Jan	Jan
100-10	New York	Sales	Actual	678	#Revert

- Refresh the sheet. See that sb1 again reflects the base value of 678.

				Base	sb1
				Jan	Jan
100-10	New York	Sales	Actual	678	678

Understand When to Aggregate Sandbox Dimensions

As you model in scenarios, you will need to determine whether or not to calculate within each sandbox.

Submit data changes to the sandbox and calculate as little other data as possible, just enough data to allow users to validate their work. This preserves the storage efficiency of the sandbox design.

For example, when all upper level members in a cube are dynamic calc, aggregations in the form of calculation script are not needed.

If you have stored upper level members, limit the scope of any sandbox calculation to the minimum needed for users to do their work.

Example: Calculate Scenarios with Dynamic Upper Level Members

Dynamic hierarchies (both dense and sparse) aggregate automatically, and users making changes in sandboxes see their changes immediately.

Let's look at an example from the Sample_Scenario.Basic block storage demo application.

Assume that Product and Market are dynamic hierarchies with data stored only at level zero, and that a scenario is created using sandbox dimension member sb0.

When the sandbox is newly created, values for sb0 are the same as the values for Base. This is because sandbox members are virtual, reflecting base values until users submit changes to them.

	A	B	C	D
1			Budget	Budget
2			Sales	Sales
3			Jan	Jan
4			Base	sb0
5	California	Cola	840	840
6	Oregon	Cola	200	200
7	Washington	Cola	160	160
8	Utah	Cola	160	160
9	Nevada	Cola	90	90
10	West	Cola	1450	1450

After modifying Sales->Budget->Jan->Cola data in member sb0, we immediately see that the dynamic sandbox member, West (in D10) aggregates to the correct total by using a combination of stored members from Base and sb0.

Values for Oregon, Utah and Nevada are stored in the Base sandbox member. Values for California and Washington have been submitted by scenario participants and are stored in the sb0 sandbox member. The total for West->Cola->sb0 aggregates dynamically using these stored values.

	A	B	C	D
1			Budget	Budget
2			Sales	Sales
3			Jan	Jan
4			Base	sb0
5	California	Cola	840	900
6	Oregon	Cola	200	200
7	Washington	Cola	160	200
8	Utah	Cola	160	160
9	Nevada	Cola	90	90
10	West	Cola	1450	1550

You can also use calculation scripts in sandboxes. Assume that Oregon is meant to be budgeted as 80% of California. The following calculation script can do this:

```
FIX("Jan", "Budget", "Cola", "Sales")
"Oregon"="California"*.8;
ENDFIX
```

When a scenario participant launches an Excel worksheet from the web interface and runs this calculation, sb0 is the default sandbox member calculated and the value for member Oregon is updated:

	A	B	C	D
1			Budget	Budget
2			Sales	Sales
3			Jan	Jan
4			Base	sb0
5	California	Cola	840	900
6	Oregon	Cola	200	720
7	Washington	Cola	160	200
8	Utah	Cola	160	160
9	Nevada	Cola	90	90
10	West	Cola	1450	2070

This view is not from a scenario-launched sheet, but rather from a Smart View private view, where Base and sb0 can both be represented on the sheet.

Example: Calculate Scenarios with Stored Upper Level Members

In some cases, a sparse or dense hierarchy may have stored upper level members, and aggregations on level- or generation-based calculations could be required.

Continuing from the last grid of the previous example, assume now that upper level members in the Market dimension are stored, rather than dynamic.

If we change the value for Oregon to 250, the West member will need to be re-calculated before we can see correct results:

	A	B	C	D
1			Budget	Budget
2			Sales	Sales
3			Base	sb0
4			Jan	Jan
5	California	Cola	840	900
6	Oregon	Cola	200	250
7	Washington	Cola	160	200
8	Utah	Cola	160	160
9	Nevada	Cola	90	90
10	West	Cola	1450	2070

The following calc script can be used to aggregate the Market dimension in the sandbox, when executed from a scenario-launched excel sheet:

```
AGG("Market");
```

	A	B	C	D
1			Budget	Budget
2			Sales	Sales
3			Jan	Jan
4			Base	sb0
5	California	Cola	840	900
6	Oregon	Cola	200	250
7	Washington	Cola	160	200
8	Utah	Cola	160	160
9	Nevada	Cola	90	90
10	West	Cola	1450	1600

Work with Cubes in Cube Designer

You can create or modify application workbooks and then deploy cubes to Essbase using Cube Designer, a Smart View extension.

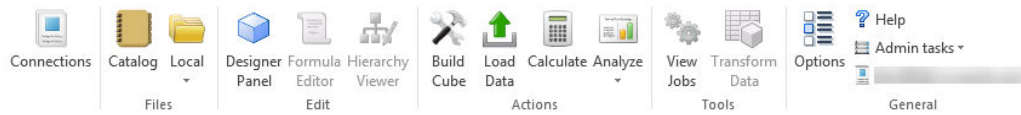
- [About Cube Designer](#)
- [About the Cube Designer Ribbon](#)
- [About the Designer Panel](#)
- [Manage Files in Cube Designer](#)
- [Download Sample Application Workbooks](#)
- [Build a Private Inventory of Application Workbooks](#)
- [Work with Application Workbooks in Cube Designer](#)
- [Create a Cube from Tabular Data in Cube Designer](#)
- [Update Cubes Incrementally in Cube Designer](#)
- [Create and Validate Member Formulas in Cube Designer](#)
- [Load Data in Cube Designer](#)
- [Calculate Data in Cube Designer](#)
- [Work with Jobs in Cube Designer](#)
- [View Dimension Hierarchies in Cube Designer](#)
- [Export Cubes to Application Workbooks in Cube Designer](#)
- [Delete Applications and Cubes in Cube Designer](#)
- [Unlock Objects in Cube Designer](#)
- [View Logs in Cube Designer](#)

About Cube Designer

The basic components of Cube Designer are the Cube Designer ribbon and the Designer Panel. See [About the Cube Designer Ribbon](#) and [About the Designer Panel](#).

About the Cube Designer Ribbon

Cube Designer helps you to design, create and modify application workbooks to meet their strict layout and syntax requirements. You can also use options on the Cube Designer ribbon to perform a number of cube management tasks, such as loading data, editing formulas and viewing jobs.



Cube Designer Ribbon Options

- **Connections:** Opens the Connections dialog box, in which you choose the Essbase URL.
- **Catalog:** Opens the Essbase Files dialog box, which contains a selection of prebuilt application workbooks, from which you can build sample applications and cubes.
Also, a catalog toolbar is available in this dialog box from which that you can perform many file operations within the catalog, such as upload, download, cut, copy, paste, delete, rename, and create a new folder.
- **Local:** Provides a drop-down menu with options to open or save an application workbook locally, or to export a cube to an application workbook.
- **Designer Panel:** Opens the Designer Panel, a series of panels in which you can design and edit application workbooks.
- **Formula Editor:** Opens the Formula Editor, which provides an interface in which to develop member formulas, with assistance for developing correct syntax.
- **Hierarchy Viewer:** Opens the Dimension Hierarchy dialog box, in which you can view the hierarchy for the selected dimension worksheet in an application workbook, and perform tasks, such as renaming members and changing storage settings. See [Work with Dimension Worksheets in Cube Designer](#).
- **Build Cube:** Opens the Build Cube dialog box, where you can build a cube from the active application workbook. In this dialog box, cube designer automatically detects existing data and calculation worksheets, and then pre-selects options to load the data and run the worksheets.
- **Load Data:** Opens the Load Data dialog box, which contains options to clear all data and to load data.
- **Calculate:** Opens the Calculate Data dialog box, in which you can select an application, a cube, and a calculation script to execute.
- **Analyze:** Provides a drop-down menu with options to create a Smart View ad hoc grid, or connect application workbook query worksheets (*Query.query_name* worksheets) to Smart View.
- **View Jobs:** Opens the Job Viewer dialog box, in which you can monitor the status of jobs, such as data loads, calculations, imports, and exports.
- **Transform Data:** Opens the Transform Data dialog box, which lets you build a cube from tabular data.
- **Options:** Provides options to specify the default working folder and to activate the cube designer log.
- **Admin Tasks:** Opens a menu from which you can delete an application, delete a cube, or view logs. Selecting one of these options opens the Delete Application or Delete Cube dialog box, or allows you to view server or application logs.
- **Server name:** Shows the currently defined connection location. When you click **Server name** and log in (if prompted to do so), the server name and the client and server versions are displayed.

About the Designer Panel


The Designer Panel uses a manual system of reading and writing to the worksheets in an application workbook. The **From Sheet** button at the bottom of the Designer Panel reads the entire application workbook's data and populates the panel with the data. The **To Sheet** button updates the entire application workbook with the data from the Designer Panel. The **Reset** button clears the data from the Designer Panel.

One common use of panel is to populate it with information from one application workbook using **From Sheet**, open a new blank workbook, and then use **To Sheet** to make a clone of the first application workbook.

You can design and edit application workbooks in the Designer Panel. Each of its five tabs correspond to one of the five types of worksheets in an application workbook. See [Design and Create Cubes Using Application Workbooks](#).



To open the panel, click **Designer Panel** on the Cube Designer ribbon.

If the Smart View panel displays when you click **Cube Designer**, then click **Switch To**  , and select **Cube Designer** from the drop down menu.

The Designer Panel contains the following tabs:

- **Cube:** You can design and modify the Essbase.Cube worksheet in an application workbook.
See [Work with the Essbase.Cube Worksheet in Cube Designer](#).
- **Settings:** You can design and modify the Cube.Settings worksheet in an application workbook.
See:
 - [Work with the Cube.Settings Worksheet: Alias Tables in Cube Designer](#).
 - [Work with the Cube.Settings Worksheet: Properties in Cube Designer](#).
 - [Work with the Cube.Settings Worksheet: Dynamic Time Series in Cube Designer](#).
 - [Work with the Cube.Settings Worksheet: Attribute Settings in Cube Designer](#).
 - [Work with Text Lists Worksheets in Cube Designer](#).
- **Dimensions:** You can design and modify the Dim.*dimname* worksheets in an application workbook.
See [Work with Dimension Worksheets in Cube Designer](#).
- **Data:** You can design and modify the Data.*filename* worksheet in an application workbook.
See [Work with Data Worksheets in Cube Designer](#).
- **Calc:** You can design and modify the Calc.*scriptname* worksheet in an application workbook.
See [Work with Calculation Worksheets in Cube Designer](#).

Manage Files in Cube Designer

Your access to view and work with Cube Designer files depends on your permissions.

In Cube Designer, you access the file folders in the Catalog using the **Catalog** option in the Cube Designer ribbon.

The **Applications** folder requires Database Manager role access to view cubes for which you have permission.

The **Gallery** folder is read-only access for all users.

The **Shared** folder is read-write access for all users.


The **Users** folder is read-write access for the logged in user.

According to your permissions, you can create, move, rename and delete custom folders. Similarly, users with access can import, export, copy, move, rename and delete files.

Related topic: [Manage Essbase Files and Artifacts](#)

Download Sample Application Workbooks

Using the sample application workbooks provided in the Essbase Files dialog box, you can quickly create sample applications and cubes. The cubes are highly portable, because they are quickly and easily imported and exported.

1. On the Cube Designer ribbon, click **Catalog**  .
2. If prompted to connect, enter your user name and password.
3. On the Essbase Files dialog box, choose the sample application workbook you want to open.

You can then edit the application workbook to fit your requirements in the Designer Panel. See [Work with Application Workbooks in Cube Designer](#).

You can save this modified application workbook to your private inventory. See [Build a Private Inventory of Application Workbooks](#).

You can upload this modified application workbook to either the user or shared catalog locations. If uploaded to the shared catalog location, the application workbook will be available to all users.


Build a Private Inventory of Application Workbooks

Cube Designer allows you to create and store application workbooks on the client computer. This lets you keep a private inventory of completed and in-progress application workbooks.

Using the Local icon menu items on the Cube Designer ribbon, you can manage your private application workbook inventory:


Open an Application Workbook

Open an existing application workbook from your inventory.

1. On the Cube Designer ribbon, click **Local** .
2. Select **Open Application Workbook**.
3. Browse to the application workbook and click **Open**.


Save an Application Workbook

Save a new or updated application workbook to your inventory.

1. Open the application workbook.
2. On the Cube Designer ribbon, click **Local** .
3. Select **Save Application Workbook**.
4. Browse to your inventory location and click **Save**.

Export to an Application Workbook

Export a cube to an application workbook and add it to your inventory.

1. On the Cube Designer ribbon, click **Local** .
2. Select **Export Cube to Application Workbook**.
3. If prompted to log in to Essbase, enter your user name and password.
4. In the **Export Cube** dialog box, select the application and cube you want to export, and from the **Export Build Method** menu, select either the **Parent-Child** or **Generation** build method; indicate if you want to export input level data and calculation scripts, and click **Run**.
5. To add the application workbook to your private inventory, click **Save Application Workbook**.

Work with Application Workbooks in Cube Designer

Using Designer Panel, you can modify an application workbook, and then you can use the modified workbook to create an updated cube, reflecting your changes.

- [Limitations of Application Workbooks](#)
- [Work with the Essbase.Cube Worksheet in Cube Designer](#)
- [Work with the Cube.Settings Worksheet: Alias Tables in Cube Designer](#)
- [Work with the Cube.Settings Worksheet: Properties in Cube Designer](#)
- [Work with the Cube.Settings Worksheet: Dynamic Time Series in Cube Designer](#)
- [Work with the Cube.Settings Worksheet: Attribute Settings in Cube Designer](#)

- [Work with the Cube.Settings Worksheet: Substitution Variables in Cube Designer](#)
- [Work with Dimension Worksheets in Cube Designer](#)
- [Work with Data Worksheets in Cube Designer](#)
- [Work with Calculation Worksheets in Cube Designer](#)
- [Work with MDX Worksheets in Cube Designer](#)
- [Create a Cube from a Local Application Workbook in Cube Designer](#)
- [Work with Text Lists Worksheets in Cube Designer](#)

Limitations of Application Workbooks

Current limitations for using application workbooks, are listed here.

The following limitations currently exist when working on application workbooks in Excel using the designer panel.

- You cannot set up a dimension worksheet using the generation format. Instead, you must import using the parent-child build method.
- Multiple dimension sheets for the same dimension are not supported. You are limited to one worksheet per dimension.
- Application workbooks do not support aggregate storage cubes.
- Changes to the Cube.Settings worksheet cannot be applied incrementally. Instead, you must rebuild the cube to apply those changes.

Work with the Essbase.Cube Worksheet in Cube Designer

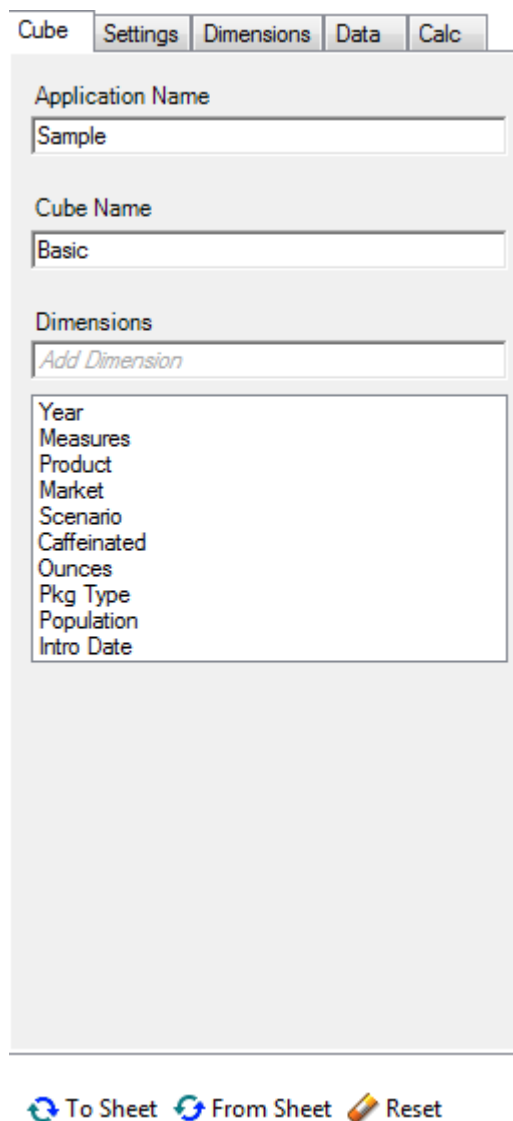
Using the Cube tab in the Designer Panel, you can modify the following fields on the Essbase.Cube worksheet:

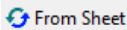
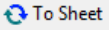
- Application Name
- Cube Name
- Dimension Definitions

You can change the application name and cube name, and delete one or more dimensions.

1. On the Cube Designer ribbon, select **Designer Panel**
2. In the Designer Panel, select the **Cube** tab.



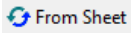
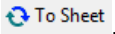


3. Select **From Sheet**  to populate the Designer Panel with the contents of the application workbook.
4. Change the application name or the cube name, if you want to.
5. Add one or more dimensions by typing the name in the text box and pressing the enter key after each one.
6. In the Dimensions list
 - If you want to delete a dimension, right click the dimension name and select **Delete Dimension**.
Alternatively, you can select a dimension name and press the delete key.
 - If you want to rename a dimension, right click the dimension name and select **Rename Dimension**.
7. Select **To Sheet**  to propagate the changes to the application workbook.
8. Examine the updated application workbook to see your changes.

See also: [Understand the Essbase.Cube Worksheet](#).

Work with the Cube.Settings Worksheet: Alias Tables in Cube Designer

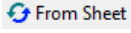
You can add new alias tables in the Cube.Settings worksheet.

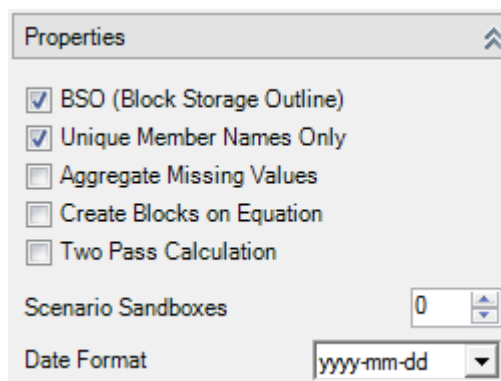
1. In the Designer Panel, select the **Settings** tab.
2. Select **From Sheet**  to populate the Designer Panel with the contents of the application workbook.
3. In the **Alias Tables** field, enter a name for the new alias table.
4. Press **Enter**.
5. Select **To Sheet** .

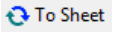
A new alias table name is added on the Cube.Settings worksheet in the application workbook. To add the alias table to a dimension worksheet, open the Dimensions tab in the Designer Panel, and add the alias table to the selected dimension worksheet. See [Work with Dimension Worksheets in Cube Designer](#). After you add the alias table to the dimension worksheet, you must populate the aliases manually, or by copying from a source.

Work with the Cube.Settings Worksheet: Properties in Cube Designer

You can add new properties in the Cube.Settings worksheet.

1. In the Designer Panel, select the **Settings** tab.
2. Select **From Sheet**  to populate the Designer Panel with the contents of the application workbook.
3. Expand the **Properties** section.

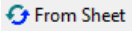


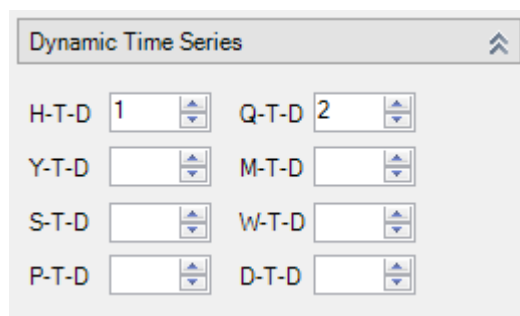
4. Make your selections.
5. Select **To Sheet**  to propagate the changes to the application workbook.

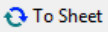
See also: [Understand the Cube.Settings Worksheet: Properties](#).

Work with the Cube.Settings Worksheet: Dynamic Time Series in Cube Designer

You can add dynamic time series members in the Cube.Settings worksheet.

1. In the Designer Panel, select the **Settings** tab.
2. Select **From Sheet**  to populate the Designer Panel with the contents of the application workbook.
3. Expand the **Dynamic Time Series** section.



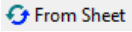
4. Make the changes that you want.
5. Select **To Sheet**  to propagate the changes to the application workbook.

There are reserved generations names used by dynamic time series. For example, using the generation name of “Year” activates dynamic time series for “Y-T-D.”

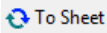
See also: [Understand Dimension Worksheets](#).

Work with the Cube.Settings Worksheet: Attribute Settings in Cube Designer

You change attribute settings on the Cube.Settings worksheet.

1. In the Designer Panel, select the **Settings** tab.
2. Select **From Sheet**  to populate the Designer Panel with the contents of the application workbook.
3. Expand the **Attribute Settings** section.

Attribute Settings	
Dimension Name	Attribute Calculations
Sum Member	Sum
Count Member	Count
Minimum Member	Min
Maximum Member	Max
Average Member	Avg
True Member	TRUE
False Member	FALSE
Attribute Date Format	Month First (mm-dd-)
Prefix/Suffix Value	Parent
Prefix/Suffix Format	Prefix
Prefix/Suffix Separator	_ Underscore
Numeric Ranges	Tops of Ranges

4. Make the changes that you want.
5. Select **To Sheet**  to propagate the changes to the application workbook.
See also: [Understand the Cube.Settings Worksheet: Attribute Settings](#).

Work with the Cube.Settings Worksheet: Substitution Variables in Cube Designer

You can add cube-level substitution variables on the Cube.Settings worksheet.

Enter the name of the substitution variable in column A. Enter the corresponding value of the substitution variable in column B.

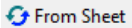
You must enclose member names in double quotes.

Substitution Variables	
CurMonth	"Jan"

Work with Dimension Worksheets in Cube Designer

1. In the Designer Panel, select the **Dimensions** tab.

The screenshot shows the 'Dimensions' tab of the Cube Designer interface. It features several dropdown menus and a list of checkboxes for configuring a dimension. The 'Dimension' dropdown is set to 'Year', 'Dimension Type' to 'Time', 'Dimension Storage Type' to 'Dense', 'Build Method' to 'PARENT-CHILD', and 'Incremental Mode' to 'Merge'. Below these is an 'Update Generation Worksheet' button. A 'Custom Properties' section is collapsed, and a 'Dimension Build Fields' section is expanded, showing a list of checkboxes: Member ID, Storage Type (checked), Consolidation Operator, Two Pass Calculation, Solve Order, Time Balance (with a sub-option 'Skip Value'), Expense Reporting, Comment, Formula, and User Defined Attribute.

