Oracle® Essbase

Scripting Reference for Oracle Essbase

F17647-06
September 2020

Oracle Essbase Scripting Reference for Oracle Essbase,

F17647-06

Primary Author: Essbase Information Development Team

Contents

3 MaxL

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

Scripting Reference Overview

You can use MaxL and the Essbase Command Line Interface (CLI) to perform operations in Essbase using scripts and shell-interfaces. This reference is intended for advanced users who need detailed information and examples.

MaxL is a language interface for administering Essbase, and the CLI is a command interface. MaxL statements begin with a verb, and enable you to perform actions on Essbase artifacts. CLI commands have a variety of options to help you specify what you want to do.

About the Scripting Reference

This Scripting Reference describes language and command interfaces available for Oracle Essbase. This reference is intended for advanced users who need detailed information and examples to perform operations on Essbase.

This document provides examples based mostly on the Sample Basic cube, provided with Essbase as a template you can build into a cube. The Sample application, as well as more samples you can build, are available in the Applications > Demo Samples section of the gallery. The gallery is available in the Files section of Essbase. See Explore the Gallery Templates.

To use this document, you need the following:

- A working knowledge of the operating system.
- An understanding of Essbase concepts and features.
- An understanding of the typical database administration requirements and tasks, including calculation, querying, security, and maintenance.

About Aggregate Storage Cubes

Consider using the aggregate storage model if the following is true for your database:

- The database is sparse and has many dimensions, and/or the dimensions have many levels of members.
- The database is used primarily for read-only purposes, with few or no data updates.
- The outline contains no formulas except in the dimension tagged as Accounts.
- Calculation of the database is frequent, is based mainly on summation of the data, and does not rely on calculation scripts.

Some MaxL grammar is applicable to aggregate storage mode, and some MaxL grammar is not relevant. To learn which statements are supported in aggregate

storage application and database operations, see MaxL Statements (Aggregate Storage).

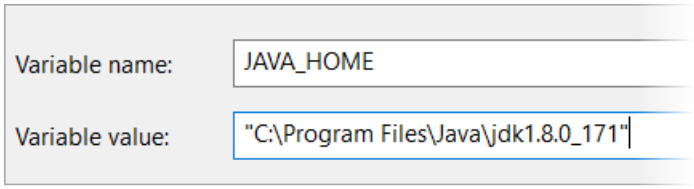
2 Essbase Command-Line Interface (CLI)

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.



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
-verbose	-v	Show extended descriptions
-essbaseurl	-url	Address of an instance of Essbase
-user	-u	User name
-password	-p	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 `.csc` 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: • 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: • 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 example:

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

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:
		
		`jdbc:oracle:thin:@host:port:SID`
		
		or with service name, as follows
		
		`jdbc:oracle:thin:@host:port/service_name`
		
		See Examples.

-user	-u	User name
-driver	-D	JDBC driver. If not provided, Oracle Database is considered the default, as `oracle.jdbc.driver.OracleDriver`
-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 `createlocalconnection` 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 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: • `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: • `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 `createlocalconnection` 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: • 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: • 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 threadscount] [-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. Minimum: 10
-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:
		
		`# ------------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`
		
		To control import order, rearrange the `import` entries in the text file.
		If –overwrite is used, the import operation deletes and recreates the entire application, importing only the artifacts present in the list. If –overwrite is not used, the import operation includes the artifacts specified in the list, without impacting any other artifacts already present in the target application.

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: • `.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**.
		• gzip—Default if compression is used. Provides smaller data transfer with slower calculation.
		• lz4—Provides faster calculation with a slower data transfer.
		Usage examples:
		`-ca gzip`
		`-ca lz4`