2. Select **From Sheet**  to populate the Designer Panel with the contents of the application workbook.
3. Make the changes that you want.

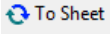
For descriptions of the options and valid values, see [Understand Dimension Worksheets](#).

4. (Optional) If you want to update the Cube.Generations worksheet in the application workbook for this dimension, click the **Update Generation Worksheet** button.

The **Update Generation Worksheet** button creates a section in the Cube.Generations worksheet for the dimension selected in the **Dimension** drop down list on the **Dimensions** tab of the Designer Panel.

The Dimension section of the Cube.Generations worksheet changes if you add or delete members on the dimension worksheet (*Dim.dimname*), causing the number of generations in the dimension to change. If you make changes to the dimension

worksheet by adding or deleting members, you should always press the **Update Generation Worksheet** button as part of the editing process.

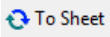
5. Select **To Sheet**  to propagate the changes to the application workbook.
 - After adding alias tables using Designer Panel, populate the alias table column with alias names manually, or by copying them from a source.
 - Use no more than 1024 characters when naming dimensions, members, or aliases.
 - The length limit for the dimension worksheet is 30 characters, including 3 characters for the "Dim." at the beginning of the sheet name. So, the name following "Dim." can contain up to 27 characters.

See [Understand the Cube.Generations Worksheet](#).

Work with Data Worksheets in Cube Designer

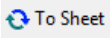
You can create data worksheets in the Designer Panel. You can also edit the display of dimensions and members in existing data worksheets.

To create a new data worksheet:

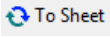
1. In the Designer Panel, select the **Data** tab.
2. Enter a name for the new data worksheet in the **Data Sheets** field.
3. Press **Enter**.
4. Select **To Sheet** .

A new data worksheet is created in the application workbook.

To change the order of dimensions in the data worksheet:

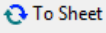
1. In the Designer Panel, select the **Data** tab.
2. In **Data Sheets**, select the sheet that you want to edit.
3. In **Dimension Column Order**, select the dimension that you want to move.
4. Use the up and down arrows to move the dimension.
5. Select **To Sheet**  to add your changes to the selected **Data** tab in the worksheet.

To change the order of members on the data worksheet:

1. In the Designer Panel, select the **Data** tab.
2. In **Data Columns**, select the member that you want to move.
3. Use the up and down arrows to move the member.
4. Select **To Sheet**  to add your changes to the selected **Data** tab in the worksheet.

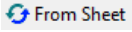
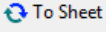
To select the members to display in a data worksheet:

1. In the Designer Panel, select the **Data** tab.
2. Click **Member Selection**.

3. In the **Member Selector**, check the members you want to display and clear the members you don't want to display.
4. Click **OK**.
5. Select **To Sheet**  to add your changes to the selected **Data** tab in the worksheet.

Work with Calculation Worksheets in Cube Designer

You can create new calculation worksheets in the Designer Panel.

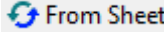
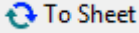
1. In the Designer Panel, select the **Calc** tab.
2. Select **From Sheet**  to populate the Designer Panel with the contents of the application workbook.
3. In the **Calculation Sheets** field, enter a name for the new calculation worksheet.
4. Press **Enter**.
5. Select **To Sheet** .

A new calculation worksheet is created in the application workbook.

Cube Designer calculation worksheets apply only to block storage cubes.

Work with MDX Worksheets in Cube Designer

You can create new MDX worksheets in the Designer Panel.



1. In the Designer Panel, select the **Calc** tab.
2. Select **From Sheet**  to populate the Designer Panel with the contents of the application workbook.
3. In the **MDX Insert Sheets** field, enter a name for the new MDX worksheet.
4. Press **Enter**.
5. Select **To Sheet** .

A new MDX worksheet is created in the application workbook.

See [Understand MDX Worksheets](#).

Create a Cube from a Local Application Workbook in Cube Designer

Using a sample local application workbook, you can create a cube from Cube Designer.

1. In Excel, on the Cube Designer ribbon, select **Local**  , and then select **Open Application Workbook**.
2. Select an application workbook, then select **Open**.
3. On the Cube Designer ribbon, select **Build Cube** .

4. On the Build Cube dialog box, verify that you want to use the selected options. Cube Designer detects data worksheets and calculation worksheets in the application workbook, and pre-selects those options for you, however you can deselect those options if you want to:
 - **Load Data Sheets Contained within Workbook** is pre-selected if data worksheets exist in the workbook. You can de-select this option if you do not want to load data.
 - **Run Calculation Sheets Contained within Workbook** is pre-selected if calculation worksheets exist in the workbook. You can de-select this option if you do not want to run the calculations.
5. Click **Run**.
6. After the asynchronous job completes a dialog box is displayed. Click **Yes** to launch Job Viewer and view the status of the Excel import, or click **No** if you don't want to launch Job Viewer.
See [Work with Jobs in Cube Designer](#).

Work with Text Lists Worksheets in Cube Designer

You can add text list definitions to application workbooks to work with text measures.

1. Open an application workbook.
2. On the Cube Designer ribbon, click **Cube Designer** to open the Designer Panel.
3. Click the **Settings** tab.
4. Click **From Sheet** to populate the Designer Panel with the contents of the application workbook.
5. In the **Text Lists** field, type the name for the new text list.
6. Press Enter.

The text list name is moved to the text box below the **Text Lists** field.

7. Click **To Sheet**.

A new text list definition section is added on the Cube.Textlists worksheet in the application workbook. If no Cube.Textlists sheet exists, one is created and the text list definition is added. Multiple text list definitions are supported, and will be added to the same worksheet.

After you add the text list, you must enter the text list information manually. This includes the associated members for the text list, the valid text items in the list and their related numeric values.

- [Understand the Cube.Textlists Worksheet](#)
- Working with Typed Measures
- Performing Database Operations on Text and Date Measures

Create a Cube from Tabular Data in Cube Designer

This workflow uses two sample tabular data Excel files to demonstrate the concepts of intrinsic and forced-designation headers. See [Transform Tabular Data to Cubes](#).



Catalog

1. In Excel, on the Cube Designer ribbon, click **Catalog**.
2. On the Essbase Files dialog box, under **Catalog**, select **Gallery**, then select a sample tabular data file:
 - Technical/Table Format/**Sample_Table.xlsx**: Intrinsic headers
 - Technical/Table Format/**Unstr_Hints.xlsx**: Forced-designation headers
3. Click **Open**.

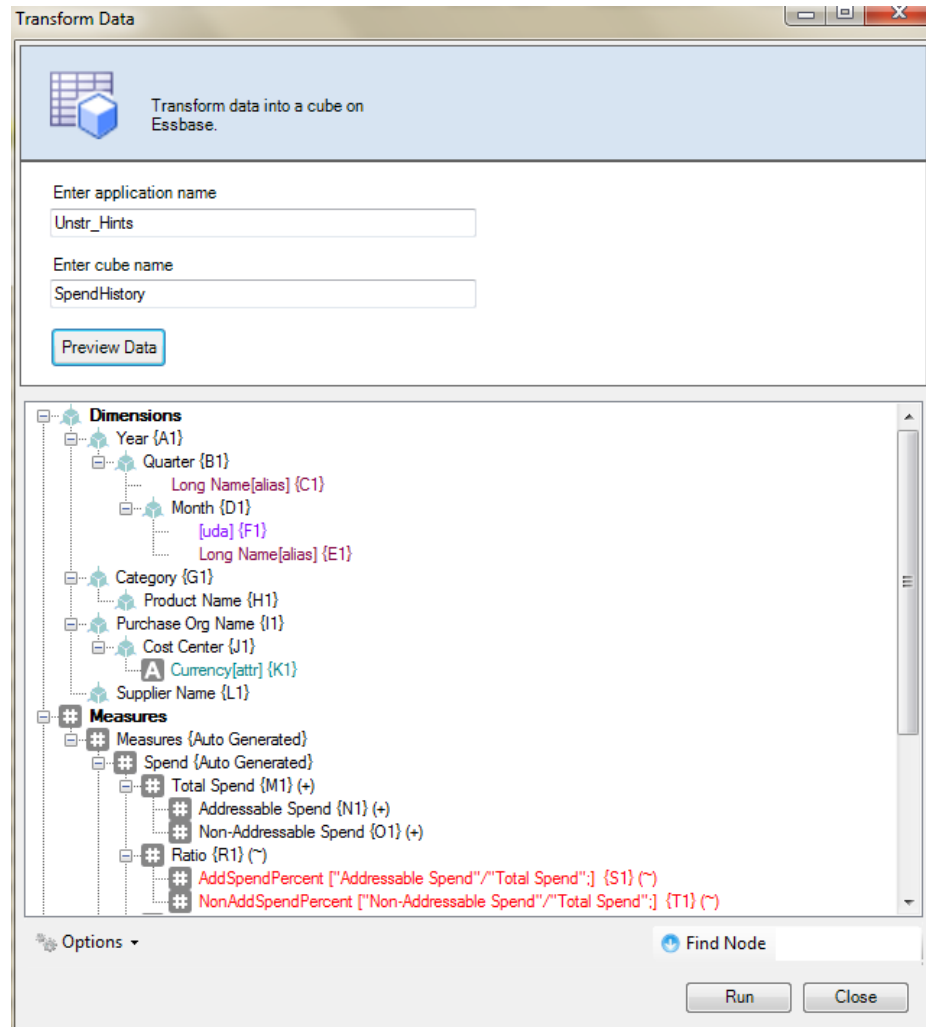




Transform
Data

4. On the Cube Designer ribbon, select **Transform Data**.
5. On the Transform Data dialog box, enter an application and cube name, if you want to change the default names that are prepopulated.

The application name is based on the source file name without the extension and the cube name is based on the worksheet name.


- **Sample_Table.xlsx**: Application name is **Sample_Table** and the cube name is **Sales**.
 - **Unstr_Hints.xlsx**: Application name is **Unstr_Hints** and the cube name is **SpendHistory**.
6. If you selected **Sample_Table.xlsx**, do not select **Preview Data**. Skip to step 8 to create the cube.
 7. If you selected **Unstr_Hints.xlsx**, press **Preview Data**. The workbook is sent to Essbase for analysis and the relationships are returned for viewing.
 - a. Using the **Tree View**, you can drag and drop columns to create dimension hierarchies, measure hierarchies, and skipped columns. You can also right click on a column name and designate the property of the column: Generation, Attribute, Alias or UDA. You can also select measures, and hierarchical or flat dimensions, in the **Options** menu, to set those options for the headers. If you select hierarchical, you get a hierarchy with the dimensions receiving the Excel header names. If you select the flat option, you get a flat display of generations that receive the Excel header names. This setting applies to the entire outline.



- b. To save changes to the Excel file, click **To Sheet** .
 - c. You can also make changes directly in the open Excel file and have those changes reflected in the grid view by clicking **From Sheet** .
 - d. If you do not want to save your changes, select **Options** and then select **Reset to Original Header**.
 - e. If you want to change the cube type and the type of dimensions to be created, before deploying, select **Options**, and then select **Cube Type**. Select **Hybrid BSO** (block storage option) or **ASO** (aggregate storage option).
8. When you are ready to create the cube, click **Run**.
 9. When prompted, save the application workbook to your private inventory directory.
 10. When asked if you want to create the cube, click **Yes**.
 11. (Optional) When asked if you want to see the cube job status, click **Yes**.

Status	Job Id	Job Type	Data File	Script	Server	Application	Cube	Start Time	Elapsed Time
Success	10	Import Excel	None	ct_Table.xlsx	http://den02azp:9000	ct_Table	ct_Sales	11/10/2016 5:50:33 PM	00:00:20
Success	9	Export Excel	None	DBX_dw_sample_dw_basic.xlsx	http://den02azp:9000	dw_sample	dw_basic	11/10/2016 2:08:16 PM	00:00:03
Success	8	Export Excel	None	DBX_Sample_Sandbox.xlsx	http://den02azp:9000	Sample	Sandbox	11/10/2016 2:06:20 PM	00:00:03
Success	7	Export Excel	None	DBX_dw_sample_dw_basic.xlsx	http://den02azp:9000	dw_sample	dw_basic	11/10/2016 2:05:52 PM	00:00:03
Success	6	Export Excel	None	DBX_dw_sample_dw_basic.xlsx	http://den02azp:9000	dw_sample	dw_basic	11/10/2016 11:47:38 AM	00:00:03

The newly created application and cube are listed on the Applications home page in the Essbase web interface and are available in Cube Designer. Now that the cube has been created from the tabular data, you can export the cube to an application workbook.

12. On the Cube Designer ribbon, select **Local**  , then select **Export Cube to Application Workbook**.
13. On the Export Cube to Application Workbook dialog box, select the application and cube, and then select **Run**

To create a cube using the web interface, see [Create and Update a Cube from Tabular Data](#).


Update Cubes Incrementally in Cube Designer

Updating a cube is how you load dimensions and members to a cube outline using a data source and a rule file.

You can also use Essbase to add dimensions and members manually (see [Creating and Updating Cubes from Tabular Data](#)).

In an existing cube, you can incrementally update a dimension, or add a new one.

You cannot use Cube Designer to delete dimensions or rename members in an existing cube.

1. In Excel, on the Cube Designer ribbon, select **Build Cube** .
2. Choose an **Update Cube** option from the **Build Option** menu.

When an outline was changed by a dimension build, the database may be restructured. Each of these options specifies how data values are handled during restructures:

- a. **Update Cube - Retain All Data**
All data values are preserved.

b. Update Cube - Retain Input Data

All blocks (both upper-and lower-level) that contain loaded data are preserved.
This option applies only to block storage cubes.

c. Update Cube - Retain Leaf Data

Only leaf (level 0) values are preserved. If all data required for calculation resides in leaf members, then you should select this option. If selected, then all upper-level blocks are deleted before the cube is restructured. Therefore, the disk space required for restructuring is reduced, and calculation time is improved. When the cube is recalculated, the upper-level blocks are re-created.

d. Update Cube - Remove All Data

All data values are cleared.

This option applies only to block storage cubes.

- Dimension build definitions are contained within the application workbook and automatically generate the necessary rules files. You do not select a rule file when building dimensions in Cube Designer.
- When making changes to user-defined attributes (UDAs) while updating a cube incrementally using Cube Designer and an application workbook, you must specify all the UDAs in the dimension sheet, both new ones you are adding and existing UDAs in the outline. If you specify some UDAs (such as those you are adding), but not all of them, those that are not specified are deleted.
- When incrementally adding a dimension to an existing cube using an application workbook, the data is automatically mapped to the new top member. There is not a way to choose a stored member to which to map the existing data. If the new dimension has a top member that is dynamic calc, the data is lost because dynamic members can't store data.

When using an application workbook to add a new dimension in which you want the top member to be dynamic calc, follow these steps:

1. Add the new dimension with the top member as stored.
2. Run a calc script to copy the data from the new top member into another stored member in that dimension.
3. Change the top member to dynamic calc.


Create and Validate Member Formulas in Cube Designer

In the Cube Designer Formula Editor, you can write formulas for specific outline members in block storage cubes. You can construct member formulas from operators, functions, dimension names, member names, substitution variables, and numeric constants, and you can validate them to check for correct syntax.

- The Cube Designer Formula Editor applies only to block storage cubes.
- Validation works against existing cubes in Essbase. It does not detect application workbook changes that have not been applied to the cube.
- Member selection works with existing cubes only.

Formula Editor provides a formula editing pane in which you can enter a formula. You can use the Tab and arrow keys to move focus within Formula Editor. You can also

use a point-and-click approach to select and insert formula components into the formula editing pane. A member selection tree helps you place the correct member names into the formula.

1. Open the application workbook for the cube that you want to modify.
2. If a dimension worksheet has been defined with the Formula property, select the cell in the Formula column for the member you wish to create a formula.
3. On the Cube Designer ribbon, click **Formula Editor** .
4. Enter your login credentials for Essbase, if prompted to do so.
5. In the Formula Editor, create the formula.
 - Use the keyboard to enter formula text. Enclose in quotation marks any member names containing blanks or special characters.
 - Select a cell containing a member name or alias from any dimension worksheet. Place the cursor in the appropriate location of the editor and right-click to paste that name surrounded by quotes into the editor.
 - Double-click on a member in the member selection tree to have that member pasted into the editor.
 - Double-click on a function to have that function syntax pasted into the editor.
6. Click **Validate** to check formula syntax.


If the validation fails, edit the formula and try again. Be sure to check the error message for guidance.

See:

- Developing Formulas for Block Storage Databases
- Understanding Formula Syntax
- Reviewing Examples of Formulas

Load Data in Cube Designer

At times, you may need to clear and reload data during cube development. The data and rules files used in the data load process must be stored in Essbase. If a data worksheet is included in the application workbook, then the data files and rule files get automatically generated during the cube build process. Individual files can also be uploaded. See [Upload Files to a Cube](#).

1. In Excel, on the Cube Designer ribbon, select **Load Data** .
2. On the Load Data dialog box, select the application and cube in which you want to load data.
3. Under **Select a Job Type**, select an option:
 - **Load Data**: to load data to the cube.
 - **Clear all Data**: to clear all data from the cube.
4. Select the data file and load rule file that you want to use.
5. Select whether to **Abort on Error**.

If you select **Abort on Error**, the data load is stopped when an error is encountered.


6. Click **Run** to start the data load.
7. When the asynchronous job completes a dialog is displayed. Click **Yes** to launch Job Viewer and view the status of the data load, or click **No** if you do not want to start Job Viewer.
8. (Optional) View the status in Job Viewer.

See Understanding Data Loading and Dimension Building.

Calculate Data in Cube Designer

Calculation scripts specify how cubes are calculated and, therefore, override outline-defined cube consolidations. For example, you can calculate cube subsets or copy data values between members. See Developing Calculation Scripts for Block Storage Databases.

During cube development, it is common to recalculate a cube many times when validating the data and formulas. The calculation script files used in the calculation process must be stored in Essbase. If a Calc worksheet is included in the application workbook, then the calculation script files are automatically generated during the cube build process. Individual calculation script files can also be uploaded to Essbase. See [Upload Files to a Cube](#).

1. In Excel, on the Cube Designer ribbon, select **Calculate**  Calculate.
2. On the Calculate Data dialog box, select an application and a cube, and select the calculation script you want to use.
3. Click **Run** to start the calculation.
4. When the asynchronous job completes a dialog box is displayed. Click **Yes** to start Job Viewer and view the status of the calculation, or click **No** if you do not want to start Job Viewer.
5. (Optional) View the status in Job Viewer.
See [Work with Jobs in Cube Designer](#).

Work with Jobs in Cube Designer

Use the Cube Designer Job Viewer to view, monitor and troubleshoot jobs that you run from your particular client. Jobs are operations such as data loads, dimension builds, and calculations.

A record of all Essbase jobs is maintained in the Essbase instance. Each job has a unique ID number.

The jobs listed in the Job Viewer are for one specific user. If a different user logs into the client, then only jobs for that user are displayed.

View Jobs in the Cube Designer Job Viewer

You can view jobs for the specific user that is logged into the client in the Cube Designer Job Viewer.

In Excel, on the Cube Designer ribbon, click **View Jobs** .

The Job Viewer dialog box opens, showing a list of jobs that have been run from that particular client.

Monitor Cube Designer Jobs

The Cube Designer ribbon shows when a job is in progress. After the job finishes, you can view the status of the job in the Cube Designer Job Viewer.

- While a job is running, the **Job Viewer** icon on the Cube Designer ribbon displays

an hourglass .

- When the job finishes running a Job Viewer status dialog box displays, indicating the status of the job.

If you close Excel while the job is running, the job continues to run, but you will not see a status dialog when it finishes. The job is a server process, so it runs regardless of whether Excel is open or not.

Troubleshoot Jobs in the Cube Designer Job Viewer

If a job fails, you can view and troubleshoot errors.

1. In Job Viewer dialog box, select a job and click **Details** to see the job details.
2. In the Job Details dialog box, select a file from the **Server Error Files** drop-down menu and click **Open** to view and troubleshoot errors.

Clear and Archive Cube Designer Jobs

Clear the Job Viewer or archive job viewer logs periodically to improve performance.

- Press **Clear All** to remove all jobs from the Job Viewer dialog box.
- To selectively remove individual jobs, select one or more jobs and press the Delete key.
 - Use the Shift key to select multiple contiguous jobs.
 - Use the Ctrl key to select multiple non-contiguous jobs.
- To archive the job viewer logs, copy and rename the log file and then delete the original.

Job viewer logs are located in `C:\Users\username\AppData\Roaming\Oracle\SmartView\DBX\Jobs`.

There is a separate log for each user on the client machine.

Removing jobs from the Job Viewer dialog box or archiving job viewer logs only affects the client. You can still view all jobs in the web interface.

View Dimension Hierarchies in Cube Designer

You can view dimension hierarchies in the Cube Designer Dimension Hierarchy viewer. To learn more about hierarchies, see [Outline Hierarchies](#).

1. Open the application workbook containing the hierarchy that you want to view.
2. Select the dimension worksheet for the hierarchy that you want to view.

3. On the Cube Designer ribbon, select **Hierarchy Viewer**



When you view a hierarchy in Cube Designer, you can perform some actions on the hierarchy. These include:

- To search for a member in the hierarchy, enter a member name in the **Find Next** text box, and click **Find Next** .
- To find a member of the dimension in the application workbook dimension worksheet, either double-click a member in the hierarchy or right click on a member in the hierarchy and select **Go To** .

The corresponding member in the application workbook is highlighted.

- To rename a member:
 1. Right-click a member in the hierarchy and select **Rename** .
 2. Enter the new member name.
 3. Press **Enter**.

The corresponding member is renamed wherever found within the Parent and Child columns of the dimension worksheet.

- To set storage for all parents (except members containing formulas or defined as label only) to dynamic calc or to stored:

1. Select the member in the hierarchy and click **Edit parents** .
2. On the drop-down menu, select **Set storage to dynamic calc** or **Set storage to stored**.