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
```

3
MaxL

Using MaxL, you can administer and query Essbase through a scripting language.

MaxL is the multi-dimensional database access language for Essbase. MaxL is a practical, expressive interface for administering and querying the Essbase system. With the MaxL language, you use statements to make requests. MaxL statements begin with a verb, such as Create, Alter, and Display.

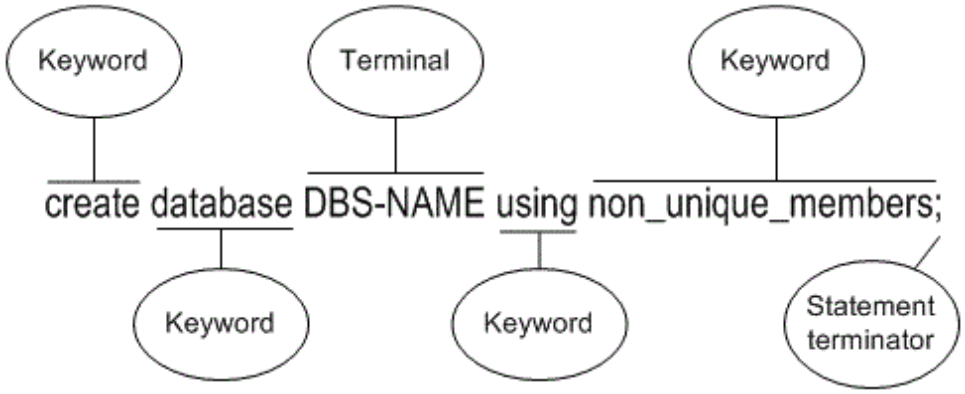
As a prerequisite to using MaxL, follow the client setup instructions in Manage Essbase Using the MaxL Client.

How to Read MaxL Railroad Diagrams

The MaxL grammar is illustrated using a railroad syntax notation. The railroad diagrams illustrate all the valid (grammatically correct) statements that can be parsed by MaxL.

Anatomy of MaxL Statements

- A keyword (see , represented in plain, lower-case font, is a unit of MaxL grammar. Keywords must be entered literally and in the correct order in MaxL statements. See the examples of keywords in the following diagram excerpt:



- A terminal, represented in upper-case without brackets, is replaced by values in the appropriate format as defined in the Terminals table. In the above diagram, DBS-NAME is a terminal. Terminals need to be replaced with a valid name; for example, `sample.basic`.

  Keywords cannot be used as terminals, unless enclosed in single quotation marks. For example, to create a database named database, the statement `create database database;` would return an error, but `create database "database";` would work.
- The semicolon indicates the end of a statement. Omitting a semicolon, or placing one before the expected end of a statement, results in a syntax error.
- A non-terminal, represented in upper-case with angle brackets <>, is defined in an additional diagram, usually below the main diagram. No non-terminal is shown here.

Railroad Diagram Symbols

The following table describes the meaning of symbols used in railroad diagrams.

Table 3-1 Railroad Diagram Symbols

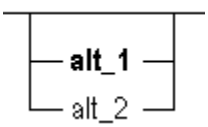
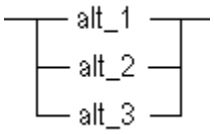
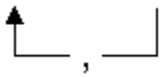
Symbol	Definition
	Statement begins here.
	Statement continues on next line.
	Statement is continued from previous line.
	Statement ends here.
	Alternatives: optionally select one keyword. Boldface indicates default if no selection is made.

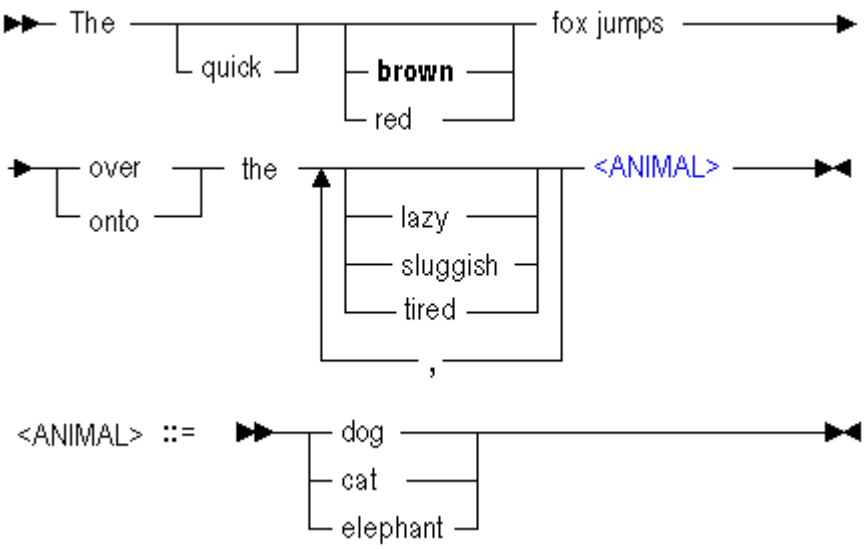
Table 3-1 (Cont.) Railroad Diagram Symbols

Symbol	Definition
	Alternatives: selection of one keyword is required.
	A comma-separated list of any length is permitted.
TERMINAL-NAME	Word is not further defined. Replace with value of format shown in the Terminals table.
<NON-TERMINAL>	Word used in statement is further defined.
<NON-TERMINAL> ::=	Non-terminal used in statements is defined here.

Sample Railroad Diagram

The following diagram illustrates a variant grammar that parses the following English sentence:

"The quick brown fox jumps over the lazy dog."



Keywords and variables on the main line (with arrow markings) are required; optional grammar is recessed (lower than the main line). A vertical stack of words represents alternatives. Bold words indicate defaults when no word is chosen.

Valid sentences parse-able by the example grammar may include:

- The fox jumps over the dog. Bold letters indicate a default value when no option is entered; therefore, entry of this statement would be interpreted as *The brown fox jumps over the dog*.

- The quick brown fox jumps over the dog.
- The red fox jumps over the lazy cat.
- The quick brown fox jumps onto the tired elephant.

MaxL Statements

The MaxL data-definition language has its own grammar that you use to create statements. In this document, the syntax for the MaxL DDL is illustrated using railroad diagrams.

The MaxL grammar is case-insensitive. Semicolon statement-terminators are required when using the MaxL Shell.

Key words of the MaxL grammar are represented in this document in lower-case. Terminals, represented in upper-case, are to be replaced by the appropriate names, numbers, privileges, or strings. For more information about components of MaxL statements, see MaxL Definitions.

Topics covered in this section:

- Performance Statistics in MaxL
- Listed By Verbs
- Listed by Objects
- MaxL Statement Reference

Performance Statistics in MaxL

Query database returns medium and long performance statistics for the database and application. The statistics appear as tables in the MaxL output. To gather performance statistics, you must first enable statistics gathering using alter database <dbs-name> set performance statistics enabled. You also use `alter database` to return to zero the statistical *persistence* (length) and *scope* (granularity).

Collecting and analyzing performance statistics can help you understand whether the databases are in good running condition or could use modifications to improve performance.

Topics related to performance statistics:

- The Essbase Performance Statistics Tables
- MaxL Script Example

The Essbase Performance Statistics Tables

The Essbase system gathers a variety of statistics regarding the performance of the system and the connected applications. The output of **query database** can vary depending on what the system has just done, how long statistics have been gathered and the persistence of the gathered statistics. The tables give information on a typical set of statistics. It can be very helpful to compare two sets of statistics gathered at similar points in the server's operation, such as after two comparable updates or after two restructure operations. Statistics should be gathered at intervals and compared to each other to identify differences. Compare the statistics gathered before and after any changes to the system and if the system performance changes.

> **Note:**
>
> Depending on the calculations you choose to perform, if any, some tables may or may not be displayed in your output log.

Performance statistics for which tables are available:

- Kernel Input/Output Statistics
- Kernel Cache Statistics
- Cache End-Transaction Statistics
- Database Synchronous Input/Output Statistics
- Database Asynchronous Input/Output Statistics
- Dynamic Calc Cache Statistics

Kernel Input/Output Statistics

The **Kernel I/O Statistics** table summarizes input/output for the entire application. There is one kernel I/O table per application.

Persistence/Scope of this table: **med/server**

Table 3-2 Kernel IO Statistics

Kernel I/O	Read (OS reads from disk)	Write (OS writes to disk)
# Index	I/O Number of reads that occurred through the index cache.	Number of writes that occurred through the index cache.
# Data I/O	Number of reads that occurred through the data cache.	Number of writes that occurred through the data cache.
# Fground I/O	Number of data reads that occurred in the foreground (while a process waited for data to be read).	Number of data writes that occurred in the foreground (while a process waited for data to be written).
# Index bytes	Number of bytes read from .IND files.	Number of bytes written to .IND files.
# Data bytes	Number of bytes read from .PAG files.	Number of bytes written to .PAG files.
Av byte/dat I/O	Average byte size of data reads. A high number is preferable.	Average byte size of data writes. A high number is preferable.

Kernel Cache Statistics

The **Kernel Cache Statistics** table assists in sizing database caches. Make caches only as large as necessary for optimum performance. Note that cache sizes are listed in order of importance: index, data file, data.

- The index cache is a buffer in memory that holds index pages.
- The data file cache is a physical data cache layer designed to hold compressed data blocks.
- The data cache is a buffer in memory that holds data pages.

The Kernel Cache Statistics table assists you in determining how to size Essbase caches. The Essbase kernel uses these caches to manage memory. As a rule, data that is useful to processes should be kept in memory rather than on a disk. Replacements occur when something needed for a process is moved from disk to cache and something in the cache is thrown away to make room for it.

Use this table to help you decide how to size your caches. Make the caches as small as possible; however, if replacements for a cache are greater than 0, the cache may be too small. Appropriate sizing of the Index cache is the most important for optimal performance; appropriate sizing of the Data cache is the least important.

Persistence/Scope of this table: **long/db**

Table 3-3 Kernel Cache Statistics

Kernel Cache Statistic	Description
# Blocks	Number of blocks actually in the Index cache, Data file cache, and Data cache. The block size multiplied by the number of blocks equals the amount of cache memory being used. Compare this figure to the block estimation you initially used to size your database.
# Replacements	Number of replacements per cache. Replacements occur when data moves from disk to cache and something in the cache is deleted to make room. If the number or replacements is low or zero, the cache might be set too large.
# Dirty repl	Number of dirty replacements per cache. A dirty replacement is one that requires a write to the disk before cache memory can be reused by a process. The data needed for the process is "dirty" because it was modified in memory but not saved to the disk. Dirty replacements are inefficient and expensive. They indicate that a cache might be too small.
Log blk xfer in	Number of logical blocks transferred to the Data file cache and Data cache (this measurement is not applicable for the Index cache.) If you are changing cache sizes, it may be instructive to study this statistic and note changes in data traffic.

Cache End-Transaction Statistics

The **Cache End-Transaction Statistics** table measures DBWriter efficiency. DBWriter is an asynchronous (or no-wait) Essbase thread, which searches the cache finding information that needs to be written to a disk.

The Cache End-Transaction Statistics table shows the cleanup state at the end of a transaction. These statistics are designed to measure DBWriter efficiency. DBWriter is an asynchronous (or no-wait) thread, which searches the cache and finds information that needs to be written to a disk. Because the DBWriter only operates during idle times, measuring the DBWriter activity can give an idea of the amount of idle time. This number should be high, indicating that the DBWriter had enough idle time to support the database effectively. Keep these statistics available for diagnostic purposes, in case you need to call technical support.

Persistence/Scope of this table: **med/db**

Database Synchronous Input/Output Statistics

The **Database Synchronous I/O** table tracks synchronous input/output. Synchronous means that the thread or program waits for the I/O to finish before proceeding. The **Tave (us)** column shows the bandwidth (bytes/Ttotal).

Persistence/Scope of this table: **med/db**

Table 3-4 DB Sync IO Statistics

DataBase Synch I/O	Count	Bytes	Ttotal (ms)	Tave (ms)
Index Read An occurrence of the OS reading index information from a .IND file on the disk.	Number of times the OS went to the disk to read a .IND file.	Number of bytes the OS read from .IND files.	Total amount of time the OS took to complete index reads.	Average amount of time the OS took to complete one index read. This equals Ttotal (ms)/Count.
Index Write An occurrence of the OS writing index information to a .IND file.	Number of times the OS wrote information to a .IND file.	Number of bytes the OS wrote to .IND files.	Total amount of time the OS took to complete index writes.	Average amount of time the OS took to complete one index write. This equals Ttotal (ms)/Count.
Data Read An occurrence of the OS reading information from a .PAG file on the disk.	Number of times the OS went to the disk to read to a .PAG file.	Number of bytes the OS read from .PAG files.	Total amount of time the OS took to complete data reads.	Average amount of time the OS took to complete one data read. This equals Ttotal (ms)/Count.
Data Write An occurrence of the OS writing data to a .PAG file.	Number of times the OS wrote information to a .PAG file.	Number of bytes the OS wrote to .PAG files.	Total amount of time the OS took to complete data writes.	Average amount of time the OS took to complete one data write. This equals Ttotal (ms)/Count.

> **Note:**
>
> Bandwidth = bytes/Ttotal. Average bandwidth = bytes/Tave.

Database Asynchronous Input/Output Statistics

The **Database Asynchronous I/O** table tracks asynchronous input/output. Asynchronous means no-wait: the I/O happens at an unknown time, while the program does other things. The effective bandwidth for the application is determined by bytes/Twait.

Persistence/Scope of this table: **med/db**

Table 3-5 DB Async IO Statistics

DataBase Asynch I/O	Count	Bytes	Ttotal (ms)	Tave (ms)	Twait (ms)
Index Read An occurrence of the OS reading index information from a .IND file on the disk.	Number of times the OS went to the disk to read a .IND file.	Number of bytes the OS read from .IND files.	Time elapsed between request for an index read, and verification of its completion.	Average time elapsed between requests for index reads, and verification of their completion.	Wait time if the OS had not completed index reads at the time polled.
Index Write An occurrence of the OS writing index information to a .IND file.	Number of times the OS wrote information to a .IND file.	Number of bytes the OS wrote to .IND files.	Time elapsed between request for an index write, and verification of its completion.	Average time elapsed between requests for index writes and verification of their completion.	Wait time if the OS had not completed index writes at the time polled.
Data Read An occurrence of the OS reading information from a .PAG file on the disk.	Number of times the OS went to the disk to read to a .PAG file.	Number of bytes the OS read from .PAG files.	Time elapsed between request for a data read, and verification of its completion.	Average time elapsed between requests for data reads, and verification of their completion.	Wait time if the OS had not completed data reads at the time polled.
Data Write An occurrence of the OS writing data to a .PAG file.	Number of times the OS wrote information to a .PAG file.	Number of bytes the OS wrote to .PAG files.	Time elapsed between request for a data write, and verification of its completion.	Average time elapsed between requests for data writes and verification of their completion.	Wait time if the OS had not completed data writes at the time polled.

Note:

(1) Because asynchronous I/O is ideally no-wait, and happens at an unknown time, you cannot determine how long reads and writes actually took to complete. (2) You cannot determine the bandwidth (bytes per microsecond). Effective bandwidth, as seen by the application, is determined by bytes/Twait.

Dynamic Calc Cache Statistics

The **Dynamic Calc Cache table** shows where blocks that are expanded to contain calculated members (BigBlks) are calculated: in dynamic calculator cache (DCC), or in regular memory (nonDCC). By viewing the total number of big blocks allocated versus the maximum number of big blocks held simultaneously, and by analyzing block wait statistics, you can determine the efficiency of your dynamic calc cache configuration settings (including DYNCALCCACHEMAXSIZE).

Table 3-6 Dynamic Calc Cache Statistics

Dynamic Calc Cache Statistic	Description
BigBlks Alloced	The number of big block allocations that have been requested, so far, irrespective of where the system got the memory (DC cache or regular). For three queries Q1, Q2, and Q3 executed, requiring 25, 35, and 10 big blocks, respectively, BigBlks Alloced would be 70. This does not mean that Q1 needed all 25 blocks at the same time. It may have used some blocks for a while, then released some of them, and so on, until the query finished and released all remaining blocks (returned to DC cache or regular memory).
Max BigBlks Held	The maximum number of big blocks simultaneously held, so far. For each query Qi executed so far, there will be a number Ni, which gives the maximum number of big blocks that the query needed to have at the same time (includes both DCC and regular memory blocks). MaxBigBlksHeld under the Total column is the maximum over all values of Ni. The values under the DCC and non-DCC columns are similar except that they restrict themselves to the maximum blocks held in the respective portions of memory.
DCC Blks Waited	The number of dynamic calculator blocks that the system had to wait for.
DCC Blks Timeout	The number of times that Essbase timed out waiting for free space in the dynamic calculator cache.
DCC Max ThdQLen	The highest number of threads that were left waiting in queue for Dynamic Calc cache memory to be freed.

MaxL Script Example

The following MaxL script creates an output file of performance statistics tables.