- To expand or collapse a hierarchy:
 1. Right-click a member in the hierarchy.
 2. Select **Expand All** or **Collapse All**.
- To show or hide aliases, storage, or operators:

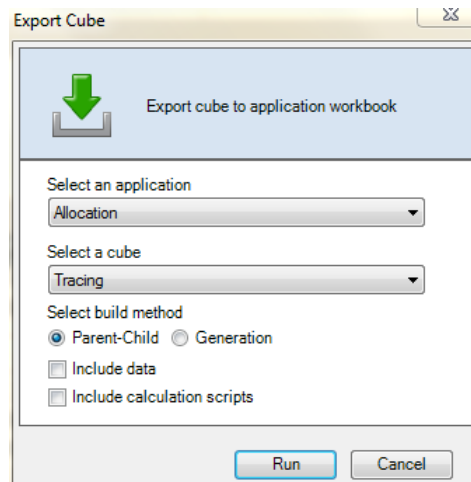
1. Click **Show** .
2. Click **Alias**, **Storage**, or **Operator**, to show or hide those items.

Export Cubes to Application Workbooks in Cube Designer

In Cube Designer, you can export any cube that exists in Essbase.

1. Select the build method, either parent-child or generation format.
2. In Excel, on the Cube Designer ribbon, select **Local** , then select **Export cube to application workbook**.
3. On the Export Cube dialog box, select the application and cube that you want to export.

- Select **Include Data** if you want input level data included in the application workbook.
 - In block storage cubes, if the size of the data is 400 MB or less, data is exported to the application workbook, on the Data worksheet. If the data size exceeds 400 MB, data is exported to a flat file named *Cubename.txt*, which is included in a file named *Cubename.zip*. The *.zip* file is created in the specified export directory if the export process is successful.
 - In aggregate storage cubes, regardless of the size of the data, it is always exported to a flat file named *Cubename.txt*, which is included in a file named *Cubename.zip*. The *.zip* file is created in the specified export directory if the export process is successful.
 - Select **Include Calculation Scripts** if you want calculation scripts in your block storage cube included in the application workbook.
- Aggregate storage cubes do not have calculation scripts.



4. Click **Run**.
5. When the export is completed, click **OK**.

The application workbook is saved to the local folder location: `C:\Users\username\AppData\Roaming\Oracle\smartview\DBX`. Because it is saved to the local folder

location, you can open it using the **Local**  icon on the Cube Designer ribbon.

The exported application workbook can be imported to Essbase. See these topics:

- [Create a Cube from an Application Workbook](#)
- [Create a Cube from a Local Application Workbook in Cube Designer](#)

Delete Applications and Cubes in Cube Designer

In Cube Designer, you can delete any application or cube that exists in Essbase. Deleting an application or cube cannot be undone.

1. In Excel, on the Cube Designer ribbon, select **Admin tasks**  Admin tasks ▾ .


2. From the menu, select **Delete Application** or **Delete Cube**.
3. From the Delete Application or Delete Cube dialog box, select the application or cube you want to delete.

Unlock Objects in Cube Designer

Essbase uses a checkout facility for cube objects (such as calculation scripts and rules files). Objects are locked automatically when they are in use and the locks are deleted when they are no longer in use.


You can view and unlock objects, according to your security role. Users with the Service Admin role can unlock any object. Users without the Service Admin role can unlock only those objects that they locked.

To unlock an object in cube designer:

1. In Excel, on the cube designer ribbon, select **Admin tasks**  Admin Tasks ▾
2. Select **Unlock Essbase objects**.
3. Enter your login credentials if prompted to do so.
4. Under **Select an application**, select the application containing the object you want to unlock.
5. Under **Select a locked object**, select the object you want to unlock.
6. Click **Unlock**.

View Logs in Cube Designer

In cube designer, you can view the platform log or an application log.

1. In Excel, on the cube designer ribbon, select **Admin tasks**  Admin tasks ▾
2. From the menu, select **View Logs**.
3. Select a log to view:
 - Select **View Platform Log** to view the log for the platform service.
 - Select **View Application Log** to view the log for an individual application.

15

Track Changes to Data

Use an audit trail to track changes to cube data, including changes to Linked Reporting Objects (LROs): adding notes, attaching files, and referencing URLs. Export your log to an Excel spreadsheet, and perform ad hoc queries.

To view data audit trail records, you must be at least a power user with Database Update permission on the application. You can only view those records where your user name matches the user name registered in the audit records. To delete data audit trail records, you must be at least a power user with Application Manager permission on the application. See [Understand Your Access Permissions in Essbase](#).

- [Turn on Data Audit Trail and View the Data Audit Trail](#)
- [Link a Report Object to a Cell](#)
- [Export Logs to a Sheet](#)
- [Refresh the Audit Log](#)
- [View and Manage Audit Trail Data in the Essbase Web Interface](#)

Limitations of Audit Trail

- Supported only on block storage cubes
- Not supported with text measures

Turn on Data Audit Trail and View the Data Audit Trail

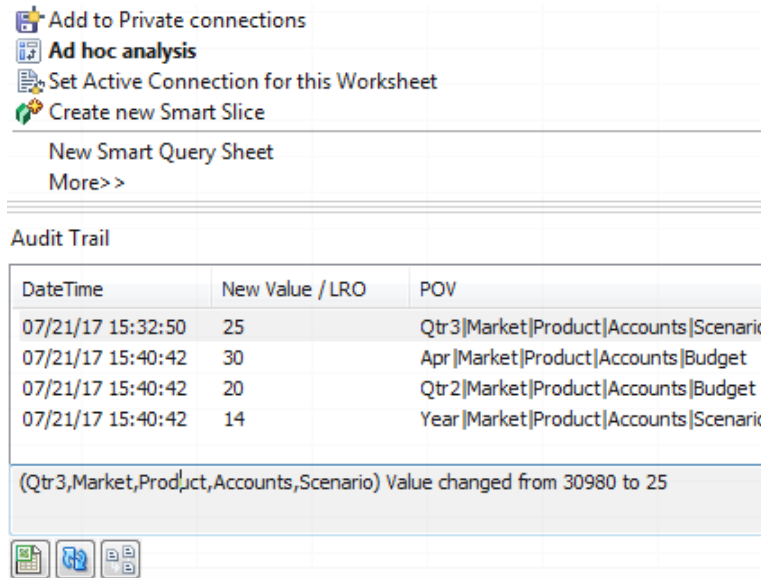
You turn on data audit trail by adding AUDITTRAIL DATA as an application level configuration setting.

See [Set Application-Level Configuration Properties](#).


1. To turn on Data Audit Trail, add the following to the application configuration parameters: AUDITTRAIL DATA.
2. Perform ad hoc analysis through Smart View, make data changes through Smart View and click on **Submit** - this results in an audit record being stored.

When doing ad hoc analysis, there are many ways of getting a particular Point of View (POV) onto the grid. One of them is by using the POV toolbar, which allows you to zero in on certain members in one or more dimensions. See these topics in *Working with Oracle Smart View for Office*

- [Selecting Members from the POV Toolbar](#)
 - [Displaying the POV Toolbar](#)
 - [Selecting Members Using the Cell-Based POV](#)
3. With Data Audit Trail enabled, you can view the audit trail in the connection Panel in Smart View. Under the connection information, click on the menu of operations under **More** and you will see a menu option titled **Audit Trail**. Click on **Audit Trail** to view the data audit trail records for a cube.



- The audit trail record shows the date and time of the change in the first column, the new value or the linked reporting object in the second column and the POV in the third column. The time corresponds to your time zone. Click on an item in the audit trail and you will see a description of the change at the bottom of the pane.
- You can display a sheet with the new POV and refreshed data value by clicking

Ad hoc  below the **Audit Trail** pane. When you click on subsequent audit records and click this icon, you see a different sheet with the POV for that audit record and refreshed data for that POV. This way, you can do further analysis on targeted data.


Link a Report Object to a Cell

You can link a reporting object to a cell. When you do, this change displays in the data audit trail. You can add a note to a cell, attach a file, or reference a URL. When you make these changes, the cells are highlighted in your cube. See these topics in the *Working with Oracle Smart View for Office* on how to link reporting objects to cells:

- Linked Reporting Objects
- Attaching a Linked Reporting Object to a Data Cell
- Launching a Linked Reporting Object from a Data Cell

Export Logs to a Sheet

You can easily export your logs to a new Excel sheet just by clicking an icon.

Export your log onto a new sheet using **Export** . Click this icon to export the logs with all the details for each entry onto a new sheet that looks like this:

	A	B	C	D	E	F	G	H	I	J
1	User	DateTime	Cell Note	New Value	Old Value	Operation	POV			
2	weblogic	07/21/17 15:32:50		25	30980	INPUT	Qtr3	Market	Product	Accounts Scenario
3	weblogic	07/21/17 15:40:42		30	9777.5	INPUT	Apr	Market	Product	Accounts Budget
4	weblogic	07/21/17 15:40:42		20	29903.1	INPUT	Qtr2	Market	Product	Accounts Budget
5	weblogic	07/21/17 15:40:42		14	133980	INPUT	Year	Market	Product	Accounts Scenario

Once exported, you can resort columns or remove them to show the information you want to analyze.

Refresh the Audit Log

You can refresh the audit log to see your latest changes at any time.

When you make more changes to your data, you can refresh the log view any time.

Click **Refresh** .

	A	B	C	D	E	F	G	H	I	J
1	User	DateTime	Cell No	New Value	Old Value	Operation	POV			
2	weblogic	07/21/17 15:32:50		25	30980	INPUT	Qtr3	Market	Product	Accounts Scenario
3	weblogic	07/21/17 15:40:42		30	9777.5	INPUT	Apr	Market	Product	Accounts Budget
4	weblogic	07/21/17 15:40:42		20	29903.1	INPUT	Qtr2	Market	Product	Accounts Budget
5	weblogic	07/21/17 15:40:42		14	133980	INPUT	Year	Market	Product	Accounts Scenario
6	weblogic	07/23/17 16:20:13		45	-403	INPUT	Jul	East	Visual	Accounts Variance
7	weblogic	07/23/17 16:20:13		55	-271	INPUT	Sep	South	Visual	Accounts Variance
8	weblogic	07/23/17 16:20:13		65	-1840	INPUT	Qtr4	South	Visual	Accounts Variance

View and Manage Audit Trail Data in the Essbase Web Interface

You can view audit trail data in the Essbase web interface. You can also export the data to an Excel sheet (in .csv format), purge the data before a specific date, or purge all of the audit trail data.

To view and manage audit trail data:

1. On the Applications page, expand the application.
2. Click the **Actions** menu to the right of the cube name and select **Inspect**.
3. On the **Audit Trail** tab, you can:
 - View audit trail data.
 - Export the data to a CSV file.
 - Purge the audit trail data until a specific date.
 - Purge all of the audit trail data.

To purge data audit trail records, you must be a power user with Application Manager permission on the application.

16

Link Cubes Using Partitions and XREF/XWRITE

You can use either partitions or XREF/XWRITE to analyze data across cubes.

You may have more than one cube that you use for business analysis. To share data across multiple cubes, you can connect the cubes by implementing partitions, XREF/XWRITE, or both. Two cubes connected by a partition can be thought of as a source and target pair. When using XREF/XWRITE, it is easiest to think of the local cube and the remote cube.

When partitioning or using XREF/XWRITE functions between cubes on the same Essbase instance, no reference to the host instance or login credentials are required. However, if the cubes you wish to connect are on separate Essbase instances, you will first need to create a reusable connection to link the two instances.

To use partitions or XREF/XWRITE, users must be provisioned on the remote cube as well as the local cube.

The source cube and target cube of a partition must be on the same Essbase version.

- [Define a Reusable Connection for Partitions or XREF/XWRITE](#)
- [Understand Transparent and Replicated Partitions](#)
- [Create a Transparent Partition](#)
- [Create a Replicated Partition](#)
- [Refresh a Replicated Partition](#)
- [Understand XREF/XWRITE](#)
- [Create a Location Alias Based on a Defined Connection](#)

Define a Reusable Connection for Partitions or XREF/XWRITE

This topic shows you how to create a reusable connection between two Essbase instances. Using the connection, you can then create partitions or use XREF/XWRITE functions.

Create connections globally for use with all applications on the system, or at the application level for use within the context of an application. Global connections require system administrator role, whereas application connections require, at minimum, Application Manager role.

1. In the Essbase web interface, click **Sources**, and click **Create Connection > Essbase** to create a global Essbase connection. Alternatively, use the **Actions** menu on the target or local application and select **Inspect**, followed by **Sources**, **Create Connection**, and **Essbase**.

2. In the **Name** field, enter a name for the saved connection; for example `myhost01_conn`.
3. Select the **Use URL** checkbox, and enter the discovery URL of the remote Essbase instance. The discovery URL is available from your system administrator, and ends in `/agent`.
4. Enter a user name, password, and a description. The user defined in the connection must be provisioned for the source application you intend to access on the remote instance. If you have used a global connection, the user will need to be a system administrator or be provisioned for all applications you intend to access using the connection.
5. Click **Test** to verify that the connection is valid.
6. If it is valid, click **Create** to save the connection.

Now you have a remote Essbase connection defined in the service. You can use this connection to define partitions between the two instances, or combine it with a location alias to enable XREF/XWRITE functionality between the two instances. See also: [Create a Connection and Datasource to Access Another Cube](#)

Understand Transparent and Replicated Partitions

A partition is a region of a cube that is shared with another cube. You can create a transparent or replicated partition between a target and a source cube, to share congruent cube regions between them. In the Essbase web interface, you create partition definitions in the target cube.

A **transparent** partition target region is virtual; it pulls data on-demand from a source cube region containing stored data. The source cube can be in the same or another application, or on another Essbase instance.

A **replicated** partition target region is a physical copy of stored data from the source cube region. Data stored in a replicated partition target must be synchronized when data changes in the source cube. Using the replicated partition, some users access the data in the target, while others access it in the source.

Changes made to the data in a replicated partition flow from the source to the target. If users are permitted to change the data in the target partition region, it is overwritten when the replicated partition is refreshed.

If all cubes involved in a transparent or replicated partition are hosted on the same instance of Essbase, no login credentials are needed as part of the partition setup. However, the user creating the partition must be provisioned on the target application and also the source application. Business users querying the target cube must also be provisioned on both cubes, typically with Read access.

Create a Transparent Partition

This topic shows you how to create a transparent partition. Transparent partitions allow access to data from the data source as though it were stored in the data target. The data source can be in another cube, or on another Essbase instance.

If your source cube is on a different Essbase instance, you must first define an Essbase connection as described in [Define a Reusable Connection for Partitions or XREF/XWRITE](#).

1. In the Essbase web interface, on the **Applications** page, expand the target application. In the row for the target cube, click the **Actions** menu, and click **Inspect**.
2. Select the **Partitions** tab.
3. Click **Create >Transparent**.
4. On the **Connection** tab, in **Source Info**, if the source cube is on a different Essbase instance, select the name of the saved connection that you created. If the source cube is on the same Essbase instance, leave the **Connection Name** field empty. If you have not created any connections, you will not see a **Connection Name** field.
5. Provide the source **Application** and **Database** name, and an optional **Description**.
6. If the source cube is on a different Essbase instance, in the **Target Info**, type your **User name** and **password**.
7. You need to define at least one area. Go to the **Areas** tab.
8. Click **Add Area** and provide at least one source and target area definition. For example, add a source area of `@DESCENDANTS(valid upper-level member specification)`, and add the same matching target area. If the same member doesn't exist in both cubes, create an area mapping as described below.
9. Click **Cell Count** to see how many cells are in the defined partition area and to ensure that the counts are matching.
10. Optionally, you can map member names between the target and source cubes within a specific area, using the **Areas** tab, or for multiple areas, using the **Mappings** tab.
11. Click **Validate**.
12. If the validation succeeded, click **Save and Close**.

Create a Replicated Partition

This topic shows you how to create a replicated partition, which duplicates an area of a source cube into the target cube. The data source can be in another cube, or on another Essbase instance.

If your source cube is on a different Essbase instance, you must first define an Essbase connection as described in [Define a Reusable Connection for Partitions or XREF/XWRITE](#).

1. In the Essbase web interface, on the **Applications** page, expand the target application. In the row for the target cube, click the **Actions** menu, and click **Inspect**.
2. Select the **Partitions** tab.
3. Click **Create >Replicated**.
4. On the **Connection** tab, in **Source Info**, if the source cube is on a different Essbase instance, select the name of the saved connection that you created. If the source cube is on the same Essbase instance, leave the **Connection Name** field empty. If you have not created any connections, you will not see a **Connection Name** field.

5. Provide the source **Application** and **Database** name, and an optional **Description**.
6. If the source cube is on a different Essbase instance, in the **Target Info**, type a provisioned **User name** and **password**.
7. You need to define at least one area. Go to the **Areas** tab.
8. Click **Add Area** and provide at least one source and target area definition. For example, add a source area of `@DESCENDANTS(valid upper-level member specification)`, and add the same matching target area. If the same member doesn't exist in both cubes, create an area mapping as described below.
9. Click **Cell count** to see how many cells are in the defined partition area and to ensure that the counts are matching.
10. Optionally, you can map member names between the target and source cubes within a specific area, using the **Areas** tab, or for multiple areas, using the **Mappings** tab.
11. Click **Validate**.
12. If the validation succeeded, click **Save and Close**.

Refresh a Replicated Partition

If you have at least Database Manager permission on a replicated partition target application, you can replicate the data from the source.

1. In the Essbase web interface, on the **Applications** page, expand the target application containing the replicated partition definition.
2. In the row for the target cube, click the **Actions** menu, and click **Inspect**.
3. Select the **Partitions** tab.
4. From the **Actions** menu on the replicated partition, select **Replicate Data from Source**.
5. Select **Update change cells only** to update the target only with source data that has been updated since the last update, or select **Update all cells** to update the target with all source data.

Understand XREF/XWRITE

XREF is a calculation function you use to reference data in another cube, and XWRITE is a calculation function you use to write back data to another cube.

XREF and XWRITE are easiest to understand from the context of the cube containing the XREF or XWRITE formula, called the local cube. The second cube is the remote cube.

To implement XREF, you define a formula in the local cube that pulls values from a remote cube. The member containing the XREF formula can either be stored or dynamically calculated.

To implement XWRITE, you define a formula in the local cube that pushes (writes) values into a remote cube. The remote cube data intersection must be stored, since XWRITE writes values into the remote cube.

When the local and remote cube are on the same Essbase instance, no connection information is needed to implement XREF or XWRITE. However, users of the local cube must also be provisioned on the remote cube. To implement XREF or XWRITE for cubes on the same instance, the application and database name for the source cube are required in the function syntax:

```
@XREF(appName, dbName [, mbrList])
@XWRITE (expression, appName, dbName [, mbrList])
```

If the local and remote cubes are on different Essbase instances, a location alias containing connection information must be defined:


```
@XREF (locationAlias [, mbrList])
@XWRITE (expression, locationAlias [, mbrList])
```

- @XREF
- @XWRITE
- [Create a Location Alias Based on a Defined Connection](#)

Create a Location Alias Based on a Defined Connection

This topic shows you how to create a location alias, which you can use when your XREF/XWRITE formulas need to reference data from a cube on a remote cloud instance. You do not provide a user name and password when you create a location alias. You use a saved connection.

This topic assumes you have created a connection as described in [Define a Reusable Connection for Partitions or XREF/XWRITE](#).

1. In the Essbase web interface, on the **Applications** page, expand the target application. In the row for the local cube, click the **Actions** menu, and click **Inspect**.
2. Click the **Location Aliases** tab.
3. Click 
4. In the **Location alias name** field, enter a name.
5. In the **Essbase connection** field, select a saved connection to the Essbase instance hosting the remote cube.
6. Select the remote **application** and **database** and click **Save**.

Now you have created a location alias. To use it for read operations from a remote cube to the target, use the @XREF function in a member formula or calculation script on the local cube. To use it to write from the local to the remote cube, use @XWRITE on the local cube.

```
@XREF (locationAlias [, mbrList])
@XWRITE (expression, locationAlias [, mbrList])
```

Configure Oracle Essbase

Oracle Essbase is preconfigured with properties that you may never need to modify.

If necessary, you can add or modify configuration properties at the Essbase application level, and you can add or modify Provider Services properties at the Essbase server level.

- [Set Application-Level Configuration Properties](#)
- [Set Provider Services Configuration Properties](#)

Set Application-Level Configuration Properties

If you have the Service Administrator role, or the Power User role for applications that you created, you can customize applications using application-level configuration properties. Application-level configuration properties apply to all cubes in the application.

One way to specify configuration properties of an application is to do it prior to building the application and cube, using the application workbook. To see an example, go to Files in the Essbase web interface, and download the application workbook `Sample_Basic.xlsx`. It is located in the gallery, in the Demo Samples section (under Block Storage). In this application workbook, go to the `Cube.Settings` worksheet. Under Application Configuration, you can see that the `DATACACHE SIZE` property is set to 3M, and the `INDEXCACHE SIZE` property is set to 1M.

The following steps tell you how to configure an application that is already deployed, by adding properties and their corresponding values in the Essbase web interface.

1. On the Applications page, select the application you want to configure.
2. From the **Actions** menu to the right of the application, click **Inspect**, then click **Configuration**.
3. To add a property, click **+**. Select a property from the list. When done adding properties, close the list window.
4. To change a property's value, double-click a property row and edit its value.
5. When you're finished making changes, click **Apply**.

The configuration changes will take effect next time the application restarts.

For syntax and information about each of the application configuration properties you can use, see Config Settings List.

Oracle does not recommend that you modify `essbase.cfg` on the Oracle Essbase file system. This configuration is automatically set.

Set Provider Services Configuration Properties

If you have the Service Administrator role, you can customize network-related settings for Oracle Essbase using the Provider Services configuration properties.

To set the values for Provider Services configuration properties,

1. Log in to the Essbase web interface as a Service Administrator.
2. Click **Console**.
3. In the Console, click **Configuration**.
4. On the Provider Services tab, click **Add** to add a new property and set its value. If the property you want to configure is already listed, double-click the **Value** field to edit the value.
5. When you are finished editing properties, click **Save**.

18

Essbase Command-Line Interface (CLI)


The command-line interface is a nongraphical interface in which you enter shell commands to perform administrative actions on Essbase.

- [Download and Use the Command-Line Interface](#)
- [CLI Command Reference](#)

Download and Use the Command-Line Interface

1. If it is not already installed, download and install Java SE Development Kit 8 from Oracle Technology Network.
2. Set the JAVA_HOME environment variable on your system to point to the JDK installation folder. If the installation path has any spaces, enclose the path in quotation marks.