```
/* to execute:
   essmsh scriptname username password
*/
login $1 $2;
spool on to 'c:\mxlouts\pstatsouts.txt';
alter database sample.basic set performance statistics enabled;
execute calculation
  'SET MSG ERROR;
   CALC ALL;'
on Sample.basic;
alter database sample.basic set performance statistics mode to medium
persistence server scope;
query database sample.basic get performance statistics kernel_io table;
alter database sample.basic set performance statistics mode to long
```

```
persistence database scope;
query database sample.basic get performance statistics kernel_cache
table;
alter database sample.basic set performance statistics mode to medium
persistence database scope;
query database sample.basic get performance statistics end_transaction
table;
query database sample.basic get performance statistics database_synch
table;
query database sample.basic get performance statistics database_asynch
table;
spool off;
logout;
```

Listed By Verbs

Alter

trigger

Create

application

calculation

database

drillthrough

filter

location alias

outline

partition

trigger

Display

application

calculation

database

drillthrough

filter

filter row

group

location alias

lock

object

partition

privilege

session

system

tablespace

trigger

trigger spool

user

variable

Drop

- application
- calculation
- database
- drillthrough
- filter
- location alias
- lock
- object
- partition
- trigger
- trigger spool

Execute

- aggregate process
- aggregate selection
- aggregate build
- allocation
- calculation
- custom calculation (aggregate storage)

Export

- data
- LRO
- outline
- query_tracking

Grant

- Grant

Import

- data
- dimensions
- lro

query_tracking

MaxL Shell Invocation

The MaxL Shell (`essmsh`) is a pre-parser mechanism for entering MaxL statements.

You can start the shell to be used interactively, to read input from a file, or to read stream-oriented input (standard input from another process). You can log in after you start the shell, interactively or using a login statement in the input file. You can also log in at invocation time, by using the -l flag (see -l Flag: Login).

To start the `essmsh` shell, do not invoke it directly. In order for the environment to be set correctly, you must start `essmsh` using `startMAXL.bat` (Windows) or `startMAXL.sh` (UNIX).

- Prerequisites for Using MaxL
- MaxL Invocation Summary
- Interactive Input
- File Input
- Standard Input
- Login
- LoginAs
- Encryption
- Query Cancellation

Prerequisites for Using MaxL

Before the Essbase Server can receive MaxL statements,

1. The Essbase Server must be running.
2. The MaxL Shell (`essmsh`) must be invoked (see MaxL Invocation Summary), if you are using the shell.
3. You must log in (see Login) to the Essbase Server from the MaxL Shell. If you are running a MaxL script, the first line of your script must be a login statement.

You must use a semicolon (;) to terminate each MaxL statement.

MaxL Invocation Summary

The following MaxL Shell help page summarizes invocation options. This help is also available at the operating-system command prompt if you type `startMAXL.bat -h | more`.

> **Note:**
>
> The following help text is for `essmsh` shell; however, in order for the environment to be set correctly, you must start `essmsh` using `startMAXL.bat` (Windows) or `startMAXL.sh` (UNIX). You can pass the same arguments to `startMAXL` as you would formerly pass to `essmsh`. For example, instead of `essmsh -l username password`, you should now use `startMAXL.bat -l username password`.

```
essmsh(1)

NAME
     essmsh -- MaxL Shell

SYNOPSIS
     essmsh [-hlsmup] [-a | -i | file] [arguments...]

DESCRIPTION
     This document describes ways to invoke the MaxL Shell.
     The shell, invoked and nicknamed essmsh, takes input in the
following
     ways: interactively (from the keyboard), standard input (piped
from another
     program), or file input (taken from file specified on the command
line).
     The MaxL Shell also accepts any number of command-line arguments,
     which can be used to represent any name.

OPTIONS
     essmsh accepts the following options on the command line:

     -h
      Prints this help.

     -l <user> <pwd>
      Logs in a user name and password to the local Essbase Server
instance.

     -u <user>
      Specifies a user to be logged in to an Essbase Server instance.
      If omitted but the '-p' or '-s' flags are used, essmsh will
      prompt for the username.

     -p <pwd>
      Specifies a password of the user set by the '-u' option to
      be logged in to an Essbase Server instance. If omitted, essmsh
      will prompt for the password, and the password will be hidden
      on the screen.

     -s <server>
      Used after -l, or with [-u -p], logs the specified user into a
named
```

```
      server. When omitted, localhost is implied.

     -m <msglevel>
      Sets the level of messages returned by the shell. Values for
<msglevel>
      are: all (the default), warning, error, and fatal.

     -i
      Starts a MaxL session which reads from <STDIN>, piped in from
another program.
      The end of the session is signalled by the EOF character in that
program.

     -a
      Allows a string of command-line arguments to be referenced from
within the
      subsequent INTERACTIVE session. These arguments can be referenced
with positional
      parameters, such as $1, $2, $3, etc. Note: omit the -a when using
arguments with
      a file-input session.

NOTES

     No option is required to pass a filename to essmsh.

     Arguments passed to essmsh can represent anything: for example, a
user name, an
     application name, or a filter name. Arguments must appear at the
end of the
     invocation line, following '-a', '-i', or filename.

EXAMPLES

     Interactive session, simplest case:
          essmsh

     Interactive session, logging in a user:
          essmsh -l user pwd

     Interactive session, logging user in to a server:
          essmsh -l user pwd -s server

     Interactive session, logging in with two command-line arguments
     (referenced thereafter at the keyboard as $1 and $2):
           essmsh -l user pwd -a argument1 argument2



     Interactive session, with setting the message level:
      essmsh -m error

     Interactive session, hiding the password:
      essmsh -u user1
      Enter Password > ******
```

```
File-input session, simplest case:
     essmsh filename

File-input session, with three command-line arguments
(referenced anonymously in the file as $1, $2, and $3):
     essmsh filename argument1 argument2 argument3


Session reading from <STDIN>, logging into a server with two
command-line arguments:
     essmsh -l user pwd -s server -i argument1 argument2
```

Interactive Input

You can log into the MaxL Shell for interactive use (typing statements at the keyboard) in the following ways. See MaxL Invocation Summary for more descriptions of login flags.

No Flag

-a Flag: Arguments

-l Flag: Login

-u, -p, and -s Flags: Login Prompts and Hostname Selection

-m Flag: Message Level

No Flag

Invoked without a flag, file name, or arguments, the MaxL Shell starts in interactive mode and waits for you to log in. Note to UNIX users: In the following examples, replace `startMAXL.bat` with `startMAXL.sh`.

Example:

```
startMAXL.bat



 Essbase MaxL Shell - Release 11.1.2
 Copyright (c) 2000, 2010, Oracle and/or its affiliates.
 All rights reserved.
MAXL> login Fiona identified by sunflower;

      49 - User logged in: [Fiona].
```

-a Flag: Arguments

With the -a flag, the MaxL Shell starts in interactive mode and accepts space-separated arguments to be referenced at the keyboard with positional parameters.

> **Note:**
>
> If interactive arguments are used with spooling turned on, variables are recorded in the log file just as you typed them (for example, $1, $2, $ARBORPATH).

Example:

```
startMAXL.bat -a Fiona sunflower appname dbsname



Essbase MaxL Shell - Release 11.1.1
Copyright (c) 2000, 2008, Oracle and/or its affiliates.
All rights reserved.
MAXL> spool on to 'D:\output\createapp.out';

MAXL> login $1 identified by $2;

      49 - User logged in: [Fiona].

MAXL> create application $3;

      30 - Application created: ['appname'].

MAXL> create database $3.$4 as Sample.Basic;

      36 - Database created: ['appname'.'dbsname'].

MAXL> echo $ARBORPATH;

C:\Hyperion\products\Essbase\EssbaseClient

MAXL> spool off;
```

Contents of logfile createapp.out:

```
MAXL> login $1 identified by $2;

 OK/INFO - 1051034 - Logging in user Fiona.
 OK/INFO - 1051035 - Last login on Friday, January 18, 2008 4:09:16 PM.
 OK/INFO - 1241001 - Logged in to Essbase.

MAXL> create application $3;

 OK/INFO - 1051061 - Application appname loaded - connection
established.
 OK/INFO - 1054027 - Application [appname] started with process id
[404].
 OK/INFO - 1056010 - Application appname created.

MAXL> create database $3.$4 as Sample.Basic;
```

```
 OK/INFO - 1056020 - Database appname.dbname created.

MAXL> echo $ARBORPATH;

C:\Hyperion\products\Essbase\EssbaseClient

MAXL> spool off;
```

-l Flag: Login

When the -l flag is used followed by a user name and password, the MaxL Shell logs in the given user name and password and starts in interactive or non-interactive mode. The user name and password must immediately follow the -l, and be separated from it by a space.

Example:

```
startMAXL.bat -l Fiona sunflower
```

Entered at the command prompt, this starts the MaxL Shell in interactive mode and logs in user **Fiona**, who can henceforth issue MaxL statements at the keyboard.

-u, -p, and -s Flags: Login Prompts and Hostname Selection

The MaxL Shell can be invoked using -u and -p options in interactive mode, for passing the user name and password to the shell upon startup. To be prompted for both username and password, use the -s option with the host name of the Essbase Server.

- If -s <host-name> is passed to the shell, MaxL will prompt for the user name and password, and the password will be hidden.

  ```
  startMAXL.bat -s localhost
  Enter UserName> admin
  Enter Password> ********

   OK/INFO - 1051034 - Logging in user admin.
   OK/INFO - 1051035 - Last login on Monday, January 28, 2003
  10:06:16 AM.
   OK/INFO - 1241001 - Logged in to Essbase.
  ```

- If -u <username> is passed to the shell and -p <password> is omitted, MaxL Shell will prompt for the password, and the password will be hidden.