Variable name:	<input type="text" value="JAVA_HOME"/>
Variable value:	<input type="text" value='"C:\Program Files\Java\jdk1.8.0_171"'/>

3. In the Essbase web interface, click **Console**.
 4. In the Console, go to **Desktop Tools** and expand **Command Line Tools**.
 5. Click **Download**

- next to the utility labeled **Command-line Tool**.
6. Download `cli.zip` to a local drive. For best results, choose a path that has no spaces; for example, `C:\Oracle`.
 7. Uncompress `cli.zip`, and see the extracted files under the `cli` folder.
 8. To issue commands interactively,

- a. Navigate to the CLI folder containing the shell script, `esscs.bat` or `esscs.sh`.

- b. Set the proxy and launch the CLI:

For Windows:

```
set HTTPS_PROXY=www-proxy.example.com:80
esscs.bat login -u MyAdmin -p mypass7YG -url https://192.0.2.1/
essbase
```

For Linux:

```
export HTTPS_PROXY=www-proxy.example.com:80
esscs.sh login -u MyAdmin -p mypass7YG -url https://192.0.2.1/
essbase
```

For more examples and details, see the [login](#) command topic.

If the CLI was installed correctly, a list of supported commands is displayed.

9. To execute multiple CLI commands, add them to any shell script and execute it.

In any script you run that contains CLI commands, Oracle recommends you include the following directive before the CLI login statement:

For Windows:

```
set ESSCLI_ID=%USERNAME%_%random%
```

For Linux:

```
export ESSCLI_ID=`whoami`_$$PPID
```

This helps store session information and prevent execution errors when multiple scripts are run concurrently.

CLI Command Reference

The following commands are available in the command-line interface. Arguments to commands can be issued in any order.

- [calc](#)
- [clear](#)
- [createlocalconnection](#)
- [dataload](#)
- [deletefile](#)
- [deploy](#)
- [dimbuild](#)
- [download](#)
- [help](#)
- [lcmexport](#)
- [lcmimport](#)
- [listapp](#)
- [listdb](#)
- [listfiles](#)
- [listfilters](#)
- [listlocks](#)

- [listvariables](#)
- [login, logout](#)
- [setpassword](#)
- [start](#)
- [stop](#)
- [unsetpassword](#)
- [upload](#)
- [version](#)

To display help for all commands, enter `esscs -h`. To display help for a specific command, enter `esscs command -h`.

To turn on verbose output for any command, meaning that extended information (if available) is displayed, enter `esscs command -v command arguments`.

Login/Logout: CLI Authentication

Before you can issue CLI commands to Essbase, you must log in. If a secure connection is required, then the URL must begin with `https`.

You can authenticate in the following ways using CLI:

- Use `setpassword` once to have the password stored for your client/user combination. In subsequent sessions, you can use the `login` command without being prompted to enter a password.
- Use the `-user` and `-password` options with the `login` command (Caution: the password appears in the shell window as cleartext).
- Use only the `-user` option with the `login` command. You are prompted to enter the password, which is hidden.

If you're a [federated](#) SSO user in Oracle Identity Cloud Service, logging in using MaxL or CLI is not supported. Federated SSO login requires a browser window. Create a native Identity Cloud Service user, and use that instead to log in using MaxL or CLI.

Syntax (login)

```
login [-verbose] -essbaseurl https://instance-name.example.com/essbase -
user username [-password password]
```

Option	Abbreviation	Description
<code>-verbose</code>	<code>-v</code>	Show extended descriptions
<code>-essbaseurl</code>	<code>-url</code>	Address of an instance of Essbase
<code>-user</code>	<code>-u</code>	User name
<code>-password</code>	<code>-p</code>	Optional. Password for user. Alternatively, set the password using setpassword . If issuing the the login command from a script, and the password contains special characters, enclose it in double quotation marks (for example, "aNb3^5%9\$!").

Example 1 (login)

```
esscs login -url https://myEssbase-test-myDomain.analytics.us2.example.com/  
essbase -u smith
```

Example 2 (login)

In the following example, the user logging in, `admin1@example.com` is an Identity Cloud Service administrator who was set as the initial Essbase administrator during Essbase stack deployment on Oracle Cloud Infrastructure. As the password is not entered in this example, the administrator will be prompted to provide it next. The URL is the **essbase_url** from the job outputs resulting from the stack deployment.

```
esscs login -u admin1@example.com -url https://192.0.2.1/essbase
```

Syntax (logout)

```
logout
```

Example (logout)

```
esscs logout
```

Calc: Run a Calculation Script

This CLI command executes a calculation script on the cube. To run it, you need at least Database Update permission, as well as provisioned access to the calculation script.

To run calculation scripts, you must first upload the scripts, as `.csc` files, to the cube directory. You can use the CLI to upload files. See [Upload: Add Cube Files](#).

Syntax

```
calc [-verbose] -application appname -db cubename -script scriptfilename
```

Option	Abbreviation	Description
-verbose	-v	Show extended descriptions
-application	-a	Application name
-db	-d	Database (cube) name
-script	-s	Calculation script name. Must have <code>.csc</code> file extension. You do not need to give a full path. Files are assumed to be in the relevant cube directory.

Example

```
esscs calc -v -a Sample -d Basic -s CALCALL.CSC
```

You can also run calculation scripts using the Calculate option in Cube Designer or Smart View, Jobs in the Essbase web interface or REST API, or **execute calculation** in MaxL.

Clear: Remove Data from a Cube

This CLI command clears data from a cube. To run it, you need at least Database Update permission.

Syntax

```
clear [-verbose] -application appname -db cubeName [-option clearOption[-  
regionspec regionSpec]]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-application	-a	Application name
-db	-d	Database (cube) name
-option	-O	Optional. Keyword specifying what to clear. Default option, if omitted, is ALL_DATA. The options for block storage cubes are: <ul style="list-style-type: none"> • ALL_DATA—All data, linked objects, and the outline are cleared • UPPER_LEVEL—Upper level blocks are cleared • NON_INPUT—Non input blocks are cleared The options for aggregate storage cubes are: <ul style="list-style-type: none"> • ALL_DATA—All data, linked objects, and the outline are cleared • ALL_AGGREGATIONS —All aggregated data is cleared • PARTIAL_DATA —Only specified data region is cleared. Use with -regionspec
-regionspec	-rs	MDX expression specifying the region to clear

Example

```
esscs clear -a ASOSamp -d Basic -O PARTIAL_DATA -rs "{([Jan],[Sale],  
[Cash])}"
```

You can also clear data using the Load Data option in Cube Designer, Jobs in the Essbase web interface or REST API, or **alter database DBS-NAME reset** in MaxL.

Createlocalconnection: Save a JDBC Connection

This CLI command creates a JDBC connection and stores it locally. To use it, you need at least Service Administrator role.

Description

You must use this command to create and save the local connection before you can use the CLI [dataload](#) or [dimbuild](#) commands with the streaming option. You must also set an environment variable `EXTERNAL_CLASSPATH` to point to the `.jar` file for your database driver. For examples of setting this variable, see [Build Dimensions and Load Data by Streaming from a Remote Database](#).

Syntax

```
createLocalConnection [-verbose] -name streamConnection -connectionstring
connectionString -user userName [-driver jdbcDriver] [-password password]
```

Option	Abbreviation	Description
-verbose	-v	Show extended descriptions
-name	-N	Connection name
-connectionstring	-cs	JDBC connection string. Format can be with SID, as follows: <pre>jdbc:oracle:thin:@host:port:SID</pre> or with service name, as follows <pre>jdbc:oracle:thin:@host:port/service_name</pre> See Examples.
-user	-u	User name
-driver	-D	JDBC driver. If not provided, Oracle Database is considered the default, as <pre>oracle.jdbc.driver.OracleDriver</pre>
-password	-p	Password (optional)

Examples

The following examples reflect various data sources.

If the `-driver` option and `jdbcDriver` parameter are not provided, Oracle database is the assumed database by default.

Oracle DB – Example with SID (Service ID)

```
esscs createLocalConnection -N OracleDBConnection1 -cs
jdbc:oracle:thin:@myhostname01:1521:ORCL -u OracleUser -D
oracle.jdbc.driver.OracleDriver
```

Oracle DB – Example with Service Name

```
esscs createLocalConnection -N OracleDBConnection2 -cs
jdbc:oracle:thin:@host1.example.com:1521/ORCL.esscs.host1.oraclecloud.com -
u OracleUser
```

DB2

```
esscs createLocalConnection -N DB2conn -cs jdbc:db2://
myhostname02.example.com:50000/TBC -u myDB2User -D
com.ibm.db2.jcc.DB2Driver
```

MySQL

```
esscs createLocalConnection -N MySQLconn -cs jdbc:mysql://
myhostname03.example.com:3306/tbc -u MySQLUsr -D com.mysql.jdbc.Driver
```

Microsoft SQL Server

```
esscs createLocalConnection -N MSSQLConn -cs jdbc:sqlserver://
myhostname04.example.com:1433 -u MSSQLUsr -D
com.microsoft.sqlserver.jdbc.SQLServerDriver
```

Teradata

```
esscs createLocalConnection -N TeraDconn -cs jdbc:teradata://
myhostname05.example.com/DBS_PORT=1025 -u MSSQLUsr -D
com.teradata.jdbc.TeraDriver
```

If you have network connectivity between an external source of data and Essbase, it is most efficient to define application-level or global connections and Datasources in the Essbase web interface. These definitions help you to easily "pull" data from the external source. If you have no network connectivity between Essbase and the external source of data, then you can stream data loads or dimension builds using the CLI by first using this command to create a local connection, and then issuing the `dataload` or `dimbuild` command with the `stream` option; this "push" method is slower.

Dataload: Load Data to a Cube

This CLI command loads data to a cube. To use it, you need at least Database Update permission.

This command requires one of the following sets of options:

- Data file and optional rule file
- Rule file with user name and password
- Stream option referencing a saved local connection

The source database should be accessible within the client network, as not all database drivers can work with Java proxies.

To load data, you must first upload the data load and rule files to the cube directory. You can use the CLI to upload files. See [Upload: Add Cube Files](#).

Syntax

```
dataload [-verbose] -application appName -db cubeName -file fileName [| -
catalogfile catalogFile] [-rule rulesFile | -catalogrulefile
catalogRulesFile] [-user username [-password password]] [-stream] [-
connection connectionName][-query queryString] [-rows n] [-abortOnError]
```

Option	Abbreviation	Description
-verbose	-v	Show extended descriptions
-application	-a	Application name
-db	-d	Database (cube) name
-file	-f	Data load file name. You do not need to give a full path. Files are assumed to be in the relevant database directory. You can use -catalogfile in place of this option.
-rule	-r	Optional. Rule file name. You do not need to give a full path. Files are assumed to be in the relevant database directory. You can use -catalogrulefile in place of this option.
-catalogfile	-CF	Data load file name from the file catalog. You can use this option in place of -file.
-catalogrulefile	-CRF	Rule file name from the file catalog. You can use this option in place of -rule.
-user	-u	Optional. User name. Requires password if used. If you are using a saved connection and Datasource, no user name and password are required. If you are not using a saved connection, and the rule file connects to an RDBMS, specify the user name and password to connect to the RDBMS.
-password	-p	Optional. Password for user. If omitted, user will be prompted for password.
-stream	-S	Optional. Use streaming data load. Requires -conn option if used.
-connection	-conn	Required if streaming option is used. Name of a saved connection that was created using the <code>createlocalconnection</code> CLI command.
-query	-q	Optional. Database query to submit along with the streaming data load.
-rows	-rows	Optional. Number of rows to stream simultaneously. Default is 100.
-abortOnError	-abort	Abort data load if error is encountered

Examples

```
esscs dataload -a Sample -db Basic -f Calcdat.txt -abort true
```

```
esscs dataload -a Sample -db Basic -r Basic.rul -S -conn oraConn -q
>Select * from Data" -rows 50
```

```
esscs dataload -a Sample -db Basic -CF /users/weblogic/Data_Basic.txt -r
Data.rul -abortonerror
```

```
esscs dataload -a Sample -db Basic -CF /users/weblogic/Data_Basic.txt -
CRF /shared/Data.rul -abort
```

```
esscs dataload -a Sample -db Basic -CRF /shared/Data.rul -S -conn
localConnectionName -q "Select * from Table"
```

You can also load data using Cube Designer, Jobs in the Essbase web interface, the Essbase web interface or REST API, or **import data** in MaxL.

Deletefile: Remove Cube Files

This CLI command removes cube artifacts from the application, database, or user home directory. To delete files from a cube, you need at least Database Manager permission for the cube. No special permissions are required to delete files from your user directory.

Syntax

```
deletefile [-verbose] -file fileName [-application application [-db
database] [| -catalogfile catalogFile]]
```

Option	Abbreviation	Description
-verbose	-v	Show extended descriptions
-file	-f	Name of the file to delete
-application	-a	Optional. Application name. If not provided, files are assumed to be in your user home directory.
-database	-db	Optional. Database (cube) name
-catalogfile	-CF	File path and name from the file catalog. You can use this option in place of -file.

Examples

```
esscs deletefile -a Sample -d Basic -f Act1.rul
```

```
esscs deletefile -CF /shared/Data.txt
```

You can also manage files in Cube Designer, the Essbase web interface, or REST API.

Deploy: Create a Cube from a Workbook

This CLI command creates a cube from an Excel application workbook. To do this, you need at least Power User role.

Syntax

```
deploy [-verbose] -file fileName [-application application [-database database] | -catalogfile catalogFile] [-restructureoption restructureOption] [-loaddata] [-recreateapplication] [-createfiles] [-executescript]
```

Option	Abbreviation	Description
-verbose	-v	Show extended descriptions
-file	-f	Name of the application workbook file
-application	-a	Optional. Application name. If not provided, application name will be taken from the workbook.
-database	-db	Optional. Database (cube) name. If not provided, database name will be taken from the workbook.
-catalogfile	-CF	Application workbook from the file catalog. You can use this option in place of -file.
-loaddata	-l	Optional. Load data, if the application workbook contains a data worksheet. Otherwise, only metadata is imported into the cube.
-restructureoption	-R	Optional. Keyword indicating the desired restructuring option. The options for block storage cubes are: <ul style="list-style-type: none"> ALL_DATA—Preserve all data NO_DATA—Preserve no data LEAFLEVEL_DATA—Preserve level 0 (leaf level) data INPUT_DATA—Preserve input data The options for aggregate storage cubes are: <ul style="list-style-type: none"> ALL_DATA—Preserve all data NO_DATA—Preserve no data
-recreateapplication	-ra	Optional. Re-create the application, if it already exists
-createfiles	-cf	Optional. Create cube artifacts in the files directory in Essbase.
-executescript	-e	Optional. Execute calculation scripts. Applicable only if the application workbook contains a calculation worksheet with Execute Calc set to Yes in the definitions.

Examples

```
esscs deploy -v -a SampleD1 -d BasicD1 -f Sample_Basic.xlsx -l -ra -cf -e
```

```
esscs deploy -CF "/gallery/Applications/Demo Samples/Block Storage/
Sample_Basic.xlsx" -a Sample1 -l -cf -e -R ALL_DATA
```

You can also deploy cubes using Cube Designer, or by using the Import option in the **Applications** section of the Essbase web interface.

Dimbuild: Load Dimensions to a Cube

This CLI command loads dimensions to a cube. To do this, you need at least Database Manager permission.

To load dimensions, you must first upload the dimension-build and rule files to Essbase. You can use the CLI to upload files. See [Upload: Add Cube Files](#).

Syntax

```
dimbuild [-verbose] -application appname -db cubename -file fileName [| -
catalogfile catalogFile] -rule rulesFile [| -catalogrulefile
catalogRulesFile]] [-user userName [-password password]] [-stream] [-
connection connectionName][-query queryString] [-rows n]] [-
restructureOption restructureOption] [-forcedimbuild]
```

Option	Abbreviation	Description
-verbose	-v	Show extended descriptions
-application	-a	Application name
-db	-d	Database (cube) name
-file	-f	Dimension build file name. You do not need to give a full path. Files are assumed to be in the relevant application or database directory. You can use -catalogfile in place of this option.
-rule	-r	Rule file name. You do not need to give a full path. Files are assumed to be in the relevant application or database directory. You can use -catalogrulefile in place of this option.
-catalogfile	-CF	Dimension build file name from the file catalog. You can use this option in place of -file.
-catalogrulefile	-CRF	Rule file name from the file catalog. You can use this option in place of -rule.
-user	-u	Optional. User name. Requires password if used. If you are using a saved connection and Datasource, no user name and password are required. If you are not using a saved connection, and the rule file connects to an RDBMS, specify the user name and password to connect to the RDBMS.
-password	-p	Optional. Password for user. If omitted, user will be prompted for password.

Option	Abbreviation	Description
-stream	-S	Optional. Use streaming dimension build. Requires -conn option if used.
-connection	-conn	Required if streaming option is used. Name of a saved connection that was created using the <code>createlocalconnection</code> CLI command.
-query	-q	Optional. Database query to submit along with the streaming dimension build.
-rows	-rows	Optional. Number of rows to stream simultaneously. Default is 100.
-restructureOption	-R	Controls your preservation choices for the outline restructure. For block storage, possible options are: <ul style="list-style-type: none"> • ALL_DATA: Preserve all data when loading dimensions. • NO_DATA: Do not preserve data. • LEAFLEVEL_DATA: Preserve only level 0 data values. If all data required for calculation resides in level-0 members, then you should select this option. All upper-level blocks are deleted before the cube is restructured. When the cube is recalculated, the upper-level blocks are re-created. • INPUT_DATA: Preserve only input data. For aggregate storage, possible options are: <ul style="list-style-type: none"> • ALL_DATA: Preserve all data when loading dimensions. • NO_DATA: Do not preserve data.
-forcedimbuild	-F	Continue the dimension build even if other user activities are in progress. This cancels active user sessions.

Examples

```
esscs dimbuild -a Sample -d Basic -r Basic.rul -u smith -p password -R NO_DATA -F
```

```
esscs dimbuild -a Sample -d Basic -r Basic.rul -S -conn oraConn -q "Select * from Data" -rows 50 -R NO_DATA
```

```
esscs dimbuild -a Sample -db Basic -CRF /users/weblogic/Dim_Market.rul -CF /shared/Market.txt -R ALL_DATA -F
```

You can also load dimensions using Cube Designer, Jobs in the Essbase web interface Essbase web interface or REST API, or **import dimensions** in MaxL.

Download: Get Cube Files

This CLI command downloads cube artifacts from an instance of Essbase to a local directory. You may need to download text files, rule files, or calculation script files from a cube, so you can work on them or upload them to another cube. To download cube artifacts, you need at least Database Update permission.

Syntax

```
download [-verbose] -file filename[ | -catalogfile catalogFile] [-
application appName [-db cubeName]] [-localdirectory path] [-overwrite] [-
nocompression]
```

Option	Abbreviation	Description
-verbose	-v	Show extended descriptions
-file	-f	Name of file to download
-application	-a	Optional. Application name. If not provided, artifacts are downloaded from your user home directory.
-db	-d	Optional. Database (cube) name
-catalogfile	-CF	File in the file catalog. You can use this option in place of -file.
-localdirectory	-ld	Optional. A local directory path
-overwrite	-o	Optional. Overwrite existing file
-nocompression	-nc	Optional. Disable compression of data transfer

Examples

```
esscs download -v -f Product003.rul -a Sample -d Basic -ld c:/temp -o
```

```
esscs download -f Acli.rul -ld c:/temp -o
```

```
esscs download -CF /shared/Acli.rul -ld c:/temp -o
```

You can also manage files in Cube Designer, the Essbase web interface, or REST API.

Help: Display Command Syntax

Displays CLI command level help in the console or terminal.

Syntax

```
[command] -help | -h
```

Examples

```
esscs -help
```

```
esscs -h
```

```
esscs dataload -help
```

LcmExport: Back Up Cube Files

This CLI command backs up cube artifacts to a Lifecycle Management (LCM) .zip file. To do this, you need at least Application Manager permission.

Syntax

```
lcmExport [-verbose] -application appName [-zipfilename filename] [-localDirectory path] [-threads threadscout] [-skipdata] [-overwrite] [-generateartifactlist] [-include-server-level]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-application	-a	Name of application to back up
-zipfilename	-z	Optional. Name of compressed file to hold backup files
-localdirectory	-ld	Optional. A local directory path
-threads	-T	Optional. Number of threads to spawn if using parallel export
-skipdata	-skip	Optional. Do not include data in the backup
-overwrite	-o	Optional. Overwrite existing backup file
-generateartifactlist	-gal	Optional. Generate a text file containing a complete list of the exported artifacts. You can use this text file to manage the import of artifacts. For example, you can rearrange the order of artifacts in the list to control the order in which they are imported. You can skip importing some artifacts by removing or commenting out items in the list.
-include-server-level	-isl	Optional. Include globally defined connections and Datasources as part of the export

Notes

This command, like other CLI commands, can be used from outside the Essbase machine, whereas the LCM utility must be run on the Essbase machine.

Example

```
esscs lcmExport -v -a Sample -z Sample.zip -ld c:/temp -skip -o -gal -isl
```

LcmImport: Restore Cube Files

This CLI command restores cube artifacts from a Lifecycle Management (LCM) .zip file. To do this, you must be the power user who created the application, or a service administrator.

Syntax

```
lcmImport [-verbose] -zipfilename filename [-overwrite] [-targetappName targetApplicationName] [-artifactlist artifactList]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-zipfilename	-z	Name of compressed file containing backup files
-overwrite	-o	Optional. Recreate the target application.
-targetappName	-ta	Optional. Target application name, if you want it to be different from the source name.
-artifactlist	-al	Optional. Name of the file containing the list of artifacts to import. This file can be generated from lcmexport. To skip artifacts, comment out or delete entries from the list. For example, to skip importing audit records, comment out that line, as shown: <pre># -----IMPORT----- import @Provisions import @Databases/Basic #import @Databases/Basic/Audit import @Databases/Basic/Text_files import @Databases/Basic/Xml_files import @Databases/Basic/Calc_scripts import @Databases/Basic/ Open_XML_Excel_files import @Databases/Basic/ScenarioManagement import @Databases/Basic/Provisions import @Databases/Basic/Rule_files</pre> <p>To control import order, rearrange the <code>import</code> entries in the text file.</p> <p>If <code>-overwrite</code> is used, the import operation deletes and recreates the entire application, importing only the artifacts present in the list. If <code>-overwrite</code> is not used, the import operation includes the artifacts specified in the list, without impacting any other artifacts already present in the target application.</p>

Notes

- This command, like other CLI commands, can be used from outside the Essbase machine, whereas the LCM utility must be run within the Essbase machine.
- When partitions exist between cubes being migrated, you must import the data source before the data target. Otherwise, partition definitions may not be restored.

Example

```
esscs lcmImport -z C:/Sample/Sample.zip -o -al C:/Sample/Sample.txt
```

Listapp: Display Applications

This CLI command lists applications that you have access to on this instance of Essbase.

Syntax

```
listapp [-verbose] [-details]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-details	-dtl	Optional. Display more details in the output (application type and current status).