  ```
  startMAXL.bat -u smith
   Enter Password > ******
  ```

- If -p <password> is passed to the shell and -u <username> is omitted, MaxL Shell will prompt for the user name.

  ```
  startMAXL.bat -p password
   Enter Username > smith
  ```

- If -m <messageLevel> is passed to the shell, only the specified level of messages will be returned by the shell.

  ```
  startMAXL.bat -m error
  ```

  Values for <messageLevel> include: **default**, **all**, **warning**, **error**, and **fatal**. The default value is **all** (same as specifying **default**).

-m Flag: Message Level

If -m <messageLevel> is passed to the shell, only the specified level of messages will be returned by the shell.

Example:`startMAXL.bat -m error`

Values for the <messageLevel> include: **default**, **all**, **warning**, **error**, and **fatal**. The default value is **all** (same as specifying **default**).

File Input

You invoke the MaxL Shell to run scripts (instead of typing statements at the keyboard) in the following ways. See MaxL Invocation Summary for a complete description of login flags.

File Only

File Only

File Only

If you type `startMAXL.bat` followed by a file name or path, the shell takes input from the specified file.

Examples:

```
startMAXL.bat
C:\Hyperion\products\Essbase\EssbaseClient\scripts\filename.msh
```

Entered at the command prompt, this starts the shell, tells it to read MaxL statements from a file, and terminates the session when it is finished.

```
startMAXL.bat filename
```

Starts the shell to read MaxL statements from `filename`, located in the current directory (the directory from which the MaxL Shell was invoked).

File with Arguments

If you type `startMAXL.bat` followed by a file name followed by an argument or list of space-separated arguments, essmsh remembers the command-line arguments, which can be referenced as `$1`, `$2`, etc. in the specified file. If spooling is turned on, all variables are expanded in the log file.

Example:

```
D:\Scripts>startMAXL.bat filename.msh Fiona sunflower localhost
```

Starts the shell to read MaxL statements from `filename.msh`, located in the current directory.

Contents of script filename.msh:

```
spool on to $HOME\\output\\filename.out;
login $1 $2 on $3;
echo "Essbase is installed in $ESSBASEPATH";
spool off;
exit;
```

Contents of logfile filename.out:

```
MAXL> login Fiona sunflower on localhost;

      49 - User logged in: [Fiona].

Essbase is installed in C:\Hyperion\products\Essbase\EssbaseClient
```

Standard Input

With the -i flag, essmsh uses standard input, which could be input from another process. For example,

```
program.sh | startMAXL.bat -i
```

When **program.sh** generates MaxL statements as output, you can pipe `program.sh` to `startMAXL.bat -i` to use the standard output of `program.sh` as standard input for essmsh. Essmsh receives input as `program.sh` generates output, allowing for efficient co-execution of scripts.

Example:

```
echo login Fiona sunflower on localhost; display privilege user;|
startMAXL.bat -i
```

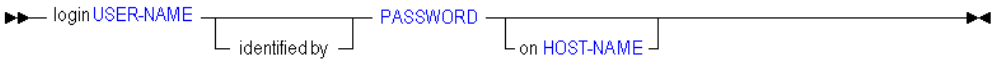
The MaxL Shell takes input from the echo command's output. User Fiona is logged in, and user privileges are displayed.

Login

Before you can send MaxL statements from the MaxL Shell to Essbase Server, you must log in to an Essbase Server session.

As a prerequisite to using MaxL, follow the client setup instructions in Manage Essbase Using the MaxL Client.

> **Note:**
>
> Before logging in to an Essbase Server session, you must start the MaxL Shell (see MaxL Invocation Summary).Or, you can start the MaxL Shell and log in (see -l Flag: Login) at the same time.



- USER-NAME
- PASSWORD
- HOST-NAME

Example

```
login admin pa5sw0rd on "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent";
```

Establishes a connection for user Admin.

LoginAs

To facilitate creating scheduled reports with user-appropriate permissions, administrators can log in as another user from MaxL.

Example of "log in as" statement:

```
loginas USER-NAME PASSWORD MIMICKED-USER-NAME [on HOST-NAME];
```

Example of "log in as" invocation method:

```
essmsh -la USER-NAME PASSWORD MIMICKED-USER-NAME [-s HOST-NAME]
```

Interactive example:

```
MAXL>loginas;
Enter UserName> username
Enter Password> password
Enter Host> machine_name
Enter UserName to Login As> mimicked_user_name
```

Encryption

You can encrypt user and password information stored in MaxL scripts.

The following MaxL Shell invocation generates a public-private key pair that you can use to encrypt a MaxL script.

```
essmsh -gk
```

The following MaxL Shell invocation encrypts the input MaxL script, obscuring user name and password, and changing the file extension to .mxls.

```
essmsh -E scriptname.mxl PUBLIC-KEY
```

Nested scripts are also encrypted. To avoid this and encrypt only the base script, use `-Em`.

The following MaxL Shell invocation decrypts and executes the MaxL script.

```
essmsh -D scriptname.mxls PRIVATE-KEY
```

The following invocation encrypts input data and returns it in encrypted form. This is useful if there is a need to manually prepare secure scripts.

```
essmsh -ep DATA PUBLIC-KEY
```

The following invocation enables you to encrypt the base script while saving any nested scripts for manual encryption.

```
essmsh –Em scriptname.mxl PUBLIC-KEY
```

Query Cancellation

You can use the Esc key to cancel a query running from MaxL Shell.

Logout

Log out from Essbase without exiting the interactive MaxL Shell.

Syntax

```
logout;
```

Example

```
logout;
```

Query

database

database backup archive file

application (for aggregate storage)

application (for block storage)

Refresh

outline

replicated partition

Listed by Objects

aggregate_build

aggregate_process

aggregate_selection

application

archive_file

calculation

data

database

dimensions

drillthrough

filter

group

location alias

lock

lro

object

outline

partition

privilege

session

system

tablespace

trigger

trigger spool

user

variable

Aggregate Build

execute aggregate build

Aggregate Process

execute aggregate process

Aggregate Selection

execute aggregate selection

Allocation

execute allocation

Application

alter

create

display

drop

query (for aggregate storage only)

Archive_file

query

Calculation

create

display

drop

execute

execute custom (aggregate storage)

Data

- export
- import

Database

- alter
- create
- display
- drop
- query

Dimensions

- import

Drillthrough

- alter
- create
- display
- drop

Filter

- alter filter
- create filter
- display filter
- display filter row
- drop filter

Group

- display

Location Alias

- create
- display
- drop

Lock

display

drop

LRO

export

import

Object

alter

display

drop

Outline

create

refresh

see also Dimensions

Partition

alter

create

display

drop

refresh replicated

refresh outline for outline synchronization

Privilege

display

grant

Session

alter

display

alter system to stop a session

System

alter

display

Tablespace

alter

display

Trigger

alter

create or replace

display

drop

Trigger Spool

display

drop

User

display

grant to assign permissions

Variable

display variable

To add, drop, or set substitution variables:

alter application

alter database

alter system

MaxL Statement Reference

Consult the Contents pane for an alphabetical list of MaxL statements, or see Listed By Verbs.

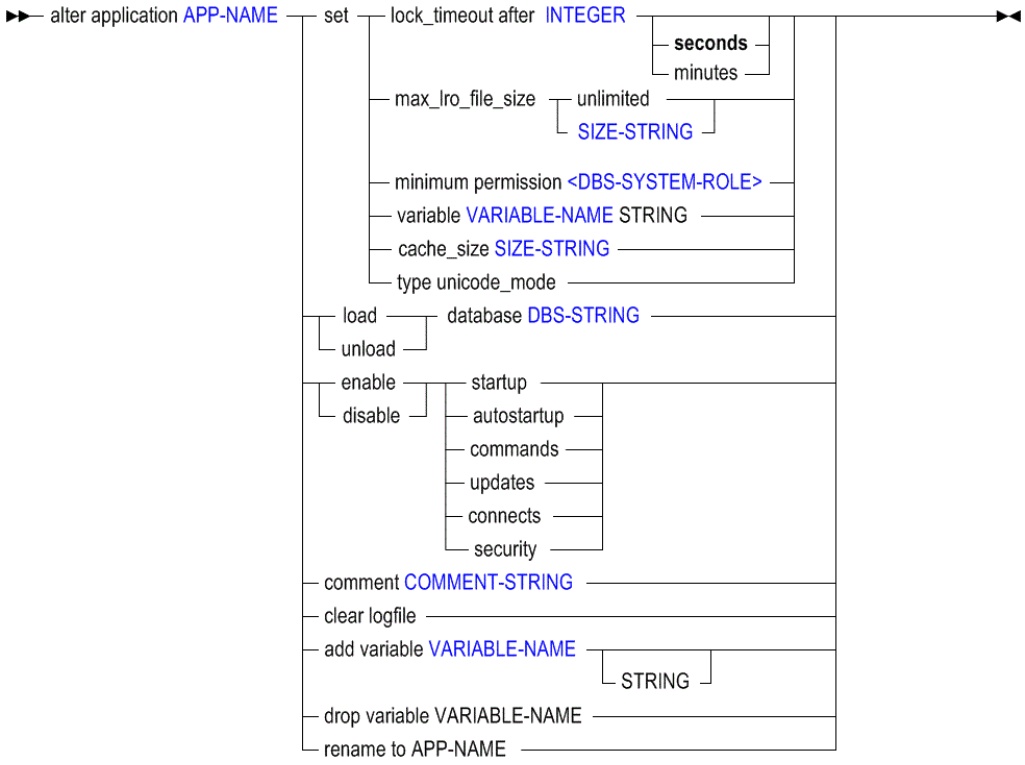
Alter Application

Click here for aggregate storage version

Change application-wide settings.

Permission required: Application Manager.

Syntax



- APP-NAME
- INTEGER
- SIZE-STRING
- DBS-SYSTEM-ROLE
- VARIABLE-NAME
- DBS-STRING
- COMMENT-STRING

Use **alter application** to change the following application-wide settings:

Keywords

set lock_timeout
Change the maximum time interval that locks on data blocks can be held by Smart View (or other grid clients') users. When a client data-block lock is held for more than the time out interval, Essbase removes the lock and the transaction is rolled back. The default interval is 60 minutes. This setting affects all databases in the application.

set max_lro_file_size
Specify a maximum file size for Linked Reporting Objects (LRO) attachments. There is no default. There is no minimum or maximum value, excepting limitations imposed by your system resources.

set minimum permission
Grant all users a minimum level of permission to all databases in the application. Users with higher permissions than this minimum are not affected.

set variable
Assign a string value to an existing substitution-variable name. If the variable does not exist, first create it using **add variable**. Substitution variables may be referenced by calculations in the application.

set cache_size
Set the maximum size to which the application cache may grow. The application cache grows dynamically until it reaches this limit. The application cache is used for hybrid aggregation in block storage databases, and can help you manage memory usage for retrievals. This setting takes effect after you restart the application. To check the currently set limit, use the following MaxL statement:

```
query application APP-NAME get cache_size;
```

set type unicode_mode
Migrate an application to Unicode mode. Migration to Unicode mode cannot be reversed.

load database
Start (by loading into memory) an idle database. The statement will fail if you do not have at least read privilege for the database.

unload database
Stop (by unloading from memory) an active database. The statement will fail if you do not have at least read privilege for the database.

enable startup
Permit all users to load (start) the application. This only applies to users who have at least read privilege for the application. Startup is enabled by default.

disable startup
Prevent all users from loading (starting) the application. Startup is enabled by default.

enable autostartup
Start the application automatically when Essbase Server starts. By default, autostartup is disabled.

disable autostartup
Do not start the application automatically when Essbase Server starts. By default, autostartup is disabled.

enable commands
Allow all users with sufficient permissions to make requests to databases in the application. Use to reverse the effect of **disable commands**. The disable commands setting remains in effect only for the duration of your session. By default, commands are enabled.

disable commands
Prevent all requests to databases in the application, including non-data-specific requests, such as viewing database information or changing database settings. All users are affected, including other administrators. Administrators are affected by this

setting as a safety mechanism to prevent accidental updates to databases during maintenance operations. This setting remains in effect only for the duration of your session. The setting takes effect immediately, and affects users who are currently logged in, as well as users who log in later during your session.

> **Caution:**
>
> If performing maintenance operations that require disabling commands, you must make those maintenance operations within the same session and the same script as the one in which commands were disabled.

By default, commands are enabled.

enable updates
Allow all users with sufficient permissions to make requests to databases in the application. Use to reverse the effect of **disable updates**. Disabling updates remains in effect only for the duration of your session. By default, updates are enabled.

disable updates
Prevent all users from making requests to databases in the application. Use before performing update and maintenance operations. The disable updates setting remains in effect only for the duration of your session.

> **Caution:**
>
> If performing maintenance operations that require updates to be disabled, you must make those maintenance operations within the same session and the same script as the one in which updates were disabled. By default, updates are enabled.

enable connects
Allow all users with sufficient permissions to make connections to databases in the application. Use to reverse the effect of **disable connects**. By default, connections are enabled.

disable connects
Prevent any user with a permission lower than Application Managers from making connections to the databases that require the databases to be started. This includes starting the databases or performing the ESSCMD shell SELECT command on the databases. Database connections remain disabled for all databases in the application, until the application setting is re-enabled by the administrator.
By default, connections are enabled.

enable security
When security is disabled, Essbase ignores all security settings in the application and treats all users as Application Managers. By default, security is enabled.

disable security
When security is disabled, Essbase ignores all security settings in the application and treats all users as Application Managers. By default, security is enabled.

comment
Enter an application description (optional). The description can contain up to 80 characters.

clear logfile
Delete the application log located in the application directory. A new log is created for entries recording subsequent application activity.

add variable
Create an application-level substitution variable by name, and optionally assign a string value for the variable to represent. You can assign or change the value later using **set variable**. A substitution variable acts as a global placeholder for information that changes regularly. Substitution variables may be referenced by calculations and report scripts.
If substitution variables with the same name exist at server, application, and database levels, the order of precedence for the variables is as follows: a database level substitution variable supersedes an application level variable, which supersedes a server level variable.

drop variable
Remove a substitution variable and its corresponding value from the application.

rename to
Rename the application. When you rename an application, the application and the application directory (*ARBORPATH*\app*appname*) are renamed.

Example

```
alter application Sample set minimum permission read;
```

Grants all users read access to all databases in the Sample application. Users can retrieve data values and run report scripts.

```
alter application Sample disable commands;
```

Prevents all users from making requests to the application scope. Use this statement before performing application-wide update and maintenance operations.

```
alter application Acme set variable Current_month July;
```

Assigns the string value July to the substitution variable "Current_month." "Current_month" may be referenced by calculations in the Acme application.

Alter Database

Click here for aggregate storage version

Select a subset of **alter database**:

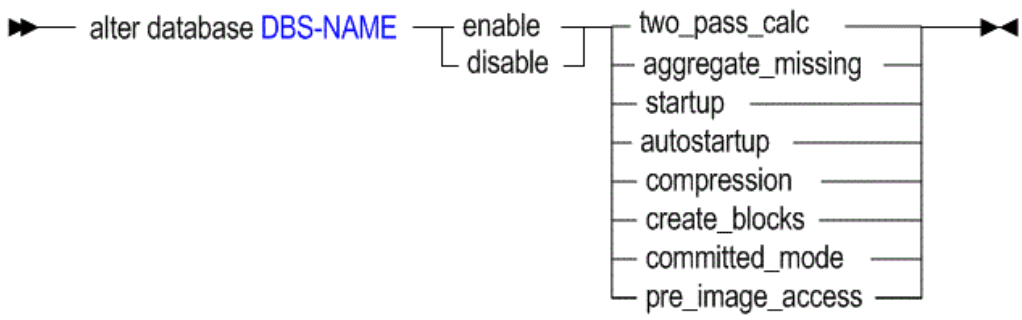
Alter Database enable | disable

Click here for aggregate storage version

Change database-wide settings.

Permission required: create_application.

Syntax



DBS-NAME

Use **alter database** to change the following database-wide settings:

Keywords

enable two_pass_calc
Recalculate (after a default calculation) database outline members tagged as Two Pass, so they will be recalculated after other database members have been consolidated. This setting is enabled by default.
Members that usually require a two-pass calculation are those members of the Accounts dimension that are calculated by a formula rather than by hierarchical consolidation. These members are typically ratios, such as "Profit % Sales" (profit percentage of sales), which has a member formula.
This setting is ignored during a calculation script; it is used only during a default calculation. To use two-pass calculation in a non-default calculation, use the CALC TWOPASS command in the calculation script.

disable two_pass_calc
Do not recalculate database outline members tagged as Two Pass after a default calculation. Two-pass calculation is enabled by default.

enable aggregate_missing
Consolidate #MISSING values along with the regular database consolidation. If you never load data at parent levels, aggregating #MISSING values can improve calculation performance, depending on the ratio between upper level blocks and input blocks in the database.

If this setting is enabled and you load values directly at the parent level, these parent-level values will be replaced by the results of the consolidation, even if the results are #MISSING values. The aggregate missing setting is disabled by default.

disable aggregate_missing
Do not consolidate #MISSING values. This is the default. Data that is loaded at parent levels is not overwritten by #MISSING values of children below it. However, if any of the child data values are not #MISSING, these values are consolidated and overwrite the parent values.

enable startup
Enable users to start the database directly or as a result of requests requiring the database to be started. Startup is enabled by default.

disable startup
Prevent all users from starting the database directly or as a result of requests that would start the database. Startup is enabled by default.

enable autostartup
Automatically start the database when the application to which it belongs starts. Autostartup is enabled by default. This setting is applicable only when startup is enabled.

disable autostartup
Prevent automatic starting of the database when the application to which it belongs starts. Autostartup is enabled by default.

enable compression
Enable data compression. By default, Bitmap compression is enabled. To switch to a different compression type, use `alter database set compression`.

disable compression
Disable data compression. By default, Bitmap compression is enabled.

enable create_blocks
Allow Essbase to create a data block when you assign a non-constant value to a member combination for which a data block does not already exist. Block creation on equation is disabled by default, because it can result in a very large database.
When you assign a constant to a member on a sparse dimension, you do not need to enable Create Blocks on Equation, because Essbase would create a data block anyway. For example, `"West = 5;"` would result in the creation of data blocks, with or without the Create Blocks on Equation setting enabled.
You do need to check this option if you want blocks created when you assign anything other than a constant to a member on a sparse dimension for which a data block does not already exist. For example, if no data exists for Actuals, a member of a sparse Scenario dimension, then you need to enable Create Blocks on Equation in order to perform the following allocation:

```
2002Forecast = Actuals * 1.05;.
```

disable create_blocks
Turn off the Create Blocks on Equation setting. The setting is disabled by default.

enable committed_mode
Set the database isolation level to committed access, meaning that only one transaction at a time can update data blocks. Essbase holds read/write locks on all data blocks until the transaction and the commit operations are performed. If pre-image access is enabled, users (or transactions) can still have read-only access to data at its last commit point. For more information, see the `enable pre_image_access` setting. The default isolation-level mode is Uncommitted.

disable committed_mode
Turn off the Committed Mode setting, reverting to the default isolation level of Uncommitted for the database.

> **Note:**
>
> Smart View and other grid clients' data-update operations are always in committed mode.

In uncommitted mode, Essbase allows transactions to hold read/write locks on a block-by-block basis. Essbase releases a block after it is updated, but does not commit blocks until the transaction is completed, or until a specified number of blocks or rows (a "synchronization point") has been reached. You can set this limit using the `implicit_commit` settings.

enable pre_image_access
Allow users (or other transactions) read-only access to data at its last commit point, when the database is in committed mode (meaning that data blocks may be locked for the duration of a concurrent transaction). Pre-image access is enabled by default when the database is in committed mode.
See also the `enable committed_mode` setting.

disable pre_image_access
Disable pre-image access, disallowing read-only access to locked blocks of data at their last commit point (this setting is only applicable while the database is in committed mode). Pre-image access is enabled by default when the database is in committed mode.

Example

```
alter database Sample.Basic disable two_pass_calc;
```

Prevents recalculation (after a default calculation) of members tagged as Two Pass.

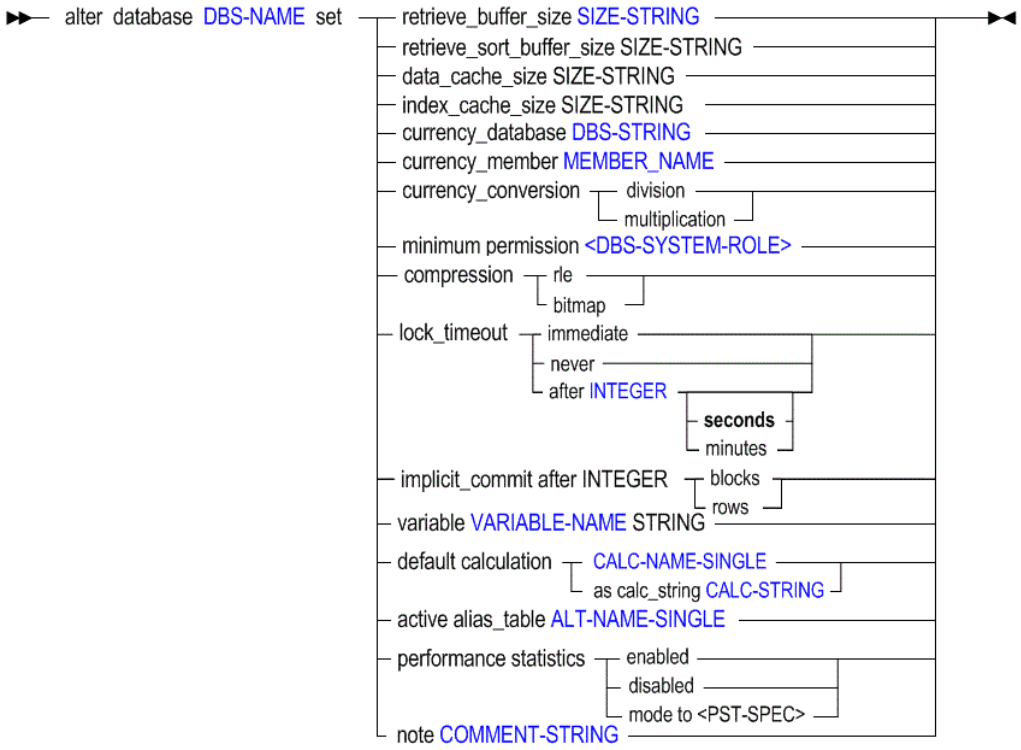
Alter Database Set

Click here for aggregate storage version

Change database-wide settings.

Permission required: create_application.

Syntax



- DBS-NAME
- SIZE-STRING
- DBS-STRING
- MEMBER-NAME
- DBS-SYSTEM-ROLE
- INTEGER
- VARIABLE-NAME
- CALC-NAME-SINGLE
- CALC-STRING
- ALT-NAME-SINGLE
- COMMENT-STRING

Use **alter database set** to change the following database-wide settings:

Keywords

retrieve_buffer_size
Change the database retrieval buffer size. This buffer holds extracted row data cells before they are evaluated by the RESTRICT or TOP/BOTTOM Report Writer commands. The default size is 20 KB. The minimum size is 2 KB. Increasing the size may improve retrieval performance.

retrieve_sort_buffer_size
Change the database retrieval sort buffer size. This buffer holds data until it is sorted. The Report Writer and Essbase Query Designer use the retrieval sort buffer. The default size is 20 KB. The minimum size is 2 KB. Increasing the size may improve retrieval performance.

data_cache_size
Change the data cache size. The data cache is a buffer in memory that holds uncompressed data blocks. Essbase Server allocates memory to the data cache during data load, calculation, and retrieval operations as needed. The default and minimum size is 3072 KB.

index_cache_size
Change the index cache size. The index cache is a buffer in memory that holds index pages. When a data block is requested, Essbase looks at the index pages in the index cache to find its location on disk.

- Minimum value: 1 MB (1,048,576 bytes)
- Maximum value: 256 TB

  Default value for buffered I/O: 1 MB (1,048,576 bytes)

Buffered I/O is the default for this release.

currency_database
Link the database with a currency database. A currency database enables you to convert currency values in a database from one currency into another currency.

currency_member
Specify the member to use as a default value in currency conversions. You can specify any valid member of the dimension defined as "Currency Type" in the currency database.

currency_conversion
Specify whether during currency conversion, the calculation method multiplies the currency database exchange rates with the main database values, or that the currency database exchange rates are divided by the main database values.

minimum permission
Set a level of permission that all users or groups can have to the database. Users or groups with higher granted permissions than the minimum permission are not affected.

compression rle
Set the database to use run-length encoding (RLE) compression. Essbase compresses repetitive, consecutive values, including zeros and #MISSING values. The default compression type is bitmap.
When a compressed data block is brought into the data cache, Essbase expands the block to its full size, regardless of the scheme that was used to compress it.

compression bitmap
Set the database to use bitmap compression, the default. Essbase stores only non-missing values and uses a bitmapping scheme.
When a compressed data block is brought into the data cache, Essbase expands the block to its full size, regardless of the scheme that was used to compress it.

lock_timeout
Change the interval to wait for blocks to be unlocked when the database is in committed mode. If a transaction request is made that cannot be granted in the allotted time, the transaction is rolled back until a lock can be granted.

> **Note:**
>
> Smart View and other grid clients' data-update operations are always in committed mode.

implicit_commit after <number> blocks
When uncommitted access is enabled, set the frequency at which Essbase commits data blocks (after the specified number of blocks has been reached).
The default frequency, if unspecified, is 3000, and may adjust dynamically during a calculation.
If Essbase Server runs on Oracle Exalytics In-Memory machine, for calculation and data load requests, the commit happens at the end of the command or request, and the default interval of 3000 (or any other value you specify) is ignored.

implicit_commit after <number> rows
When uncommitted access is enabled, set the frequency at which Essbase commits data blocks (after the specified number of rows has been reached).

variable
Change the value of an existing substitution variable on the database. The value must not exceed 256 bytes. It may contain any character except a leading ampersand (&).

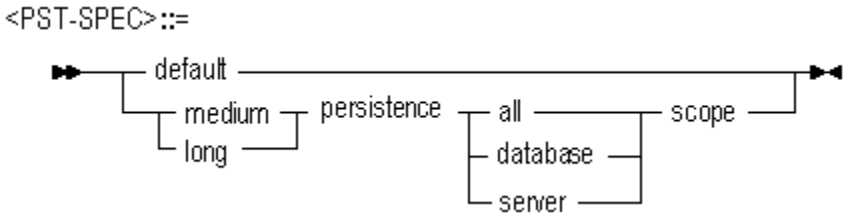
default calculation
Change the default calculation (which, by default, is `CALC ALL;`) to the stored calculation script you specify, or to an anonymous (unstored) calculation string.

active alias_table
Set an alias table as the primary table for reporting and any additional alias requests. Only one alias table can be used at a time. This setting is user-specific; it only sets the active alias table for the user issuing the statement.

performance statistics enabled
Turn on performance-statistics gathering. You might do this when you want to tune the system, change hardware configuration, or monitor I/O. The measurement begins for current processes as soon as you enable it. Any subsequent queries for statistics return measurements spanning from the time of enablement to the time of the query. Performance statistics can be retrieved using query database.

performance statistics disabled
Turn off performance-statistics gathering. This halts the collection of statistics; it does not prevent anyone from retrieving old statistics using query database.

performance statistics mode to <PST-SPEC>
Reset performance-statistics gathering for a specified persistence and scope. Each of the statistics tables available using `query database` has a pre-defined persistence and scope. When you use `set performance statistics mode`, you select the persistence and scope to reset, and the collecting of measurements starts over for the applicable tables.



note
Create an informational note about the database that Smart View or other grid client users can see from the login dialog box. For example, `'Calc in progress: do not update.'` Database notes can be up to 64 kilobytes long.

Example

```
alter database Sample.Basic set lock_timeout after 120;
```

Changes the number of seconds to wait for blocks to be unlocked. If a transaction request is made which cannot be granted in 120 seconds, the transaction is rolled back until a lock can be granted.

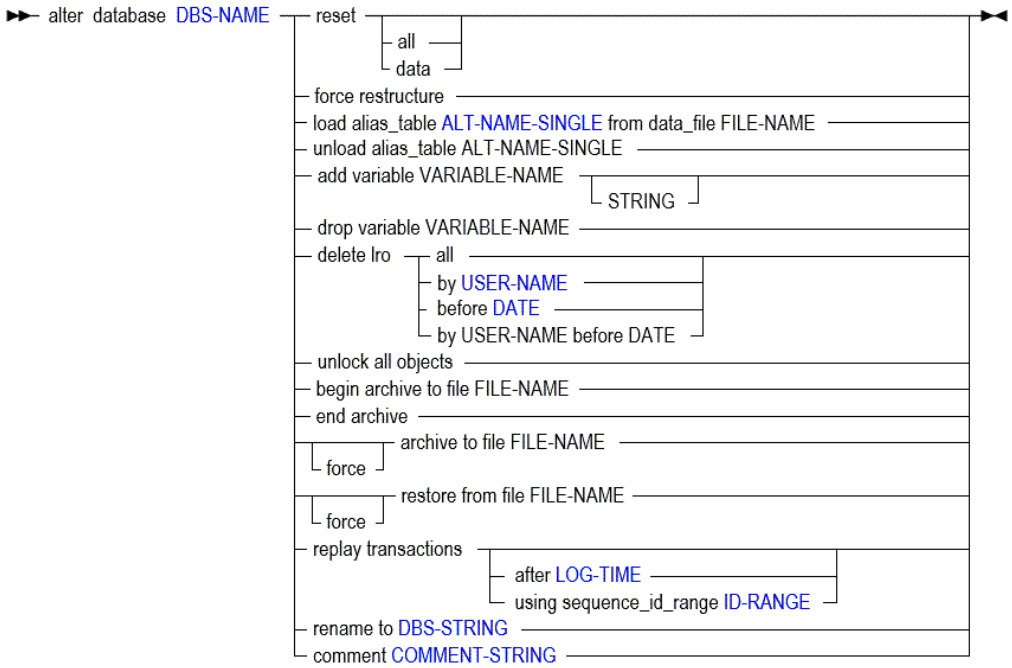
Alter Database (Misc)

Click here for aggregate storage version

Change database-wide settings.

Permission required: create_application.

Syntax



- DBS-NAME
- FILE-NAME
- ALT-NAME-SINGLE
- USER-NAME
- DATE
- LOG-TIME
- ID-RANGE
- DBS-STRING
- COMMENT-STRING

Use **alter database** to change the following database-wide settings:

Keywords

reset
Clear all data and linked-reporting objects from the database, but preserve the outline.

reset all
Clear all data, Linked Reporting Objects, and the outline.

reset data
Same as using `reset`.

force restructure
Explicitly restructure the database to eliminate or reduce fragmentation. By default, this statement is run in serial. To enable parallel restructuring, use the RESTRUCTURETHREADS configuration setting.

load alias_table
Load an alias table from a file to the current database. The feeder file (`FILE-NAME`) must follow these rules:

- Must be correctly formatted.
- Must be located on the Essbase Server computer, not on a client computer.
- `FILE-NAME` must include the full path.

Sample contents of a feeder file for loading an alias table:

```
$ALT_NAME
"400-10"        Guava
"400-20"        Tangerine
"400-30"        Mango
$END
```

unload alias_table
Delete the specified alias table.

add variable
Create a database-level substitution variable by name, and optionally assign a string value for the variable to represent. You can assign or change the value later using `set variable`. A substitution variable acts as a global placeholder for information that changes regularly. Substitution variables may be referenced by calculations and report scripts.
If substitution variables with the same name exist at server, application, and database levels, the order of precedence for the variables is as follows: a database level substitution variable supersedes an application level variable, which supersedes a server level variable.

drop variable
Remove a substitution variable and its corresponding value from the database.

delete lro
Delete Linked Reporting Objects linked to the active database for a given user name or modification date.

unlock all objects
Unlock all objects on the database that are in use by a user or process.

begin archive to file
Prepare the database for backup by an archiving program, and prevent writing to the files during backup.
This statement requires the database to be started.
Begin archive achieves the following outcomes:

- Commits any modified data to disk.
- Switches the database to read-only mode. The read-only state persists, even after the application is restarted, until it is changed back to read-write using `end archive`.
- Reopens the database files in shared, read-only mode.

- Creates a file containing a list of files that need to be backed up. Unless a different path is specified, the file is stored in the database directory.

Begin archive and end archive do not perform the backup; they simply protect the database during the backup process.

> **Note:**
>
> Using the **begin archive to file** and **end archive** grammar is the only supported way to initiate backup and recovery of a database using MaxL.

end archive
Return the database to read-write mode after backing up the database files.
This statement requires the database to be started.
End archive achieves the following outcomes:

- Returns the database to read-write mode.
- Re-opens database files in exclusive, read-write mode.

> **Note:**
>
> Using the **begin archive to file** and **end archive** grammar is the only supported way to initiate backup and recovery of a database using MaxL.

replay transactions
Replays the database transactions that were logged after the last replay request was originally executed or after the last restored backup's time (whichever occurred later). Transactions that are executed and logged after the restore operation are not replayed, unless you replay those transactions using their sequence IDs. After restoring a database, Oracle recommends that you finish replaying the transactions that were logged after the backup and before the restore and that are needed to fully recover the database; then you can continue executing new transactions.

replay transactions after LOG-TIME
Replays the transactions that were logged after the specified time. Enclose the TIME value in quotation marks; for example: '11_20_2007:12:20:00'

replay transactions using sequence_id_range ID-RANGE
Replays the transactions specified by a comma-separated list of sequence ID ranges. A range can consist of:

- A single transaction: *n* to *n*; for example, 1 to 1
- Multiple transactions: *x* to *y*; for example, 20 to 100

Each logged transaction is assigned a sequence ID, indicating the order in which the transaction was performed. To ensure the integrity of the restored data after a replay, Essbase enforces the replay of transactions in the same order in which they were originally performed. The order of sequence IDs are tracked across multiple replay commands.

> **Note:**
>
> You can skip replaying a transaction if you are absolutely sure that the transaction results are not required to recover the database.

rename to
Rename the database. When you rename a database, the database directory is also renamed.

comment
Create a description of the database. The maximum number of characters is 80. This description is available to database administrators. To annotate the database for Smart View or other grid client users, use `set note`.

Example

```
alter database Sample.Basic begin archive to file 'samplebasic.arc';
```

Backs up the Sample.Basic database files to the specified archive file (`samplebasic.arc`).

```
alter database Sample.Basic end archive
```

Returns the Sample.Basic database to read-write mode after backing up the database files.

```
alter database Sample.Basic replay transactions using sequence_id_range
1 to 10,20 to 100;
```

Replays the transactions in the Sample.Basic database with sequence IDs 1 through 10 and 20 through 100.

```
alter database Sample.Basic replay transactions after
'11_20_2007:12:20:00';
```

Replays all transactions that were logged after the specified time.

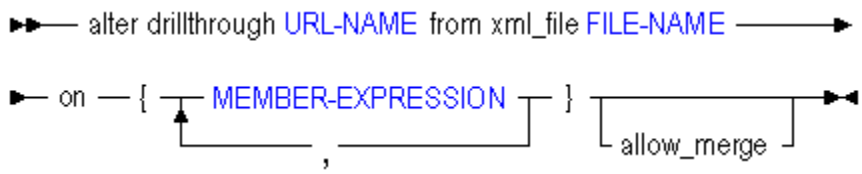
Related Topics

- Alter Database enable | disable
- Alter Database Set

Alter Drillthrough

Edit drill-through URL definitions used to link to content hosted on Oracle ERP and EPM applications.

Syntax



- URL-NAME
- FILE-NAME
- MEMBER-EXPRESSION

Use **alter drillthrough** to edit a URL definition in the following ways:

Keywords

alter drillthrough
Edit drill-through URL metadata.
The number of drill-through URLs per database is limited to 255.

from xml_file
Indicate the path to the local URL XML file that defines the link information.
The URL XML is created by the ERP or EPM application that deployed the Essbase database. The XML contains the drill-through URL display name as well as a URL enabling the hyperlink from a cell to a Web interface to occur. For a sample URL XML file, see Create Drillthrough.

on {<member-expression>,...}
Define the list of drillable regions, using the same Essbase member-set calculation language that is used to define security filters. The list of drillable regions must be enclosed in {brackets}.
The number of drillable regions in a drill-through URL is limited to 256. The number of characters per drillable region is limited to 65536.

allow_merge
Optional: Merge the drillable-region definition instead of replacing it on update.

Example

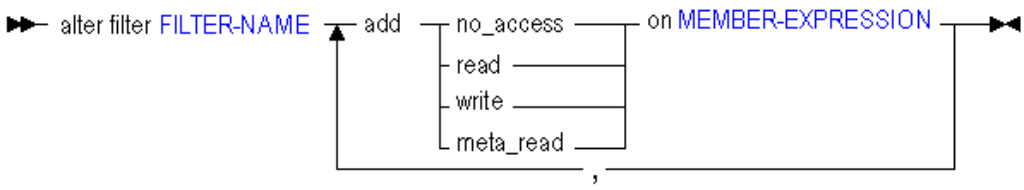
```
alter drillthrough sample.basic.myURL from xml_file "C:/drillthrough/
data/myfile.xml" on {'@Ichildren(“Qtr1”)', '@Ichildren(“Qtr2”)'}
allow_merge;
```

Alter Filter

Add filter rows to a database security filter. Filters control security for database objects. Use grant to assign filters to users and groups.

Minimum permission required: Database Manager.

Syntax



- FILTER-NAME
- MEMBER-EXPRESSION

Use **alter filter** in the following ways to edit a filter:

Keywords

alter filter ...add no_access on <member-expression>
Block access to a specified member combination.

alter filter ... add read on <member-expression>
Provide read-only access to a specified member combination.

alter filter ... add write on <member-expression>
Provide write access to a specified member combination.

alter filter ... add meta_read on <member-expression>
Restrict access to siblings and ancestors of the member expression. In case of a filtering conflict, the MetaRead filtering overrides the other filter permissions. For more information about metatdata filtering, see Metadata Filtering.

Notes

- Filters created using MaxL must be valid. For information about filter syntax, see *Designing and Maintaining Essbase Cubes*.
- MEMBER-EXPRESSION must be enclosed in single quotation marks. It can be a comma-separated list.

Example

```
alter filter sample.basic.filt7 add read on '@Descendants("East")';
```

Adds a row to a Sample.Basic filter named filt7, giving read-only access to the data for the eastern states.

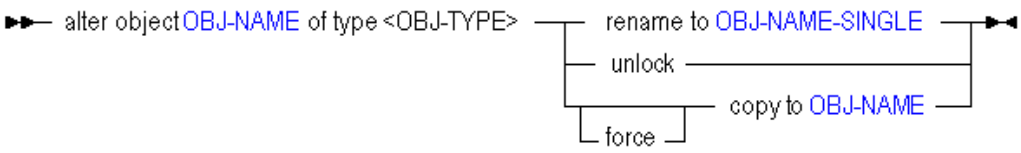
```
alter filter sample.basic.filt8 add read on '@Descendants("East")', add
write on '@Descendants("West")';
```

Adds two rows to a Sample.Basic filter named filt8.

Alter Object

Rename, unlock, or copy a database-related artifact.

Syntax



- OBJ-NAME
- OBJ-NAME-SINGLE

Use **alter object** to edit artifacts in the following ways:

Keywords

rename to
Rename the artifact. Not applicable for partition files, worksheets, or outlines.

unlock
Unlock an artifact that is locked by another user or process. Not applicable for alias tables and worksheets. Unlocking an artifact of type `lro` is applicable for stored linked-reporting objects only; that is, files with the `.LRO` extension.

> **Note:**
>
> To unlock all database artifacts, use `alter database DBS-NAME unlock all objects;`.

copy to
Make a copy of a server artifact. Not applicable for partition files, worksheets, or outlines. If an artifact of the new name already exists, it is replaced.

force copy to
Make a copy of a server artifact. Not applicable for partition files, worksheets, or outlines. If an artifact of the new name already exists, it is replaced. If an administrator issues the statement with the **force** keyword, locked artifacts are unlocked, copied, and re-locked.

Notes

- Specified artifacts must be persisted in the database directory.
- Attempting to rename or copy an artifact of type "partition_file" returns an error.

Example

```
alter object sample.basic.genref of type rules_file rename to 'level';
```

Renames a rules file in the Sample.Basic directory, named `genref.rul`, to `level.rul`.