Example

```
esscs listapp -v -dtl
```

Listdb: Display Cubes

This CLI command lists databases that you have access to within a specified Essbase application.

Syntax

```
listdb [-verbose] -application applicationName [details]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-application	-a	Application name
-details	-dtl	Optional. Display status details in the output

Example

```
esscs listdb -v -a Sample -dtl
```

Listfiles: Display Files

This CLI command lists cube artifacts that exist on an instance of Essbase. Cube artifacts may include data files, workbooks, rule files, calculation script files, or other artifacts. Cube artifacts include any files that are needed to perform actions on applications and cubes.

To list files for a cube, you need at least Database Access permission for the cube. No special permissions are required to list files from your user directory.

Syntax

```
listfiles [-verbose] [-type filetype] [-application appname [-db cubename]
| -catalogpath catalogPath]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-type	-t	Optional. File extension/type to display, not including the period. Supported file types are: <ul style="list-style-type: none"> • .csc (calculation scripts) • .rul (rule files) • .txt (text files) • .msh (MaxL scripts) • .xls, .xlsx (Excel workbooks) • .xlsm (macro-enabled Excel workbooks) • .xml (XML files) • .zip (compressed zip files) • .csv (comma-separated files)
-application	-a	Optional. Application name. If not provided, files from your user home directory are displayed.
-db	-d	Optional. Database (cube) name
-catalogpath	-CP	Optional. Catalog path to the filename. Can be used instead of -a [-d] to specify the catalog location of the file(s).

Examples

```
esscs listfiles -t rul -a Sample -d Basic
```

```
esscs listfiles -CP "/shared"
```

You can also manage files in Cube Designer, the Essbase web interface, or REST API.

Listfilters: View Security Filters

This CLI command displays a list of Essbase security filters. You need at least Database Manager permission on the application to see filters for any cubes in the application.

Syntax

```
listfilters [-verbose] -application appname -db cubename
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions

Option	Abbreviation	Description
-application	-a	Application name
-db	-d	Database (cube) name

Example

```
esscs listfilters -v -a Sample -d Basic
```

Listlocks: View Locks

This CLI command displays any locked data blocks or cube-related objects. You need at least Database Access permission on the application to see locks for any cubes in the application.

Syntax

```
listlocks [-verbose] -application appname -db cubename [-object]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-application	-a	Application name
-db	-d	Database (cube) name
-object	-obj	Optional. Display locked files/artifacts.

Example

```
esscs listlocks -v -a Sample -d Basic -obj
```

Listvariables: Display Substitution Variables

This CLI command lists substitution variables defined at the cube, application, or global scope. You need at least Database Access permission to see variables for a cube, Application Manager role to see variables for an application, and Service Administrator role to see global variables.

Syntax

```
listvariables [-verbose] [-application application [-db database]]
```

Option	Abbreviation	Description
-verbose	-v	Show extended descriptions.
-application	-a	Optional. Application name.
-database	-db	Optional. Database (cube) name.

Examples

Cube level

```
esscs listvariables -a Sample -db Basic
```

Application level

```
esscs listvariables -a Sample
```

Global level

```
esscs listvariables
```

Setpassword: Store CLI Credentials

This CLI command stores a password associated with your client/user combination. In subsequent sessions, you can log in without entering a password.

Syntax

```
setpassword [-verbose] -essbaseurl URL -user userName
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-essbaseurl	-url	Address of an instance of Essbase
-user	-u	Your user name

Example

```
esscs setpassword -url https://myEssbase-test-  
myDomain.analytics.us2.example.com/essbase -user rschmidt
```

Start: Start an Application or Cube

This CLI command starts an Essbase application or cube, loading it into memory. To do this, you need at least Database Access permission.

Syntax

```
start [-verbose] -application appname [-db cubeName]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-application	-a	Application name
-db	-d	Optional. Database (cube) name

Example

```
esscs start -v -a Sample -d Basic
```

Stop: Stop an Application or Cube

This CLI command stops an Essbase application or cube. To do this, you need at least Database Access permission.

Syntax

```
stop [-verbose] -application appname [-db cubename]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-application	-a	Application name
-db	-d	Optional. Database (cube) name

Example

```
esscs stop -v -a Sample -d Basic
```

Unsetpassword: Remove Stored CLI Credentials

This CLI command removes stored login credentials associated with your client/user combination, reversing the effect of `setpassword`.

Syntax

```
unsetpassword [-verbose] -essbaseurl URL -user userName
```

Option	Abbreviation	Description
-verbose	-v	Show extended descriptions
-essbaseurl	-url	Address of an instance of Essbase
-user	-u	The user whose password to unset

Example

```
esscs unsetpassword -url https://myEssbase-test-  
myDomain.analytics.us2.example.com/essbase -u rschmidt
```

Upload: Add Cube Files

This CLI command uploads cube artifacts from a local directory to an instance of Essbase.

To perform tasks such as data loads, dimension builds, calculations, or other operations, you may need to upload data files, rule files, calculation script files, or other artifacts to the cube directory. You can also upload the artifacts to your user directory.

To upload files to a cube, you need at least Database Manager permission. No special permissions are required to upload to your user directory.

Syntax

```
upload [-verbose] -file filename [-application appname [-db cubeName] | -
catalogpath catalogPath] [-overwrite] [-nocompression][  
compressionAlgorithm]
```

Option	Abbreviation	Description
-verbose	-v	Optional. Show extended descriptions
-file	-f	Name of file to upload
-application	-a	Optional. Application name. If not provided, files are uploaded to your user directory, or to the catalog path specified in -CP.
-db	-d	Optional. Database (cube) name. Requires -a.
-catalogpath	-CP	Optional. Catalog path to the filename. Can be used instead of -a [-d] to specify the catalog location of the file.
-overwrite	-o	Optional. Overwrite existing file
-nocompression	-nc	Optional. Disable compression of data transfer
-compressionAlgorithm	-ca	Optional. Available if -nc is not used. Defines which compression algorithm to use for data transfer. Possible choices: gzip or lz4 . <ul style="list-style-type: none"> gzip—Default if compression is used. Provides smaller data transfer with slower calculation. lz4—Provides faster calculation with a slower data transfer. Usage examples: <pre>-ca gzip</pre> <pre>-ca lz4</pre>

Examples

```
esscs upload -v -f c:/temp/Maxl01.msh -a Sample -d Basic -o -ca lz4
```

```
esscs upload -f C:/temp/Act1.rul -CP /shared
```

You can also manage files in Cube Designer, the Essbase web interface, or REST API.

Version: Display API Version

This CLI command gets the version of the REST API that is associated with this instance of Essbase.

Syntax

```
version
```

Example

```
esscs version
```

19

Manage Essbase Using the MaxL Client

MaxL is a multi-dimensional database access language for Essbase.

To run MaxL scripts or statements, you must use the MaxL Client to issue the statements over HTTP or HTTPS.

Prerequisites to Set Up the MaxL Client

Complete these tasks before you download and use the MaxL Client. To execute MaxL scripts or statements, you must be a power user or administrator.

To prepare for using the MaxL Client,

1. Get the URL for the Essbase instance from your Service Administrator. Its basic format is:

```
https://IP-address:port/essbase
```

2. Using a web browser or cURL, test that you can reach the discovery URL from the client host. A discovery URL is the URL provided by your Service Administrator, with `/agent` appended to the end. Here is a cURL example:

```
curl https://192.0.2.1:443/essbase/agent --tlsv1.2
```

3. Set up the SSL certificate, if applicable to your organization.
 - If you're using one of these deployment types, a Trusted CA Signed SSL Certificate is included:
 - Oracle Analytics Cloud
 - Oracle Analytics Cloud with Identity Cloud Service (IDCS) and Load Balancing
 - Cloud at Customer with Load Balancing
 - If you're using Oracle Analytics Cloud or Cloud at Customer with LDAP (without Load Balancing), use a self-signed certificate.
 - If you're using Essbase deployed on Oracle Cloud Infrastructure, see Set Up the SSL Certificate.
4. To check if a certificate is trusted, paste the discovery URL into a web browser. If **https** is green or a label says "Secure," it is trusted. If **https** is red or a label says "Not secure," it is untrusted.
5. If the certificate is untrusted (self-signed), import it to the client trust store (cacert.pem).
6. The client verifies the server's digital certificate using a provided ca-bundle certificate store. Provide the ca-bundle location by specifying the environment variable:

API_CAINFO=CA certificate file path;

If the path is not provided, the Essbase Runtime Client will try to get ca-bundle from the default OpenSSL installation location (applicable for Linux and Macintosh).

Oracle Data Visualization clients and MaxL Client include a ca-bundle (cacert.pem).


If you need a ca-bundle (cacert.pem), you can also download it. One sample source is: <https://curl.haxx.se/docs/caextract.html>.

Download and Use the MaxL Client

The Essbase MaxL Client enables you to use MaxL over HTTP or HTTPS. MaxL is an administrative, language-based interface for managing cubes and artifacts. Be sure you are using the latest client version provided in the Console, as older, previously downloaded versions may not work correctly.

To run MaxL statements, you must be a power user or an administrator. Before you download the MaxL Client, see [Prerequisites to Set Up the MaxL Client](#).

If you're a [federated](#) SSO user in Oracle Identity Cloud Service, logging in using MaxL or CLI is not supported. Federated SSO login requires a browser window. Create a native Identity Cloud Service user, and use that instead to log in using MaxL or CLI.

1. In the Essbase web interface, click **Console**.
2. In the Console, go to **MaxL Clients**.
3. Click **Download**
 next to the MaxL Client that is appropriate for your platform.
4. Save the compressed `EssbaseMaxl` file to your local drive.
5. Extract the contents of the compressed file to a folder.
6. If you're using a proxy, you must set the correct proxy in the MaxL execution script, `startMAXL.bat` or `startMAXL.sh`. The following example, applicable for editing `startMAXL.sh` for UNIX, tells MaxL to use the designated proxy (proxy.example.com), but to bypass using a proxy for the specific destinations listed in the exception list (127.0.0.1, localhost, and something.example.com).

```
export https_proxy=http://proxy.example.com
export no_proxy=127.0.0.1,localhost,something.example.com
```

For Windows, `startMAXL.bat` can be edited similarly but with different syntax.

```
set proxy proxy-server="https://proxy.example.com" bypass-
list="127.0.0.1;localhost;*.example.com"
```

7. If you're using Essbase deployed on Oracle Cloud Infrastructure and are using a self signed certificate, you must disable peer verification in the MaxL execution script. **Caution:** this solution should be only temporary, until you can obtain a trusted CA certificate. Here is an example using **bash** (for `startMAXL.sh`):

```
export API_DISABLE_PEER_VERIFICATION=1
```

8. Run the `startMAXL` batch or shell script. A command prompt opens, the environment setup completes, and the MaxL Client starts up.
9. Log in by providing your credentials and the Essbase URL in the MaxL **login** statement.

In the following example, the user logging in, `admin1@example.com` is an Identity Cloud Service administrator who was set as the initial Essbase administrator during Essbase stack deployment on Oracle Cloud Infrastructure. As the password is not entered in this example, the administrator will be prompted to provide it next. The URL is the **essbase_url** from the job outputs resulting from the stack deployment.

```
login admin1@example.com on "https://192.0.2.1/essbase";
```

Any Identity Cloud Service user provisioned to work with Essbase can log in to MaxL, as long as they are provisioned as a power user or administrator.

10. Execute an interactive MaxL statement.

For example:

```
display database all;
```

To learn more about MaxL, see [MaxL Statement Reference](#).

Analyze Data in the Web Interface

For convenience, you can perform analysis on cube data from the Essbase web interface.

To analyze data grids in the Essbase web interface,

1. Log in to Essbase with at least Database Access role for the application whose cube data you want to analyze.
2. On the Applications page, expand the application, and highlight the row containing the cube name.
3. From the **Actions** menu to the right of the cube name, click **Analyze Data**.

A grid is displayed in the Ad Hoc Analysis tab. In this tab, you can:

- Perform ad hoc analysis against the cube you selected when you opened the Analyze Data view.
- Save a grid layout that you can refresh when you use the Ad Hoc Analysis tab in the future.

On the Reports tab, you can use MDX to write sophisticated data queries to populate the grid and to save as named reports.

Perform Ad Hoc Analysis in the Web Interface

In the Ad Hoc Analysis tab of the Analyze Data view, a grid is displayed containing each of the base dimensions (non-attribute dimensions) from the cube.

You may or may not see data in the ad hoc grid, depending on your filter access and how data is stored in the cube. Data is not always stored at the topmost member for every dimension hierarchy.

Use the ad hoc navigation buttons at the top left of the Ad Hoc Analysis tab to navigate to data that you are allowed to see. If your filter grants you write permission on the cube, the **Submit** button enables you to update data for stored intersections within the scope of your filtered access.

Layouts		A	B	C	D	E	F
Name	Actions	1	Product	Market	Scenario		
No items to display.		2	Measures				
		3	Year	105522			
		4					
		5					

Work with Layouts

If you create a grid that you would like to use again in the future, you can save it at any time as a Layout.

To create a layout,

1. In the Analyze Data view for your cube, on the Ad Hoc Analysis tab, create an ad hoc grid that you want to save.
2. Click **Save Layout**.
3. Enter a name for your layout, and optionally, a description.
4. If you want to see this grid each time you analyze data, instead of the database default ad hoc query, check the **default layout** box.
5. Click **Save**.

The last ad hoc grid that was rendered during your session will be displayed the next time you log in, unless a default is set.

To view a grid previously saved as a layout,

1. If layouts are not listed by name in the Ad Hoc Analysis tab, click the **Layouts Panel** button to display the list.
2. Click the name of a stored layout to render it in the grid.

Layouts	
Name	Actions
layout1	
layout2	

To delete or edit layouts that you created, use the Actions menu next to the layout name. The Edit option allows you to select the layout as your default, update the description, or remove the default setting on a layout previously set as your default.

Access to Layouts

How you work with layouts depends on your cube access.

Clicking on a saved layout name causes it to render data in the Ad Hoc Analysis tab of the Analyze Data view.

Users with, at minimum, the application-level role of Database Manager can:

- See and render layouts created by others for this cube.
- Designate a layout to be the database default. This layout is shown to all cube users when they analyze data, unless they have previously created their own user default layouts.
- Delete layouts created by any user of this cube.

Layouts and reports are included when the cube is copied or moved using migration, export, and Lifecycle Management (LCM) tools.

Analyze and Manage Data with MDX

MDX (Multidimensional Expressions) is a powerful data manipulation and querying language.

With MDX, you can:

- Query and report against data and metadata in Essbase cubes
- Insert data into an Essbase cube
- Export data from an Essbase cube

An MDX query is a single MDX statement, having exactly one result set, that applies to a single cube.

An MDX report is a single MDX query, saved in the cube context. You can access MDX reports from Smart View and from the Essbase web interface.

An MDX script is a file, with an `.mdx` extension, that you can upload and then run from Jobs or in Smart View. Only MDX Insert and Export statements should be used in MDX scripts. To analyze grid data, use MDX reports rather than MDX scripts.

Topics:

- [Analyze Data with MDX Reports](#)
- [Insert and Export Data with MDX](#)
- [Run MDX Scripts](#)

Analyze Data with MDX Reports

You can store and render queries in the Essbase web interface using MDX reports. The minimum permission required to create a report is Database Manager.

Defining Layouts using the Ad Hoc Analysis tab may not always be the most efficient way to create a sophisticated report. If you know exactly what you want to query, you can use MDX to create a query to populate the grid.

To create an MDX report:

1. Log in to the Essbase web interface as a Database Manager or higher role.
2. On the Applications page, expand an application and select a cube.
3. Click the Actions menu to the right of the cube name, and select **Analyze Data**.
4. In the Analyze view, select the **Reports** tab and click **Create**.
5. Enter a name for the report, and optionally, a description.
6. In the Query field, enter an MDX query relevant to the current cube. For example:

```
SELECT
  {[West].children}
ON COLUMNS,
  {[Diet].children}
ON ROWS
```

The query must contain both row and column axes specifications. In other words, the query syntax must include specifications for both ON COLUMNS and ON ROWS, even if only an empty set {} is specified for one axis.

Because the context of Analyze Data is the active cube, we recommend that you omit the optional cube specification (the FROM clause) from MDX reports. Omitting the FROM clause allows for more flexibility—if the cube is copied or renamed, the report will work in the new cube.

Substitution variables are supported in MDX reports, but not runtime substitution variables. To use runtime substitution variables, save the MDX query as a script, and run it from Smart View using **Calculate** on the Essbase ribbon.

7. Click **Validate** to verify your MDX syntax, and then click **Save**.
8. From the Reports panel on the left, select the saved report to render a grid.

To learn more about MDX, see MDX and Writing MDX Queries.

Access to MDX Reports

How you work with reports depends on your cube access.

Users with, at minimum, the application-level role of Database Access can render saved MDX reports created by others. The data a user sees displayed in the report depends on that user's filter access.

In addition to rendering saved reports, Database Access users can export result sets in various formats: HTML, CSV, Excel, and JSON.

Database Access users can also view the MDX query that defines the report, by clicking the **Actions** menu next to the report name and selecting **View**.

If you have at least Database Manager role, you can use reports in the same ways that Database Access users can. Additionally, you can edit and delete reports using the **Actions** menu.

If you are a Service Administrator, you can additionally use the **Execute As** button to impersonate other users and check their data access. This can be useful for testing filters assigned to various users.

Examples of MDX Reports

The MDX examples in this section demonstrate special types of analyses you can perform, using MDX reports, that are not easily accomplished in the Ad Hoc Analysis view.

The following examples are designed to work on the Sample Basic cube.

Metadata Report

The following example returns only metadata (member names, but no data):

```
SELECT
  {[Product].Levels(1).Members}
ON ROWS,
  {}
ON COLUMNS
```

returning the grid:

	A
1	100
2	200
3	300
4	400
5	Diet

Attribute Report

The following example uses, on columns, members from an attribute dimension:

```
SELECT
  [Product].Children
ON ROWS,
  [Ounces].Children
ON COLUMNS
WHERE {Sales}
```

returning the grid:

	A	B	C	D	E
1		Ounces_32	Ounces_20	Ounces_16	Ounces_12
2	100	#Missing	#Missing	12841.0	93293.0
3	200	#Missing	#Missing	49990.0	59096.0
4	300	#Missing	64436.0	#Missing	36969.0
5	400	84230.0	#Missing	#Missing	#Missing
6	Diet	#Missing	#Missing	38240.0	67438.0

Filtered Report

The following example uses a slicer (WHERE clause) to limit the query to Cola. Additionally, the Filter function limits the level 0 markets in the query to those that have a negative profit.

```
SELECT
  { Profit }
ON COLUMNS,
  Filter( [Market].levels(0).members, Profit < 0)
ON ROWS
WHERE {Cola}
```

returning the grid:

	A	B
1		Profit
2	Oregon	-234.0
3	Utah	-31.0
4	Nevada	-210.0
5	Oklahoma	-102.0
6	Louisiana	-305.0
7	Ohio	-22.0
8	Wisconsin	-310.0
9	Missouri	-87.0
10	Iowa	-874.0

UDA Report

The following example shows Product data for Market dimension members that have a user defined attribute (UDA) of "Major Market." A slicer (WHERE clause) limits the query to include only Sales data.

```
SELECT
  [Product].Children
ON ROWS,
  {Intersect(UDA([Market], "Major Market"), [Market].Children)}
ON COLUMNS
WHERE {Sales}
```

returning the grid:

	A	B	C
1		East	Central
2	100	27740.0	33808.0
3	200	23672.0	29206.0
4	300	20241.0	33215.0
5	400	15745.0	33451.0
6	Diet	7919.0	42660.0

Insert and Export Data with MDX

In addition to being useful for grid-based analysis, MDX also enables you to copy and update subsets of multidimensional data.

The MDX Insert clause enables you to update the cube with data, either from another cube or from a calculated (non-physical) member that you define using MDX.

The MDX Export clause enables you to save and export query results as data subsets that you can view or import later.

Insert and Export MDX statements can be run as saved MDX scripts.

To learn more about MDX Insert and Export, see [MDX Insert Specification](#) and [MDX Export Specification](#).

Run MDX Scripts

Use MDX scripts when you need to execute Insert or Export data operations.

For analysis of grid data, use MDX reports. See [Analyze Data with MDX Reports](#).

To use MDX scripts, select a workflow:

- [Write, Upload, and Run an MDX Script](#)
- [Write an MDX Script in the Script Editor and Run It](#)
- [Create an MDX Script in Cube Designer and Run it](#)

Write, Upload, and Run an MDX Script

Use this workflow to write MDX scripts in a text editor and upload them to Essbase.

1. Write the MDX script in a text editor, and save it with an `.mdx` extension.
2. Upload the MDX script to the application or cube directory under **Files** in the Essbase web interface.
3. Run the MDX script from **Jobs** or from Smart View, using **Calculate** on the Essbase ribbon.

Write an MDX Script in the Script Editor and Run It

Use this workflow to write MDX scripts in a script editor on the cube, and run them from **Jobs**.

1. On the Applications page, expand an application and cube.
2. From the cube's Actions menu, click **Inspect**.
3. Click **Scripts**, and then click **MDX Scripts**.
4. Click **+** to open a script editor.
5. Write the MDX script. A member tree and function list can help you.
6. Validate and save the script, then close the script editor.
7. Run the MDX script from **Jobs** (see [Run MDX](#)), or if using Smart View, using **Calculate** on the Essbase ribbon.

Create an MDX Script in Cube Designer and Run it

Use this workflow to create MDX scripts using an application workbook, and run them from **Jobs**.

1. In an application workbook, create an MDX worksheet. See [Work with MDX Worksheets in Cube Designer](#).
2. Add a file name in the **File Name** field.
3. Indicate, in the **Execute MDX** field, whether to execute the MDX at the time the cube is created. Valid entries are **Yes** and **No**.
4. Add the MDX script below the **Script** line.
5. Save the application workbook.
6. Build the cube. See [Create an Application and Cube in Cube Designer](#).
7. Run the MDX script from **Jobs**, or if using Smart View, using **Calculate** on the Essbase ribbon.

Guidelines for MDX Scripts

Use the following guidelines when working with MDX scripts.

- Use MDX scripts to perform Insert or Export data operations.
- For grid analysis, use MDX reports instead of MDX scripts.
- MDX scripts can optionally include runtime substitution variables.
 - To be usable within Smart View, MDX scripts with runtime substitution variables must use the XML syntax within the SET RUNTIMESUBVARS calculation command, including <RTSV_HINT>.
 - To set a runtime substitution variable so that it calculates only the visible slice of data in Smart View, set the value of the runtime substitution variable to `POV`, and set the data type to `member`.
 - When run from the Essbase web interface, your MDX scripts may use substitution variables, but not runtime substitution variables. To use runtime substitution variables in MDX scripts, you must run the scripts from Smart View, using **Calculate** on the Essbase ribbon.

[Use Substitution Variables](#)

Examples of MDX Scripts

The following are examples of MDX scripts you can run on the Sample Basic cube, either from Jobs or in Smart View.

MDX Insert

You can save this `.mdx` script and run it from **Jobs** or from the **Calculate** dialog in Smart View.

```
INSERT "([Measures].[Payroll])" TO "([Measures].[Revised_Payroll])"  
INTO [Sample].[Basic]  
FROM (
```

```

SELECT
    {[Measures].[Payroll]} ON COLUMNS,
    {Crossjoin
        (Crossjoin(Descendants([Year]),
            Crossjoin(Descendants([Scenario]),
                Descendants([Product]))),
            Descendants([Market]))} ON ROWS
FROM [Sample].[Basic]
);

```

The above example assumes you have previously added a Revised_Payroll measure to Sample Basic.

MDX Export

You can save this .mdx script and run it from **Jobs** or from the **Calculate** dialog in Smart View.

```

EXPORT INTO FILE "sample01" OVERWRITE
SELECT
    {[Mar],[Apr]}
ON COLUMNS,
    Crossjoin({[New York]},
        Crossjoin({[Actual],[Budget]},
            {[Opening Inventory],[Ending Inventory]}))
ON ROWS
FROM [Sample].[Basic]
WHERE ([100-10])

```

After you run the script, the following export file, sample01.txt, is saved in the cube directory of the file catalog:

```

Market,Scenario,Measures,Mar,Apr
New York,Actual,Opening Inventory,2041,2108
New York,Actual,Ending Inventory,2108,2250
New York,Budget,Opening Inventory,1980,2040
New York,Budget,Ending Inventory,2040,2170

```

MDX Export Using Runtime Substitution Variable

You can save this .mdx script and run it from the **Calculate** dialog in Smart View.

```

SET RUNTIMESUBVARS
{
    States = "Massachusetts"<RTSV_HINT><svLaunch>
        <description>US States</description>
        <type>member</type>
        <allowMissing>>false</allowMissing>
        <dimension>Market</dimension>
        <choice>multiple</choice>
        </svLaunch></RTSV_HINT>;
};
EXPORT INTO FILE "sample002" OVERWRITE
SELECT

```

```
{[Mar],[Apr]}  
ON COLUMNS,  
  Crossjoin({&States}, Crossjoin({[Actual],[Budget]},  
  {[Opening Inventory],[Ending Inventory]}))  
ON ROWS  
FROM [Sample].[Basic]  
WHERE ([100-10])
```

After you run the script, the following export file, `sample002.txt`, is saved in the cube directory of the file catalog:

```
Market,Scenario,Measures,Mar,Apr  
Massachusetts,Actual,Opening Inventory,-54,-348  
Massachusetts,Actual,Ending Inventory,-348,-663  
Massachusetts,Budget,Opening Inventory,-160,-520  
Massachusetts,Budget,Ending Inventory,-520,-910
```


21

Use Logs to Monitor Performance

You can download and view logs at the applications level. You can also use Performance Analyzer, which analyzes Essbase logs and provides usage and performance statistics.

- [Download Application Logs](#)
- [About Performance Analyzer](#)

Download Application Logs

As an Application Manager, you can download applications logs. You can download the latest log, as well as rolled over logs. You can also view logs without downloading them.

1. On the Applications page, select the application.
2. To the right of the application name, click the Actions menu and select **Inspect**.
3. On the **Logs** tab, click the Download icon under **Latest**, the View icon under **Latest**, or the Download icon under **All**.
4. If you're downloading, save the file locally.

About Performance Analyzer

The Performance Analyzer, available in the Console of the Essbase web interface, helps you monitor usage and performance statistics of your Essbase service.

Performance Analyzer reads log files behind the scenes, scanning them at intervals that you specify. From the log files it creates .csv files of Essbase activity data. The data comes from the application ODL log, agent log, and WebLogic logs.

After a Performance Analyzer file grows to 10 MB, a new file is created. Essbase keeps a total of 112 files, at which point Essbase deletes the oldest file first. The most recent file is called EssbaseHpa_Data.csv. The older files are named numerically; for example, EssbaseHpa_n_Data.csv.

A template in the Essbase web interface, in **Files/gallery/System Performance/Health and Performance Analyzer**, can help you learn more about Performance Analyzer. To use the gallery template, you copy and paste CSV data into the template.

Because each .csv file contains time-stamped information from your logs in chronological order, you can use a database or reporting utility of your choice to:

- combine .csv files or file portions to create performance analysis for precise time intervals.
- build charts or other visualizations of the data.

Enable Performance Analyzer and Set the Data Collection Interval

If you are a service administrator, you can enable the Performance Analyzer in the Console of the web interface to capture information from log files about usage and performance. You can also set the interval at which Essbase captures the CSV data.

1. In the web interface, choose **Console**.
2. Click **Performance Analyzer**.
3. Click **Settings**.
4. In the **Settings** dialog box, use the toggle switch to enable **Performance Analyzer**.
5. Choose the interval at which you want new .csv files to be created. The default is every fifteen minutes.

Understand and Work With Performance Analyzer Data

Performance Analyzer generates CSV data based on logs and organizes it into columns. First, you gather the CSV data and open the .csv files in Excel, and then you can examine and work with the data using Excel filtering tools.

To gather the CSV data:

1. Locate the .csv files you want to analyze.
 - a. In the Essbase web interface, select **Console**.
 - b. Select **Performance Analyzer**.
 - c. Find the .csv file or files matching the time period you are interested in.
2. Download the files:
 - a. Select the download icon under **Actions** to download each file.
 - b. Repeat for additional files you want to download.

Open the files in Excel and examine the columns at the top of the files. Most of the columns are self-explanatory. They contain data helpful for filtering performance analysis, such as application and cube name, time stamp, and date.

Columns N and O need further discussion, as they contain key information. Column N contains information such as configuration settings, database settings, and user logins. Column O contains specific entries within those categories. In Excel, you can filter on column N and choose a category, and then filter on column O to choose specific entries within those categories.

Column N (Operation.OperationType) describes the type of the log message:

- **UserLogin** shows how long the user was active, and when the user logged out.
- **UserOperation** shows all user operations, such as data loads, calculations, and restructures. It also shows errors and exceptions.
- **SystemOperation** shows CPU, memory, disk, and I/O usage.
- **DBSettings** shows database statistics.
- **ConfigurationSetting** shows configuration settings.

- **Notification** identifies when there is a severe error.

If you filter on column N and then choose the specific category you are interested in, you can then view events within that category by filtering on column O.

Example view of a filter on column N:

- (Select All)
- ConfigSettings
- DbSettings
- Notification
- SystemOperations
- UserLogin
- UserOperations

Example view of a filter on column O:

- (Select All)
- Bytes Read
- Bytes Written
- Cpu usage in %
- Disk Usage in KB
- Memory Free in MB
- Memory usage in %
- Memory Used in MB
- Process Size in bytes
- RSS Size
- Swap Free in MB

Analyze Cube Data with Drill Through Reports

Sometimes you may require more information than what existed in the Essbase cube. You can access and analyze additional data using drill through reports.

- [About Drill Through Reports](#)
- [Create Drill Through Reports](#)
- [Execute Drill Through Reports](#)

About Drill Through Reports

When you want more information than what you can see in the Essbase cube, you can use drill through reports to access external data sources.

Drill through refers to linking the Essbase cube to further data, for example, transactional-level data stored in a relational database.

You can drill through to data from any other Oracle application, an external database, a file (delimited or Excel), or a URL-based target.

You can also select multiple cells or multiple ranges of cells, and see the merged results in drill through. Selections can be recursive, non-recursive, level 0, or non-contiguous. URL drill through is not supported for multiple cell selection.

You can create drill through reports in the Essbase interface or generate them using Smart View. You can also do a URL-based drill through from Smart View, by opening the URL provided in the report, in a browser.

Access to Drill Through Reports

How you work with drill through reports depends on your level of access.

A user role of Database Manager is required to create drill through reports on a cube. If the drill through report accesses one or more Datasources defined at the application level, a prerequisite assumption is that a connection and Datasource were already defined at the application level, by at least an Application Manager.

The Application Manager who creates the connection and Datasource must additionally have appropriate credentials to access the external source of data; for example, if the external source data is a SQL source, the Application Manager must have credentials to log in to the SQL source, in order to create the connection.

Power User is the minimum permission to create the application and cube in the first place. A Power User has implicit Application Manager permission for the applications he or she created, but not for all applications.

Any user with Database Access can access the drill through report, as long as the user's filter does not restrict access to the cells within the drillable region defined for

the drill through report. A drillable region is a specification that indicates the cell intersections from which the drill through report is accessible from Smart View.

Typical Workflow for Drill Through Reports

The workflow for running a drill through report is based upon your defined connection and data source.

You can use an application workbook, an external data source file or URL, and Cube Designer to set up an Essbase cube for drill through.

You can then use the report to analyze the cube that accesses the data source.

1. Create a connection to the data source type.
2. Define a data source and save it as part of the application.
 - a. Create a connection.
 - b. Define the data source.
 - c. Select report columns and modify data types as necessary.
 - d. Define aliases if relevant.
 - e. Define source-specific parameters if any.
 - f. Preview the data.
3. Create the drill through report.
 - a. Select the report type - Datasource or URL - and enter the details.
 - b. If you selected a Datasource report, choose the columns to display.
 - c. You can specify or add drillable regions.
4. Execute the drill through report. Use the reports to analyze an Essbase cube that accesses the data source.

Use Cases and Column Mapping

For drill through reports, you must map a data source column to a dimension, generation of a dimension, or to a level 0.

The following are examples of mapping data source columns:

- Product column, which contains SKU data, can be mapped to Product SKU in the following hierarchy: Product dimension > Products > Category > Product SKU
- Year column, which contains Month data, can be mapped to Month, in the following hierarchy: Year dimension > Year > Quarter > Month
- Scenario column, which is defined as actual or budget, can be mapped directly to the Scenario dimension — this is a flat dimension without any generations

Essbase adds a filter condition is added to the drill through report query based upon the column mapping and the related intersection in Smart View.

For column mapping and use cases descriptions, see:

- [Map Dimension to a Data Source Column](#)
- [Map Generation Name to a Data Source Column](#)

- Map Level 0 to a Data Source Column
- Map Multiple Cells and Regions

Map Generation Name to a Data Source Column

This use case maps a generation name to a data source column. Drill through results contain members that match the mapped generation members.

The mapping is as follows:

Product – Product SKU, Region – Region, Year – Month

The selected columns in the column mapping are: Product, Region, Market, Year, and Sales. In this case, the datasource is the Excel file, Excel_DS.

The generated query is:

Select Product, Region, Market, "Year" from Excel_DS where Product = <SKU value> and Region = <Region value> and "Year" = <month value>

	A	B	C	D	E	F	G	H	I	J	K
1				Scenario							
2				Sales	COGS	Margin	Total Expe	Profit	Inventory	Ratios	Measures
3	East	Cola	Jan	1812	599	1213	376	837	4643	66.9426	837
4	East	Cola	Feb	1754	588	1166	374	792	4253	66.47662	792
5	East	Cola	Mar	1805		1209	377	832	3912	66.98061	832
6	East	Cola	Qtr1	5377		3588	1127	2461	4643	66.8032	2461
7	East	Cola	Qtr2	6024	1903	4121	1181	2940	3747	68.40969	2940
8	East	Cola	Qtr3	6505	2001	4504	1206	3298	3598	69.23905	3298
9	East	Cola	Qtr4	5305	1756	3549	1119	2430	1898	66.89915	2430
10	East	Cola	Year	23205	7443	15762	4633	11129	4643	67.92502	11129
11	East	Diet Cola	Jan	200	84	116	49	67	500	58	67
12	East	Diet Cola	Feb	206	86	120	49	71	490	58.25343	71
13	East	Diet Cola	Mar	214	89	125	51	74	481	58.41121	74
14	East	Diet Cola	Qtr1	620	259	361	149	212	500	58.22581	212
15	East	Diet Cola	Qtr2	822	344	478	175	303	502	58.15085	303
16	East	Diet Cola	Qtr3	845	353	490	178	312	692	58.12574	312
17	East	Diet Cola	Qtr4	783	327	456	169	287	656	58.23755	287
18	East	Diet Cola	Year	3068	1283	1785	671	1114	500	58.18123	1114
19	East	Caffeine Free Cola	Jan	93	38	55	35	20	241	59.13978	20
20	East	Caffeine Free Cola	Feb	101	41	60	35	25	236	59.40594	25
21	East	Caffeine Free Cola	Mar	107	43	64	35	29	231	59.81308	29

select Product, Region, Market, "Year", Sales from "Excel_DS" where "Year" = 'Jan' AND Product = '100-10' AND Region = 'East'

	A	B	C	D	E
1	PRODUCT	REGION	MARKET	Year	SALES
2	100-10	East	New York	Jan	678
3	100-10	East	New York	Jan	640
4	100-10	East	Massachusetts	Jan	494
5	100-10	East	Massachusetts	Jan	460
6	100-10	East	Florida	Jan	210
7	100-10	East	Florida	Jan	190
8	100-10	East	Connecticut	Jan	310
9	100-10	East	Connecticut	Jan	290
10	100-10	East	New Hampshire	Jan	120
11	100-10	East	New Hampshire	Jan	110
12					

The report is executed on member Jan that is mapped to generation Month. The results are shown for month Jan.

Recursive Drill Through in Generation Mapping

This use case maps a generation name to a column name, where the report is executed on any top generation.

In this use case, execute the drill through report on Year member and map it to the Month generation. The generated query doesn't have a where condition for Month.

The result includes all data for Year column in the data source column (all Months).

When there is no mapping to a particular generation, find the generations under the selected generation. Verify if column mapping exists to any of these generations in the same dimension. If it exists, get the children from that generation and generate a query where all of these members are added in the Where condition.

	A	B	C	D	E	F	G	H	I	J	K	L
1				Scenario								
2				Sales	COGS	Margin	Total Expe	Profit	Inventory	Ratios	Measures	
3	East	Cola	Year	23205	7443	15762	4633	11129	4643	67.92502	11129	
4	East	Diet Cola	Year	3068	1283	1785	671	1114	500	58.18123	1114	
5	East	Caffeine Free Cola	Year			871	458	413	241	59.37287	413	
6	East	Colas	Year	21		18418	5762	12656	5384	66.3951	12656	
7	East	Root Beer	Year	21		12200	9666	2534	5957	51.53768	2534	
8	East	Cream Soda	Year	20241	10934	9307	6680	2627	6278	45.98093	2627	
9	East	Fruit Soda	Year	15745	6199	9546	3202	6344	8125	60.62877	6344	
10	East	Diet Drinks	Year	7919	3362	4557	2149	2408	1867	57.54514	2408	
11	East	Product	Year	87398	37927	49471	25310	24161	25744	56.60427	24161	
12	West	Cola	Year	14862	6059	8803	4210	4593	3348	59.2316	4593	
13	West	Diet Cola	Year	8923	5216	3707	4241	-534	3236	41.54432	-534	
14	West	Caffeine Free Cola	Year	4521	2892	1629	2139	-510	2008	36.03185	-510	
15	West	Colas	Year	28306	14167	14139	10590	3549	8592	49.95054	3549	
16	West	Root Beer	Year	34200	15144	19056	9329	9727	11755	55.7193	9727	
17	West	Cream Soda	Year	35391	15442	19949	9218	10731	8880	56.36744	10731	

	A	B	C	D	E
1	PRODUCT	REGION	MARKET	Year	SALES
2	100-20	East	Florida	Jan	200
3	100-20	East	Florida	Jan	190
4	100-20	East	Florida	Feb	206
5	100-20	East	Florida	Feb	190
6	100-20	East	Florida	Mar	214
7	100-20	East	Florida	Mar	200
8	100-20	East	Florida	Apr	267
9	100-20	East	Florida	Apr	250
10	100-20	East	Florida	May	273
11	100-20	East	Florida	May	250
12	100-20	East	Florida	Jun	282
13	100-20	East	Florida	Jun	260
14	100-20	East	Florida	Jul	336
15	100-20	East	Florida	Jul	310
16	100-20	East	Florida	Aug	277
17	100-20	East	Florida	Aug	260
18	100-20	East	Florida	Sep	230
19	100-20	East	Florida	Sep	210
20	100-20	East	Florida	Oct	218


```
select Product, Region, Market, "Year", Sales from "Excel_DS" where
Product = '100-20' AND Region = 'East'
```

The data source column is mapped to the Month generation in the Year dimension.

- Generations for Year dimension: History, Quarter, Month
- Column mapping for Year (dsColumn) == Month (gen)

```
"columnMapping" : {
  "Product" : "Product SKU",
  "Region" : "Region",
  "\"Year\"" : "Month",
  "Scenario" : "Scenario"
},
```

Top Level

When the report is executed with Year in the intersection, the actual generation name is History, which is not mapped. The next generation is Quarter, which not mapped. The following generation is Month, which is mapped.

In the Year dimension, get all members from Month generation:

(Qtr1) Jan, Feb,Mar : (Qtr2) Apr, May, Jun : (Qtr3) Jul, Aug, Sep : (Qtr4) Oct, Nov, Dec

An example of the top-level query is as follows:

```
Select Product, Region, Market, "Year" from Excel_DS where Product =
'100-20' and
Region = 'East' and "Year" IN (Jan, Feb, Mar, Apr, May, Jun, Jul, Aug,
Sep, Oct, Nov, Dec)
```

Intermediate Level

When the report is executed with Quarter in the intersection, the actual generation name is Quarter, which is not mapped. The next generation is Month, which is mapped.

In the Year dimension for Selected Quarter Qtr1, get all children from Month generation:

(Qtr1) Jan, Feb, Mar

An example of the intermediate level query is as follows:

```
Select Product, Region, Market, "Year" from Excel_DS where Product =
'100-20' and
Region = 'East' and "Year" IN (Jan, Feb, Mar)
```

Mapped Level

When the report is executed with Month in the intersection, the actual generation name is Month, which is mapped in the Year dimension for Selected Month Jan.

An example of a mapped-level query is as follows:

```
Select Product, Region, Market, "Year" from Excel_DS where Product =
'100-20' and
Region = 'East' and "Year" IN (Jan)
OR
Select Product, Region, Market, "Year" from Excel_DS where Product =
'100-20' and
Region = 'East' and "Year" = 'Jan'
```

Map Dimension to a Data Source Column

When you map a dimension to a data source column, the report result has the same member as where the drill through is executed. When you have a flat hierarchy, you also map the dimension name directly to a data source column.

When you do this mapping, the generated query has a condition such as:

```
dsColName = <actual value from Smart View intersection>
```

Example

You mapped the dimension Scenario in the Sample Basic file to the Scenario data source column. In Smart View, when you're not zoomed in on Scenario, the filter condition is Scenario = Scenario.

If you're zoomed in on Scenario, the filter condition is either Scenario = Actual or Scenario = Budget.

This can be useful when the data source column contains data from all of its generations. For example, Time data source column also contains values with Year and Month. You can map the Time dimension directly to it and, according to the intersection, the condition can be added.

Map Level 0 to a Data Source Column

When creating a drill through report, you can map a level 0, for a particular dimension, to a data source column.

The screenshot shows the 'Drillthrough Report' configuration window. At the top, there are fields for 'Name' (weekly_sales) and 'Datasource' (sales_week), along with a 'Use Temporary Tables' checkbox. Below this is a 'Column Mapping' table with the following structure:

Column	Report Columns	Dimension/Generation (Filter Condition)
PRODUCT	✓	Product Product SKU [Generation]
MARKET	✓	Market Level0 [Level]
SCENARIO	✓	Scenario None
MONTH	✓	Year Months [Generation]

So, whenever you execute a report from Smart View, on any intersection with the Product dimension member, we fetch all leaf level members for that particular member and add them to the drill through query.

In the use case of a recursive drill through, we get the members from the mapped generation, however, we always get all leaf level members in the hierarchy. If the report is executed on the root member, the dimension member itself, the query contains all leaf members from the dimension.

Map Multiple Cells and Regions

This use case describes the ability to use multi-cells and multi-regions in drill through reports.

Prerequisites: Latest versions of Smart View and Essbase.

You can select multiple cells or ranges of cells, and see the merged results in drill through. Selections can be recursive, non-recursive, level 0, contiguous, or non-contiguous.

If you have existing single-cell drill through reports, no changes in your reports are necessary. They continue to work with single as well as multi-cell and multi-range selections.

Your drill through depends on column mapping, which generates the filter conditions and "where" clause in the data source query.

If you use generation (recursive) mapping, it includes all descendants of the selected members, for example, Qtr1 includes Jan, Feb, and Mar cells.

If you use Level 0 (ragged hierarchies) mapping, all leaf-level members of the selected member hierarchy are included.

After you create the connection and data source, you specify the report columns to view drillable cells or regions in Smart View, and mapping of data source columns to cube entities.

When you use multi-cell drill through, you select contiguous cells in Excel, such as B3, B4 in column B in the following example.

	A	B	C	D	E
1		Product	Market	Scenario	
2		Sales	COGS	Margin	Total Expenses
3	Qtr1	95820	42877	52943	28240
4	Qtr2	101679	45362	56317	29210
5	Qtr3	105215	47343	57872	29960
6	Qtr4	98141	43754	54387	28587
7	Year	400855	179336	221519	115997
8					

When you use multi-region drill through, you select multiple non-contiguous areas of multiple cells. In the example below, you select: B3+B4+B5, B7+B8+B9, and B11+B12+13.

Use **Ctrl + Select** to select multiple non-contiguous regions in Excel. The selection in the example gives you a detailed report for all of the months until September, excluding quarter totals.

You can select multi-ranges from any parent or hierarchy.

	A	B	C	D	E
1		Product	Market	Scenario	
2		Sales	COGS	Margin	Total Expenses
3	Jan	31538	14160	17378	9354
4	Feb	32069	14307	17762	9416
5	Mar	32213	14410	17803	9470
6	Qtr1	95820	42877	52943	28240
7	Apr	32917	14675	18242	9598
8	May	33674	15056	18618	9689
9	Jun	35088	15631	19457	9923
10	Qtr2	101679	45362	56317	29210
11	Jul	36134	16122	20012	10134
12	Aug	36008	16272	19736	10191
13	Sep	33073	14949	18124	9635
14	Qtr3	105215	47343	57872	29960
15	Qtr4	98141	43754	54387	28587
16	Year	400855	179336	221519	115997

After you select multi-cells or multi-regions, and then select **Drill Through** in the Essbase ribbon in Smart View, a drop-down list of available and relevant drill through reports are displayed for your selection. These existing reports are based on the intersections for the selected cells.

If you want sorted data for multi-cell drill through, you can define the data source query itself with sorting, or use Excel sorting after the report runs.

Multi-region drill through data remains unsorted. You can sort the results in Excel.

Create Drill Through Reports

The steps for creating drill through reports are to create the connection and data source and then define drillable regions.

Before you set up the report, create or import the Essbase cube.

1. [Create a Drill Through Connection and Data Source](#)
2. [Define Report Columns and Drillable Regions](#)

Create a Drill Through Connection and Data Source

From the Cube Designer, you need to create a connection to the data source file.

1. On the Cube Designer ribbon, click **Connections**. Ensure that you're connected to the correct Essbase Cloud Service URL. Click **Save**.

Your connection is saved on the Server area of the ribbon.

2. If you need to build a cube, rather than use an existing one, do the following:
 - a. On the Cube Designer ribbon, click **Build Cube**.
 - b. After you log in to Essbase as a Power User, build the cube using the **Create Cube** option.
 - c. Select the options to load data sheets, but not to run calculation sheets.
 - d. Click **View Jobs** to view the status of your build.
 - e. When your job completes, go to a web browser and log in as the same user. Navigate to Applications, and check that the application named DrillThrough was created with the relevant cube.
 - f. If you're using a CSV data source file, copy any accessory data source files to the drill through application's file catalog. For example, click **Files** and navigate to the CSV file. Navigate to All Files > Applications > Drillthrough > Basic, and click **Paste**.
3. Now define the connection and a data source file. For full details, see [Using Connections and Data Sources](#).
 - a. On the Sources page, click **Connections**, then **Create connection**, and **File**.
 - b. Enter a connection file name, and provide the path to the file you uploaded to the catalog.
 - c. Click **Test** to validate the connection, and if successful, click **Create**.
 - d. Now define the data source for the DrillThrough application. On the Sources page, click **Datasources**, and then **Create Datasource**.
 - e. Select the saved connection that you created.
 - f. Enter a name for the data source, optionally add a description, and click **Next**.
 - g. On the Columns page, where relevant, change the column types, add aliases, set parameters, if any, and then click **Next**.
 - h. Preview the tabular metrics, and when you're ready, click **Create** and then **Close**.

Define Report Columns and Drillable Regions

After defining the connection and data source, the next step to define the report.

1. On the Applications page, select the cube under your drill through application, click the Actions icon to the right, and click **Inspect**.
2. Select the **Scripts** page.
3. Select **Drill Through Reports**.
4. Click **Create**, and then select one of the following report types:
 - **Datasource** - to base the drill through report on the created data source target. Proceed to the next step.
 - **URL** - to drill through directly to a URL from a drill-through point in a spreadsheet. Skip the next step.
5. For a **Datasource** type drill through report:
 - a. Enter a name for the report.
 - b. Select the data source that you created earlier. The columns of the data source are displayed on the Column Mapping view.
 - c. Select the report columns that you want in the report, map them to dimensions, and designate the appropriate generation or level, or leave as None.
 - d. If you select the option **Use Temporary Tables** for a drill through report, all members in the IN statement are added to a temporary table created in the source database. This can improve query performance. The source database must have permission to create temporary tables for this option to be enabled.
 - e. Skip the following step.
6. For a **URL** type drill through report:
 - a. Enter a name for the report.
 - b. Enter the target URL. Your syntax must be consistent with the requirements of the target URL. If you want to express dimensions, columns, and values, you must use the following syntax: \$\$<dimension-name>-VALUE\$\$\$. For example, for a Market dimension, the syntax is: \$\$Market-VALUE\$\$\$.
7. Click **Drillable Regions** to define regions that drill through to the external Datasource or URL target. Click **+** if you want to add regions.
 - A drillable region can be a combination of Essbase members or member set functions where all conditions must be satisfied. You can have one or more drillable regions. The drill through report is shown if either one of these drillable region conditions is satisfied. For example, if the first drillable region is: Jan, Sales, the report is shown if both **Jan** and **Sales** are selected in SmartView. If you have a second drillable region: Feb, New York, the report is shown if both **Feb** and **New York** are selected. Since you have two regions here, the report is shown if you have either **Jan** and **Sales** OR **Feb** and **New York** selected.
 - You can use Essbase member-set calculation language for defining security filters. See [Member Set Functions](#) in the *Essbase Technical Reference*.
8. When finished, click **Save and Close**.

Execute Drill Through Reports

Now that you have set up an application and cube for drill through, and created a report, you are ready to execute the report and analyze data. But first, let's format the report.

- [Format Drill Through Reports](#)
- [Run Drill Through Reports](#)

Format Drill Through Reports

Let's set up Smart View to show drill through members and data cells in a different style.

1. In the workbook, on the Smart View ribbon, click **Options**.
2. Under Formatting, ensure that **Use Cell Styles** is selected.
3. Under Cell Styles:
 - a. Expand Essbase, and then Member Cells. Select **Member Drill-through**, then right-click it and choose a style (for example, a blue background).
 - b. Expand Data Cells, select **Drill-through**, then right-click it and choose the same style.

The report is now formatted and can be executed.

Run Drill Through Reports

After setting up the drill through report, you're ready to run it.

1. On the Cube Designer ribbon, click **Analyze** and **Connect Query Sheets**. If prompted, select **Reuse sheet contents and POV**. This connects you to your drill through cube, moves the workbook focus to the first query sheet, and selects the Essbase ribbon.

The drillable regions are displayed in the style you chose.

2. Drill through one of the cells to see the data source for the cell, for example, select a cell and click **Drill Through**.

In the new sheet, examine the drill through report. You have drilled through to the external data source to see the next level data. Select an entire column in the new sheet. In the lower right of Excel, notice the sum. This number matches the value of the cell you drilled through from.

3. Optionally, click the **Data** ribbon to filter data in the drill through report

A

Application Workbooks Reference

Oracle recommends that you download a sample application workbook and examine the worksheets to familiarize yourself on how to design your own application and cube.

- [Understand the Essbase.Cube Worksheet](#)
- [Understand the Cube.Settings Worksheet](#)
- [Understand the Cube.Generations Worksheet](#)
- [Understand the Cube.Textlists Worksheet](#)
- [Understand Dimension Worksheets](#)
- [Understand Data Worksheets](#)
- [Understand Calculation Worksheets](#)
- [Understand MDX Worksheets](#)

Also see [Download a Sample Application Workbook](#).

Understand the Essbase.Cube Worksheet

The Essbase.Cube worksheet defines the application and cube name and dimension information, such as dimension names, types, storage (dense or sparse) and outline order.

The following image shows the Essbase.Cube worksheet in a sample application workbook.

Application Name	Sample			
Database Name	Basic			
Version	1.0			
Dimension Definitions				
	Dimension Type	Storage Type	Outline Order	Base Dimension
Year	Time	Dense	1	
Measures	Accounts	Dense	2	
Product	Regular	Sparse	3	
Market	Regular	Sparse	4	
Scenario	Regular	Sparse	5	
Caffeinated	Attribute-Boolean		6	Product
Ounces	Attribute-Numeric		7	Product
Pkg Type	Attribute-Text		8	Product
Population	Attribute-Numeric		9	Market
Intro Date	Attribute-Date		10	Product

Table A-1 Essbase.Cube Worksheet Fields and Values

Property or Field	Valid Values	Description
Application Name	<ul style="list-style-type: none"> • The application name must not exceed 30 characters. • Do not use spaces. • Application names are not case-sensitive. • The following special characters are not allowed: % \$ - { } () ! ~ ` # & @ ^ 	Enter the name of the application.
Database Name	<ul style="list-style-type: none"> • The cube name must not exceed 30 characters. • Do not use spaces. • Cube names are not case-sensitive. • The following special characters are not allowed: % \$ - { } () ! ~ ` # & @ ^ 	Enter the name of the cube.
Version	This must be a positive integer.	This is the application workbook version.
Dimension Name	Dimension names cannot be the same as the cube name.	<p>Enter the name of each dimension. There must be at least two dimensions in a cube. For block storage, one dimension must be a dense dimension.</p> <p>Use no more than 1024 characters when naming dimensions, members, or aliases.</p> <p>The following special characters are not allowed: @, ., ,, !, {, }, [,], /, \, *,</p>
Dimension Type	<ul style="list-style-type: none"> • Time • Accounts • Regular • Attribute-Boolean • Attribute-Numeric • Attribute-Text • Attribute-Date 	Describes the type of dimension. Regular is the Default. Per cube, you can only use one Time and one Accounts dimension type.
Dimension Storage	<ul style="list-style-type: none"> • Dense • Sparse 	<p>Sparse is the default.</p> <p>There must be at least one dense dimension.</p>
Outline Order	This must be a positive integer.	<p>This is the order of the dimension in the outline.</p> <p>Attribute dimensions must be ordered after base dimensions.</p>

Table A-1 (Cont.) Essbase.Cube Worksheet Fields and Values

Property or Field	Valid Values	Description
Base Dimension	This must be an existing dimension name.	This is the dimension pairing for the attribute dimension.

You can modify the Essbase.Cube worksheet in the Designer Panel. See [Work with the Essbase.Cube Worksheet in Cube Designer](#).

Understand the Cube.Settings Worksheet

The Cube.Settings worksheet defines the application type (aggregate storage or block storage) and many cube and outline properties such as dynamic time series members and substitution variables.

Each of the five sections in the Cube.Settings worksheet has information about its fields and values, and how to modify those fields and values by using the Designer Panel.

- [Understand the Cube.Settings Worksheet: Alias Tables](#)
- [Understand the Cube.Settings Worksheet: Properties](#)
- [Understand the Cube.Settings Worksheet: Dynamic Time Series](#)
- [Understand the Cube.Settings Worksheet: Attribute Settings](#)
- [Understand the Cube.Settings Worksheet: Substitution Variables](#)

Understand the Cube.Settings Worksheet: Alias Tables

This section of the Cube Settings worksheet lists alias tables that need to be created for the cube.

It must contain at least the Default row.

Property or Field	Valid Values	Description
Default	Default	Every cube has a table named Default. You can create additional alias tables in the rows following the Default row.
Rows following the default row. These new rows can be created manually, or using the Designer Panel.	Naming conventions for member names apply. See Naming Conventions for Dimensions, Members, and Aliases.	You can set multiple aliases for a member using multiple alias tables.

You define alias table names on the Cube.Settings worksheet. You define the contents of the alias tables on the dimension worksheets.

See Setting Aliases.

Understand the Cube.Settings Worksheet: Properties

The following table shows the fields, values and descriptions for the Properties section on the Cube.Settings worksheet:

Table A-2 Properties Section of the Cube.Settings Worksheet

Property or Field	Valid Values	Description
Application Type	<ul style="list-style-type: none"> ASO BSO 	<p>This is an application property. Defines whether the cubes in the application use aggregate storage (ASO) or block storage (BSO).</p>
Outline Type	<ul style="list-style-type: none"> Unique Duplicate 	<p>This is a database property.</p> <ul style="list-style-type: none"> Unique: member names in the outline must be unique. Duplicate: Duplicate member names are permitted in the outline.
Aggregate missing values	<ul style="list-style-type: none"> Yes No 	<p>This is a database property. Defines whether missing (#MISSING) values are aggregated during a cube calculation.</p>
Create blocks on equations	<ul style="list-style-type: none"> Yes No 	<p>This is a database property. If you enter Yes, then when you assign a nonconstant value to a member combination for which no data block exists, a data block is created. Entering Yes can produce a very large cube. Sometimes, new blocks are not desired; for example, when they contain no other values. In large databases, creation and processing of unneeded blocks can increase processing time and storage requirements.</p> <p>For more specific control, you can use the SET CREATEBLOCKONEQ calculation command within a calculation script to control creation of blocks at the time the command is encountered in the script. See the SET CREATEBLOCKONEQ calculation command.</p>

Table A-2 (Cont.) Properties Section of the Cube.Settings Worksheet

Property or Field	Valid Values	Description
Two-Pass calculation	<ul style="list-style-type: none"> • Yes • No 	This is a database property. If you enter Yes, then after a default calculation, members that are tagged as two-pass are recalculated, overwriting the aggregation results from the first calculation pass. The two-pass tag is effective on members of the dimension tagged as Accounts and on Dynamic Calc and Dynamic Calc and Store members of any dimension.
Date Format	<p>There are many valid date formats. These are some examples:</p> <ul style="list-style-type: none"> • mm dd yyyy • dd mm yy • mm/dd/yy • mm-dd-yyyy 	This is a database property. You can set the format of member names in date attribute dimensions. If you change the date format, then you must rebuild the date attribute dimensions and reassociate dimension members.
Scenario Sandboxes	<ul style="list-style-type: none"> • 0 • A positive integer less than 1000. 	This value defines whether the cube contains a sandbox dimension for creating scenarios of the data, and the number of sandbox members within the sandbox dimension. A value of 0 indicates no sandbox dimension.

You can modify the Properties section on the Cube.Settings worksheet in the Designer Panel. See [Work with the Cube.Settings Worksheet: Properties in Cube Designer](#).

Understand the Cube.Settings Worksheet: Dynamic Time Series

Table A-3 Dynamic Time Series Section of the Cube.Settings Worksheet

Property or Field	Valid Values	Description
H-T-D	Integer value representing the generation number	History to date
Y-T-D	Integer value representing the generation number	Year to date
S-T-D	Integer value representing the generation number	Season to date
P-T-D	Integer value representing the generation number	Period to date
Q-T-D	Integer value representing the generation number	Quarter to date

Table A-3 (Cont.) Dynamic Time Series Section of the Cube.Settings Worksheet

Property or Field	Valid Values	Description
M-T-D	Integer value representing the generation number	Month to date
W-T-D	Integer value representing the generation number	Week to date
D-T-D	Integer value representing the generation number	Day to date

You can modify the Dynamic Time Series section on the Cube.Settings worksheet in the Designer Panel. See [Work with the Cube.Settings Worksheet: Dynamic Time Series in Cube Designer](#).

See Using Dynamic Time Series Members.

Understand the Cube.Settings Worksheet: Attribute Settings

The following table shows the fields, values and descriptions for the Attribute Settings section on the Cube.Setting worksheet:

Table A-4 Attribute Settings

Property or Field	Valid Values	Description
Dimension Name	Default: Attributes Calculation	To avoid duplicating names in an outline, you can change the names of members of the attribute calculations dimension. Regardless of the name that you use for a member, the function of the member remains the same. For example, the Sum member always calculates a sum, no matter what you name it. See Changing the Member Names of the Attribute Calculations Dimension.
Sum Member	Default: Sum	This is a member of the attribute calculations dimension. The name to use when requesting sum data.
Count Member	Default: Count	This is a member of the attribute calculations dimension. The name to use when requesting count data.

Table A-4 (Cont.) Attribute Settings

Property or Field	Valid Values	Description
Minimum Member	Default: Min	This is a member of the attribute calculations dimension. The name to use when requesting minimum data.
Maximum Member	Default: Max	This is a member of the attribute calculations dimension. The name to use when requesting maximum data.
Average Member	Default: Avg	This is a member of the attribute calculations dimension. The name to use when requesting average data.
False Member	Default: False	The initial Boolean member names in a cube are set as True and False. See Setting Boolean Attribute Member Names.
True Member	Default: True	The initial Boolean member names in a cube are set as True and False. See Setting Boolean Attribute Member Names.
Prefix/Suffix Value	<ul style="list-style-type: none"> • None • Dimension • Parent • Grandparent • Ancestors 	See Setting Prefix and Suffix Formats for Member Names of Attribute Dimensions.
Prefix/Suffix Format	<ul style="list-style-type: none"> • Prefix • Suffix 	You can define unique names by attaching a prefix or suffix to member names in Boolean, date, and numeric attribute dimensions in the outline. See Setting Prefix and Suffix Formats for Member Names of Attribute Dimensions.
Prefix/Suffix Separator	<ul style="list-style-type: none"> • _ Underscore • Pipe • ^ Carat 	You can define unique names by attaching a prefix or suffix to member names in Boolean, date, and numeric attribute dimensions in the outline. Select a separator (to place between the prefix or suffix and the original name): underscore (_), pipe (), or caret (^).

Table A-4 (Cont.) Attribute Settings

Property or Field	Valid Values	Description
Attribute Numeric Ranges	<ul style="list-style-type: none"> • Tops of ranges • Bottoms of ranges 	See Setting Up Member Names Representing Ranges of Values.
Date Member	<ul style="list-style-type: none"> • Month First (mm-dd-yyyy) • Day First (dd-mm-yyyy) 	You can change the format of members of date attribute dimensions. See Changing the Member Names in Date Attribute Dimensions.

You can modify the Attribute Settings section on the Cube.Settings worksheet in the Designer Panel. See [Work with the Cube.Settings Worksheet: Attribute Settings in Cube Designer](#).

Understand the Cube.Settings Worksheet: Substitution Variables

Substitution variables act as global placeholders for information that changes regularly. You create the variable and a corresponding string value, and the value can then be changed at any time.

A substitution variable can be used in a query or calculation script to represent a member in the outline. By default, there are no substitution variables defined for a cube.

There is not an option to add substitution variables in the Designer Panel, however you can add them directly in the application workbook.

1. On the Cube.Settings worksheet, in the Substitution Variables section, create a new row.
2. Enter the variable name in column A and its value in column B, enclosing the value in quotation marks if it represents a member name.
Example:

```
CurrMonth "Jan"
```

See Using Substitution Variables.

Understand the Cube.Generations Worksheet

Cube.Generations Worksheets

The Cube.Generations worksheet is used for naming generations in an outline.

The term "generation" indicates the distance of a member from the root of the dimension. Using a generation number, you can determine the location of members within the database tree. All members in a database that are the same number of branches from their root have the same generation number. The dimension is generation 1, its children are generation 2, and so on.

You can create names for generations in an outline, such as a word or phrase that describes the generation. For example, you might create a generation name called Cities for all cities in the outline.

You can also use generation names in calculation scripts wherever you need to specify a list of generation numbers. For example, you could limit a calculation in a calculation script to all members in a specific generation.

You can specify only one name per generation. The specified name must be unique; that is, it cannot duplicate a generation, level, or member name or an alias or conventional alias.

If you build a cube using an application workbook that has names reserved for Dynamic Time Series on the Cube.Generations sheet for the time dimension, Essbase automatically creates and enables the corresponding Dynamic Time Series member.

 **Note:**

The Dimension section of the Cube.Generations worksheet changes if you change the dimension worksheet (Dim.*dimname*) by adding or deleting members in such a way that the number of generations in the dimension is changed. If you make changes to the dimension worksheet by adding or deleting members, you should always press the **Update Generation Worksheet** button on the **Dimensions** tab of the Designer Panel as part of the editing process.

Cube.Generations Worksheet Format

The following image shows a Cube.Generations worksheet in a sample application workbook.

Generation Properties

Dimension Name Year

Generation Number	Generation Name	Unique
1	History	Yes
2	Quarter	Yes
3		Yes

Dimension Name Product

Generation Number	Generation Name	Unique
2	Category	Yes
3	Line	No

Dimension Name Market

Generation Number	Generation Name	Unique
1	Market1	Yes
2	m2	No
3	m3	No

Table A-5 Fields and Valid Values in Generation Worksheets

Property or Field	Valid Values	Description
Dimension Name	For dimension naming restrictions, see Naming Conventions for Dimensions, Members, and Aliases for naming restrictions.	The dimension name.
Generation Number	A generation number, 1 or greater.	A root branch of the tree is generation 1. Generation numbers increase as you count from the root toward the leaf member.
Generation Name	You can define only one name for each generation. When you name generations, follow the same naming rules as for members. See Naming Conventions for Dimensions, Members, and Aliases.	The generation name. You can use this field to create or change generation names. Enter the generation name and then build or update the cube using the application workbook. See Update Cubes Incrementally in Cube Designer .

Table A-5 (Cont.) Fields and Valid Values in Generation Worksheets

Property or Field	Valid Values	Description
Unique	<ul style="list-style-type: none"> Yes No 	For duplicate member name outlines, enter Yes to require unique member names within the associated generation.

Understand the Cube.Textlists Worksheet

In application workbooks, the Cube.Textlists worksheet defines text lists. Text lists are used to work with text measures, which extend the analytical capabilities of Essbase.

In addition to numeric values, measures can be associated with text-typed values. Storage and analysis of textual content can be useful when a cell needs to have one of a finite list of textual values; for example, a product may be sold in 5 different colors. The color is a text measure whose value must be one of the 5 colors. The colors are a set of text strings mapped to corresponding numeric IDs. These mappings are contained in tables in the Cube.Textlists worksheet.

You can add multiple text list tables to the same sheet and they can be associated with multiple measures.

The following image shows the Cube.Textlists worksheet in a sample application workbook.

Text List Properties		
List Name	sample text list	
Associated Members	[replace with member name...]	[replace with another member name...]
ID	Text	
#Missing	Blank	
#OutOfRange	N/A	
[replace with integer value]	[replace with string value]	
[replace with integer value]	[replace with string value]	

Table A-6 Cube.Textlists Worksheet Fields and Values

Property or Field	Valid Values	Description
List Name	Must not exceed 80 characters.	A text list must start with a list name followed by its value in the adjacent cell.
Associated Members	Existing member names.	Member names added in adjacent cells. Multiple members can be added in adjacent cells to the right.

Table A-6 (Cont.) Cube.Textlists Worksheet Fields and Values

Property or Field	Valid Values	Description
ID	The first two values under ID are #Missing and #OutOfRange. These two values must exist in every text list table. The other IDs must be integers.	Each ID, including the #Missing, #OUTOFRANGE and numeric values, must map to a text value. The first two IDs, #Missing and #OUTOFRANGE, are for handling cases where the textual data is invalid or empty. For example, if you try to load an unmapped value such as "Average" to a text measure, the cell value would not be updated, and would display as #Missing in a subsequent query. If you load a numeric cell value that is unmapped, the subsequent query would return N/A.
Text	Up to 80 characters.	The text column contains the text values for each text measure. Each text value must map to an integer in the ID column. Any text value that does not map to an integer in the text list is considered by Essbase to be invalid.

See:

- Working with Typed Measures
- Performing Database Operations on Text and Date Measures

Understand Dimension Worksheets

Application workbooks contain one dimension worksheet for each of the dimensions listed in the Essbase.Cube worksheet. The name of each dimension worksheet is *Dim.dimname*; for example, the Year dimension worksheet is called *Dim.Year*. Dimension names can contain up to 1024 characters, but long dimension names (longer than 31 characters, including "Dim.") are truncated in the dimension sheet name.

Dimension worksheets use load rule syntax. For example, an X in the Storage column means that the data value is not stored.

The following image shows a dimension worksheet in a sample application workbook.

Dimension Name	Year					
Definitions						
File Name	Dim_Year		Delimiter	,		
Rule Name	Year		Header Rows to Skip	0		
Build Method	PARENT-CHILD		Allow Moves	No		
Incremental Mode	Merge					
Members						
Columns	PARENT	CHILD	STORAGE	ALIAS.ChineseNames	IGNORE	ALIAS.JapaneseNames
		Year	X	年		1 年
	Year	Qtr1	X	第一季		2 第一四半期
		Qtr1		一月		3 1月
		Qtr1		二月		4 2月
		Qtr1		三月		5 3月
	Year	Qtr2	X	第二季		6 第二四半期
		Qtr2		四月		7 4月
		Qtr2		五月		8 5月
		Qtr2		六月		9 6月
	Year	Qtr3	X	第三季		10 第三四半期
		Qtr3		七月		11 7月
		Qtr3		八月		12 8月
		Qtr3		九月		13 9月

Table A-7 Fields and Valid Values in Dimension Worksheets

Property or Field	Valid Values	Description
Dimension Name	The name of the dimension. Do not change the dimension name in this field.	Any dimension or attribute dimension in the outline. Defined on the Essbase.Cube worksheet. Use no more than 1024 characters when naming dimensions, members, or aliases. The following special characters are not allowed: @, ,, ,!, {, }, [,], /, \, *
File Name	A valid string. The file name cannot be longer than thirty characters.	The build process creates a data file with a .txt extension in Essbase for every data worksheet in the application workbook. You can give them meaningful names so that they are easily recognizable if they need to be used again.
Rule Name	A valid string. See Name and Related Artifact Limits. The rule name cannot be longer than thirty characters.	The build process creates a rule file with a .rul extension in Essbase for every dimension worksheet in the workbook. You can give them meaningful names so that they are easily recognizable if they need to be used again.

Table A-7 (Cont.) Fields and Valid Values in Dimension Worksheets

Property or Field	Valid Values	Description
Build Method	<ul style="list-style-type: none"> PARENT-CHILD GENERATION 	In Designer Panel, you can build a cube with either build method, but you cannot edit a cube built using the Generation build method using the panel, and you cannot view hierarchies using Cube Designer Dimension Hierarchy viewer.
Incremental Mode	<ul style="list-style-type: none"> Remove Unspecified Merge 	<p>Incremental dimension builds enable you to update existing dimensions with new members.</p> <p>Merge is the default. This option adds the new members to the dimension while retaining the existing members.</p> <p>Remove Unspecified removes members that are not specified in the source file.</p>
Delimiter	The values can be a tab, a space, or any single character except “.	This value must be updated directly in the Excel sheet. It cannot be updated using the Cube Designer interface.
Header Rows to Skip	A positive number or zero. Zero is the default.	<p>The number of header rows to skip when performing a data load or dimension build.</p> <p>This value must be updated directly in the Excel sheet. It cannot be updated using the Cube Designer interface.</p>
Allow Moves	<ul style="list-style-type: none"> Yes No 	<p>Within a dimension, moves members and their children to new parents; recognizes primary members and matches them with the data source; not available for duplicate member outlines.</p> <p>This value must be updated directly in the Excel sheet. It cannot be updated using the Cube Designer interface.</p>
Data Source	A valid Data Source name.	This value is used to retrieve data from the source defined in the data source definition. This value must be updated directly in the application workbook. It can't be updated using the Cube Designer interface.

Table A-7 (Cont.) Fields and Valid Values in Dimension Worksheets

Property or Field	Valid Values	Description
Member ID	Any unique key	Used to uniquely identify a member in an outline. Required for duplicate outlines.
Storage Type	<ul style="list-style-type: none"> • N Never allow data sharing. • O Tag as label only (store no data). • S Set member as stored (non dynamic calc and not label only). • X Create as dynamic calc. 	Uses load rules member property codes. See Using the Data Source to Work with Member Properties.
Consolidation Operator	<ul style="list-style-type: none"> • + • - • * • / • % • ~ • ^ 	<ul style="list-style-type: none"> • + (add) • - (subtract) • * (multiply) • / (divide) • % (percent) • ~ (no operation) • ^ (never consolidate)
IGNORE	Ignore	Data in a column with the heading, IGNORE is ignored during data loads and dimension builds. This value must be updated directly in the Excel sheet. It cannot be updated using the Cube Designer interface.
Two-Pass Calculation	<ul style="list-style-type: none"> • Yes • No 	If you enter Yes, after a default calculation, then members that are tagged as two-pass are recalculated. The two-pass tag is effective on members of the dimension tagged as Accounts and on Dynamic Calc and Dynamic Calc and Store members of any dimension. Two-pass calculation applies only to block storage outlines.

Table A-7 (Cont.) Fields and Valid Values in Dimension Worksheets

Property or Field	Valid Values	Description
Solve Order	Any number, 0 to 127	<p>You can set solve order for dimensions or members, or you can use the default solve order. The minimum solve order you can set is 0, and the maximum is 127. A higher solve order means the member is calculated later; for example, a member with a solve order of 1 is solved before a member with a solve order of 2.</p> <p>Members that are not assigned a solve order are assigned the solve order of their dimension.</p>
Time Balance	<ul style="list-style-type: none"> • A Treat as an average time balance item (Applies to accounts dimensions only). • F Treat as the first time balance item (Applies to accounts dimensions only). • L Treat as the last time balance item (Applies to accounts dimensions only). 	<p>Uses load rules member property codes. See Using the Data Source to Work with Member Properties.</p> <p>Time balance properties provide instructions about how to calculate data in the Accounts dimension. See Setting Time Balance Properties.</p>
Skip Value	<ul style="list-style-type: none"> • B Exclude data values of zero or #MISSING in the time balance (applies to accounts dimensions only). • M Exclude data values of #MISSING from the time balance (applies to accounts dimensions only). • Z Exclude data values of zero from the time balance (applies to accounts dimensions only). 	<p>Uses load rules member property codes. See Using the Data Source to Work with Member Properties.</p> <p>If you set the time balance as first, last, or average, then set the Skip property to indicate what to do when missing values or values of 0 are encountered. See Setting Skip Properties.</p>
Expense Reporting	E	Treat as an expense item (applies to accounts dimensions only)

Table A-7 (Cont.) Fields and Valid Values in Dimension Worksheets

Property or Field	Valid Values	Description
Comment	Any string	Enter a comment.
Formula	Valid calculation syntax	Enter a member formula.
User Defined Attribute	Attribute names, such as specific colors or sizes	Defined attribute names used to aid in the analysis of the data. When making changes to user-defined attributes (UDAs) while updating a cube incrementally using Cube Designer and an application workbook, you must specify all the UDAs in the dimension sheet, both new ones you are adding and existing UDAs in the outline. If you specify some UDAs (such as those you are adding), but not all of them, those that are not specified are deleted.
Number of UDAs	A numeral	The number of UDAs for this member.
Available Alias Tables	Naming conventions for member names apply. See Naming Conventions for Dimensions, Members, and Aliases.	ALIAS. <i>table_name</i> After the column heading with ALIAS. <i>table_name</i> , the column is populated with the aliases for the cube.

You can modify dimension worksheets in the Designer Panel. See [Work with Dimension Worksheets in Cube Designer](#).

See Working with Rules Files.

Understand Data Worksheets

Data Worksheets

You can include one or more data worksheets in an application workbook. The name of each data worksheet is Data.*name*. For example, for values for the eastern region, the data worksheet might be called Data.East. The *name* can be anything you choose. You can choose meaningful names so that you can recognize them if you need to use them again.

Note:

Multiple data worksheets are allowed in an application workbook, but they must share the exact same column layout.

Data Worksheet Format

When loading data, a member from every dimension must be defined before a data value. Therefore, the data worksheet places all but one dimension under the column headings titled, *Dimension.dimension_name*. One dimension is selected as the Measures dimension and members from that dimension must be added manually under the remaining column headings titled *Measure.member_name*. Only place members that will contain data in the columns titled *Measure.member_name*.

When scenarios are enabled, cubes have a hidden dimension called sandbox. The sandbox dimension, named *Dimension.sandbox*, is the first column in the data worksheet. It contains a member called base that you must define when loading data.

The following image shows a data worksheet in a sample application workbook.

Definitions							
File Name	Cube_Basic		Sign Flip Dimension	Measures			
Rule Name	Basic		Sign Flip UDA	Flip			
Data Load Option	Replace						
Delimiter	,						
Header Rows to Skip	0						

Data							
Columns	Dimension.Product	Dimension.Market	Dimension.Year	Dimension.Scenario	IGNORE	Measure.Sales	Measure.COGS
	100-10	New York	Jan	Actual		1 678	271
	100-10	New York	Feb	Actual		2 645	258
	100-10	New York	Mar	Actual		3 675	270
	100-10	New York	Apr	Actual		4 712	284
	100-10	New York	May	Actual		5 756	302
	100-10	New York	Jun	Actual		6 890	356
	100-10	New York	Jul	Actual		7 912	364
	100-10	New York	Aug	Actual		8 910	364
	100-10	New York	Sep	Actual		9 790	316
	100-10	New York	Oct	Actual		10 650	260
	100-10	New York	Nov	Actual		11 623	249
	100-10	New York	Dec	Actual		12 699	279
	100-10	New York	Jan	Budget		13 640	260

The following table describes the settings on the *data.name* worksheets in application workbooks.

Property or Field	Valid Values	Description
File Name	A valid string. See Name and Related Artifact Limits.	The build process creates a data file with a .txt extension in the Essbase web interface for every data worksheet in the application workbook. You can give them meaningful names so that they are easily recognizable if they need to be used again.
Rule Name	A valid string. See Name and Related Artifact Limits.	The build process creates a rule file with a .rul extension in the Essbase web interface for every dimension worksheet in the workbook. You can give them meaningful names so that they are easily recognizable if they need to be used again.

Property or Field	Valid Values	Description
Data Load Option	<ul style="list-style-type: none"> Add Subtract Replace 	<p>If you enter Replace, then the existing values of the database are overwritten with the values of the data source.</p> <p>You can also use incoming data values to add to or subtract from existing database values. For example, if you load weekly values, then you can add them to create monthly values in the database.</p>
Delimiter	<p>The values can be a tab, a space, or any single character except “.</p> <ul style="list-style-type: none"> Tab Space Any single character except “ 	<p>This value must be updated directly in the Excel sheet. It cannot be updated using the Cube Designer interface.</p>
Header Rows to Skip	A positive number or zero.	<p>The number of header rows to skip when performing a data load or dimension build.</p> <p>This value must be updated directly in the Excel sheet. It cannot be updated using the Cube Designer interface.</p>
Sign Flip Dimension	<i>Dimension name</i>	<p>Reverses the values of data fields by flipping their signs.</p> <p>Enter the name of the dimension in the Sign Flip Dimension field, and enter the selected UDA within the specified dimension in the Sign Flip UDA field.</p> <p>This value must be updated directly in the Excel sheet. It cannot be updated using the Cube Designer interface.</p>
Sign Flip UDA	<ul style="list-style-type: none"> Flip Blank 	<p>Reverses the values of data fields by flipping their signs.</p> <p>Enter the name of the dimension in the Sign Flip Dimension field, and enter the selected UDA within the specified dimension in the Sign Flip UDA field.</p> <p>This value must be updated directly in the Excel sheet. It cannot be updated using the Cube Designer interface.</p>

Property or Field	Valid Values	Description
Ignore column header	Ignore	Data in a column with the heading, IGNORE is ignored during data loads and dimension builds. This value must be updated directly in the Excel sheet. It cannot be updated using the Cube Designer interface.
Data Source	A valid Data Source name.	This value is used to retrieve data from the source defined in the Data Source definition. This value must be updated directly in the application workbook. It can't be updated using the Cube Designer interface.

Data Operations

When you load data, values can replace, add to, or subtract from existing data values in the cube. You indicate which of these options to use in the **Data Load Option** field on the data worksheet.

- **Replace:** Overwrites cube values with the data source values. Replace is the default.
- **Add:** Adds data source values to the cube values. For example, if you load weekly data values, you can add them to create cumulative data values in the cube.
- **Subtract:** Subtracts data source values from the database values. For example, to track available budget by week, you can subtract weekly data expenditures from the previous week's budget values.

Rule Files

When you build a cube, data files and data load rule files are created in the Essbase web interface. Those files can then be used later if you want to load data to a cube. Data files are named with the file name specified in the definitions area of the data sheet and a .txt extension. For example, cube_basic.txt. Rule files are named with the file name specified in the definitions area of the data sheet and a .rul extension. For example, cube_basic.rul.

You can modify data worksheets in the Designer Panel. See [Work with Data Worksheets in Cube Designer](#).

See Data Sources.

Understand Calculation Worksheets

You can have one or more calculation worksheets in an application workbook.

The following image shows a calculation worksheet in a sample application workbook.

Definitions

File Name	CalcAll
Execute Calc	Yes

Script

```
SET UPDATECALC OFF;
SET CACHE HIGH;
SET MSG SUMMARY;
```

```
CALC ALL;
```

Within the calculation worksheet, the calculation script begins in cell C6.

The name of each calculation worksheet is Calc.*scriptname*, for example, for the sample CalcAll calculation script, the calculation worksheet is called Calc.calcall.

The contents of the calculation worksheet are used to create a calculation script in Essbase. The calculation script uses the file name specified in the definitions area of the calculation sheet and has a .csc extension. For example, *filename.csc*.

You can execute the calculation script when you build the cube in Cube Designer, if you select **Run Calculation Sheets Contained within Workbook** in the Build Cube dialog box. If you do not want to execute the calculation, do not select this option.

The calculation scripts are executed in the order they appear in the application workbook.

Property or Field	Valid Values	Description
File Name	A valid calculation script file name. <i>filename.csc</i> .	The File Name defines the calculation script name. The calculation script created in Essbase when the cube is created is the File Name with a .csc extension.
Execute Calc	<ul style="list-style-type: none"> • Yes • No 	If you enter Yes, then the calculation is executed at the time you build the cube. If you enter No, then the calculation is not executed right away. In either case, each calculation worksheet creates a calculation script in Essbase, using the specified file name with a .csc extension. That way, any of the calculations can be executed at a later time.

You can modify calculation worksheets in the Designer Panel. See [Work with Calculation Worksheets in Cube Designer](#).

Understand MDX Worksheets

You can have one or more MDX Insert worksheets in an application workbook. With these worksheets, you can create corresponding MDX files in the cube, and you can optionally execute the MDX at the time you build the cube.

- To execute the MDX when you build the cube, indicate **Yes** in the **Execute MDX** field on the MDX worksheet in the application workbook.
- To execute the MDX after the cube is created, run the MDX script from the Essbase web interface, from **Jobs**.

The following image shows an MDX Insert worksheet in a sample application workbook.

	A	B	C
1	Definitions		
2	File Name	mdxTest1	
3	Execute MDX	Yes	
4			
5	Script		
6	EXPORT INTO FILE "sample3"		
7	SELECT {[Mar],[Apr]} ON COLUMNS,		
8	Non Empty Crossjoin({&States} , crossjoin({[Actual],[Budget]},		
9	{[Opening Inventory],[Ending Inventory]})) ON ROWS		
10	FROM [Sample].[Basic]		

The name of each MDX worksheet is MDX.*scriptname*, for example, for the mdxTest1 MDX script, the MDX worksheet is called MDX.mdxTest1.

The contents of the MDX worksheet are used to create an MDX Insert script in the cube. The MDX script uses the file name specified in the definitions area of the MDX sheet and has an .mdx extension. For example, *filename.mdx*.

Property or Field	Valid Values	Description
File Name	A valid MDX script file name.	The File Name field defines the MDX script name. The MDX script is created in Essbase when the cube is created. The script name in Essbase is the file name with an .mdx extension.

Property or Field	Valid Values	Description
Execute MDX	<ul style="list-style-type: none"> • Yes • No 	<p>If you enter Yes, then the MDX script is executed at the time you build the cube. If you enter No, then the MDX script is not executed right away. In either case, each MDX worksheet creates an MDX script in Essbase, using the specified file name with a .mdx extension. That way, any of the MDX scripts can be executed at a later time.</p>

You can create and delete MDX worksheets in the Designer Panel. See [Work with MDX Worksheets in Cube Designer](#).

To learn more about MDX Insert, see [Insert and Export Data with MDX](#) and MDX Insert Specification.

B

Set up Cube Designer

You might find it easier to work with application workbooks in Excel using the Cube Designer extension for Smart View.

- [Workflow to Set up Cube Designer](#)
- [Download and Run the Smart View Installer](#)
- [Connect to Essbase](#)
- [Install the Smart View Cube Designer Extension](#)
- [Update the Smart View Cube Designer Extension](#)
- [Delete Smart View Connection URLs](#)

Workflow to Set up Cube Designer

This is the workflow for setting the Smart View Cube Designer extension:

1. Install Smart View.
2. Set up a data source connection to Essbase.
3. Install Cube Designer Smart View extension.
4. Update Cube Designer Smart View extension.

Download and Run the Smart View Installer

Smart View Prerequisites

- The latest release of Smart View
- Microsoft Office 2010, 2013 or 2016
- .NET Framework 4.0

On the [Oracle Technology Network Downloads](#) tab, the latest release for Smart View is always certified.



Note:

You must use .NET Framework 4.5 if you are installing Smart View from Essbase without saving the installer locally.

Installing Smart View

1. Log into Essbase.
2. Click **Console**.

3. On the **Desktop Tools** tab, click the Browse icon to the right of **Smart View for Essbase**.
4. On the Smart View download page on Oracle Technology Network, click **Accept License Agreement**, and then click **Download Now**.
If the Oracle sign-in page is displayed, then sign in with your Oracle user name (usually your email address) and password.
5. Follow the steps for your browser to download the .zip file, and save it to a folder on your computer.
6. Go to the folder that you used in Step 5, and then double click `smartview.exe` to start the installation wizard.
7. Select a destination folder for Smart View, and then click **OK**. For new installations, Smart View is installed by default in: `C:\Oracle\smartview`.
If you are upgrading an installation of Smart View, then the installer defaults to the folder where you previously installed Smart View.
8. When the installation is complete, click **OK**.

Continue the setup process with [Connect to Essbase](#).

Connect to Essbase

After you install Smart View, you can create connections to Essbase.

Connections require information about the server and port. Your Essbase administrator should provide you with the information you need to create the connection.

See [Connect to a Cube in Smart View](#).

Continue the setup process with [Install the Smart View Cube Designer Extension](#).

Install the Smart View Cube Designer Extension

Before you perform this procedure, you must complete the steps in [Connect to Essbase](#).

Installing Cube Designer from Smart View

1. On the Smart View ribbon, select **Options**, and then **Extensions**.
2. Click the **Check for updates** link.
Smart View checks for all extensions that your administrator has made available to you.
3. Locate the extension named **Oracle Cube Designer** and click **Install** to start the installer.
4. Follow the prompts to install the extension.

Installing Cube Designer from Essbase

1. In Essbase, click **Console**.
2. On the Desktop Tools tab, to the right of **Cube Designer Extension**, click **Download**.

3. In the **Opening CubeDesignerInstaller.svext** dialog box, select **Save File** and click **OK**.
Save the file to a local directory.
4. Close all Microsoft Office applications and make sure Microsoft Office applications are not running in the background.
5. Double click the CubeDesignerInstaller.svext file.
6. Restart Microsoft Office applications.

Update the Smart View Cube Designer Extension

If an extension is available for you to update, you can update it from Smart View Excel, on the **Extensions** tab of the Options dialog box.

To check for Cube Designer Smart View extension updates and install them:

1. From the Smart View ribbon, select **Options** and then **Extensions**.
2. Click the **Check for Updates, New Installs, and Uninstalls** link to check for updates.

You are prompted to log in.


If an update is available, the **Update Available** icon is displayed in the **Cube Designer** row.

Note:

This process uses a server locations list, which was created by previous Smart View connections . If there are connection definitions that are no longer valid, you receive errors when the process tries to connect to those servers. See [Delete Smart View Connection URLs](#).

3. Click **Remove** to uninstall the extension.
4. Close Excel.
5. Restart Excel.
6. From the Smart View ribbon, select **Options** and then **Extensions**.
7. Click **Check for Updates, New Installs, and Uninstalls**.

You are prompted to log in.

8. In the Cube Designer row, click **Install**. .
9. Close Excel.
10. Open Excel.
11. Ensure that the Cube Designer ribbon is displayed in Excel.



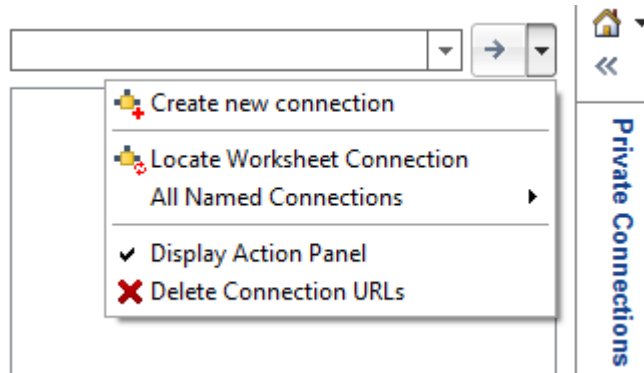
Delete Smart View Connection URLs

When you connect to Essbase from Cube Designer, the list of server locations that are used to connecting is created by previous Smart View connections . If there are connection definitions that are no longer valid, you receive errors.

You can reset the list of connection definitions to remove those that you are unwanted, or are invalid.

To reset the list of server locations:

1. Click the down arrow next to the **Private Connection** drop down list and select **Delete Connection URLs**.



2. In the Delete Connection URLs dialog box, select **Extension Update URLs** from the drop down menu.
3. Select all of the URLs except the one you want to use, and click **Delete**.