```
alter object sample.basic.Calcdat of type text rename to 'c_data';
```

Renames a text file in the Sample.Basic directory, named `calcdat.txt`, to `c_data.txt`.

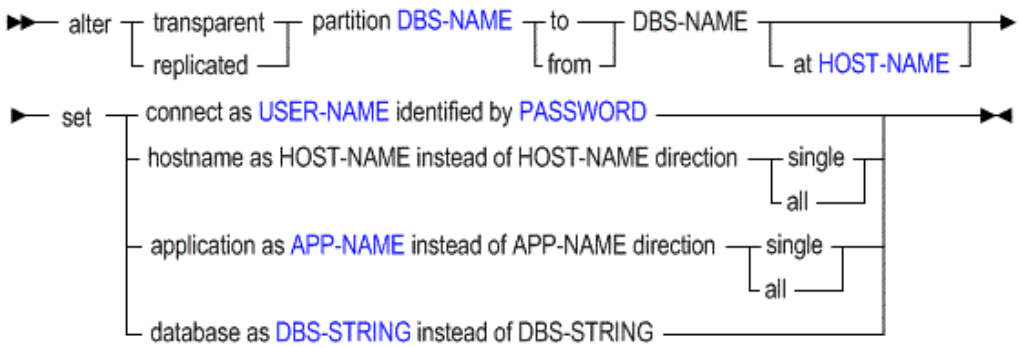
```
alter object samppart.company.company of type partition_file unlock;
```

Unlocks the partition definition file for the Samppart Company database.

Alter Partition

Fix invalid or dangling partition references. Change the authorized user who can connect to both cubes. Change the name of an application, cube, or host (in the event that something was renamed).

Syntax



- DBS-NAME
- HOST-NAME
- USER-NAME
- PASSWORD
- APP-NAME
- DBS-STRING

Use **alter partition** to edit partitions in the following ways:

Keywords

...set connect
Change the user authorized to access the partitioned cubes.

...set hostname
Edit the partition definition to include the correct URL for the partition source cube, target cube, or both.

...set application as
Edit the partition definition to include a corrected application name. This is useful if *one* application name was changed; if both application names changed, the partition definition cannot be corrected and you must re-create it.

...set database as
Edit the partition definition to include a corrected cube name. This is useful if *one* cube name was changed; if both cube names changed, the partition definition cannot be corrected and you must re-create it.

...direction single
See Examples 2, 4, and 5.

...direction all
See Example 3.

Notes

- The first DBS-NAME is the local cube, and the second DBS-NAME is the remote cube.
- Directing a partition *to* the remote site means the current cube is the source. Creating a partition *from* the remote site means the current cube is the target.
- To change the authorized partition user, you must change the user for both partitioned cubes, as shown in Example 1.
- If a partitioned host, application, or cube is renamed, the rename does not propagate to the partition definition, so you must use alter partition to change the name in the partition definition. As shown in Examples 2 through 5, you must give the old name and the new name. If both names were changed, the partition definition is not recoverable, and must be re-created.

Example

Example 1 (Alter Partition)

The following example changes the user authorized to access the partitioned cubes.

```
/* To change authorized partition user on target, log in to source &
then use: */
       alter transparent partition app1.source to app2.target
       set connect as newuser identified  by newpasswd;

/* To change authorized partition user on source, log in to target &
then use: */
       alter transparent partition app2.target from app1.source
       set connect as newuser identified  by newpasswd;
```

Example 2 (Alter Partition)

In the following example, `alter partition` is used to fix a partition definition that became invalid when a URL changed and affected only one half of the partition definition (app2.target):

```
alter transparent partition app1.source to app2.target at "https://
myEssbase-myDomain.analytics.us1.example.com/essbase/agent"
     set hostname as "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent" instead of "https://
myEssbase-myDomain.analytics.us1.example.com/essbase/agent" direction
single;
```

where `direction single` indicates that only the target URL needs to be changed.

Example 3 (Alter Partition)

In the following example, `alter partition` is used to fix a partition definition that became invalid when a host-name change affected both the source and the target, because both applications were on the same host:

```
alter transparent partition app1.source to app1.target at "https://
myEssbase-myDomain.analytics.us2.example.com/essbase/agent"
     set hostname as "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent" instead of "https://
myEssbase-myDomain.analytics.us1.example.com/essbase/agent" direction
all;
```

where `direction all` indicates that the host-name change needs to be made on both the target and source halves of the partition definition.

Example 4 (Alter Partition)

In the following example, `alter partition` is used to fix a partition definition that became invalid when the source application name (`oldAppName`) changed to `newAppName`, and affected only one half of the partition definition:

```
alter transparent partition newAppName.source to app2.target
    set application as newAppName instead of oldAppName direction
single;
```

where `direction single` indicates that only one half of the partition definition needs to be corrected.

> **Note:**
>
> The old application name can be discovered by issuing the `display partition` statement prior to correcting the partition definition.

Example 5 (Alter Partition)

In the following example, `alter partition` is used to fix a partition definition that became invalid when the source application name (`oldAppName`) changed to `newAppName`, and affected both halves of the partition definition because both partitioned cubes were on the same application:

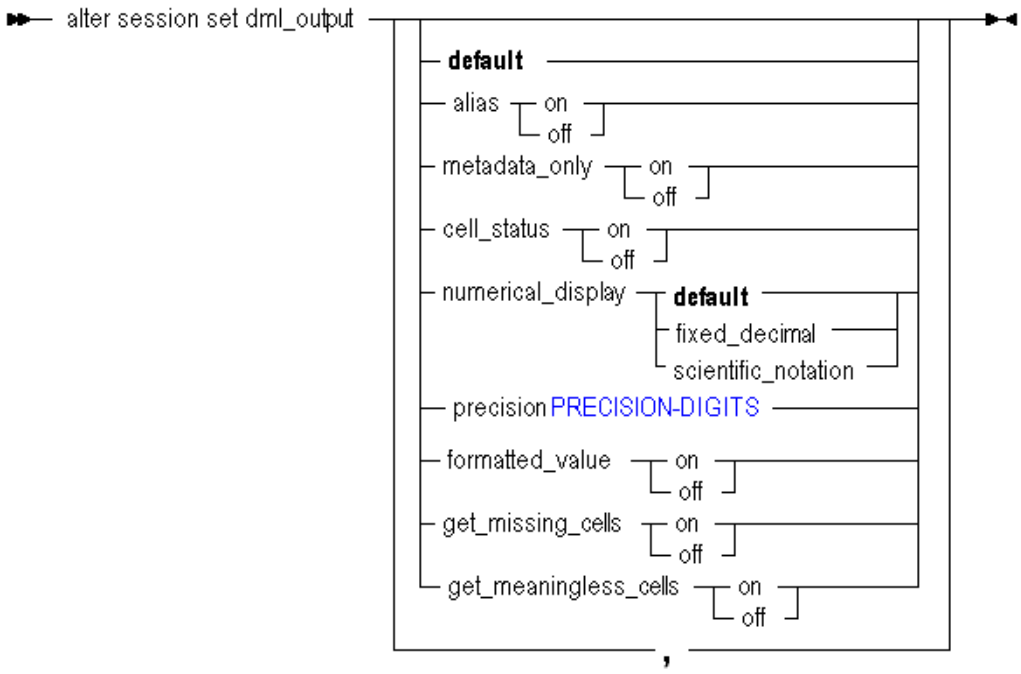
```
alter transparent partition newAppName.source to newAppName.target
     set application as newAppName instead of oldAppName direction all;
```

where `direction single` indicates both halves of the partition definition need to be corrected.

Alter Session

Set MDX display options.

Syntax



PRECISION-DIGITS

Use **alter session** to change the following MDX output settings:

Keywords

default
Revert to the default MDX display settings in the MaxL Shell. The default settings are: alias ON, metadata_only OFF, cell_status OFF.

alias on|off
Set whether to use aliases instead of member names.

metadata_only on|off
Set whether to show only the metadata, with no data.

cell_status on|off
Set whether to display cell status. Cell status is additional information returned with each cell value in MDX query outputs.

> **Note:**
>
> Every cell consists of one member from each dimension. Up to four cell-status types may be returned with the output:

- DC: Dynamic Calc. If any of the members defining the cell is Dynamic Calc, this status is on.
- RO: Read Only. If the cell cannot be written to (for example, by lock-and-send), this status is on. Security filters in the database might cause cells to be read-only. Dynamic Calc cells are automatically read-only.
- CM: Calculated Member. If any of the members defining the cell is a calculated member, this status is on.
- LO: Linked Object. If the cell has any associated Linked Reporting Objects, this status is on.

numerical_display fixed_decimal| scientific_notation|default
Set whether MaxL returns data values in MDX query output as fixed decimals, scientific notation, or default format (values are returned in a reasonable combination of decimals or scientific notation).

precision <precision-digits>
Set the number (0-15) of decimal places to include for the data values in MDX query output.

formatted_value on|off
Set whether to return formatted values for all cells of type text or date, or cells associated with a format string. By default, this setting is on.

get_missing_cells on|off
Set whether to return #Missing valued cells for all cells of type text or date, or cells associated with a format string. By default, this setting is on.

get_meaningless_cells on|off
Set whether to return #Meaningless for cells that are empty only because they are unassociated with the context attribute or varying attribute. By default, this setting is off, and the empty cells display as #Missing.
The following example query gets sales for all products, but the aggregation is specified by the slicer context only for Ounces_12.

```
SELECT
{Sales, Cogs}
```

```
ON COLUMNS,
 {Product.Levels(0).Members}
ON ROWS
FROM Sample.Basic
WHERE (Ounces_12)
;
```

A value of #Meaningless is displayed for any members not associated with the attribute Ounces_12.

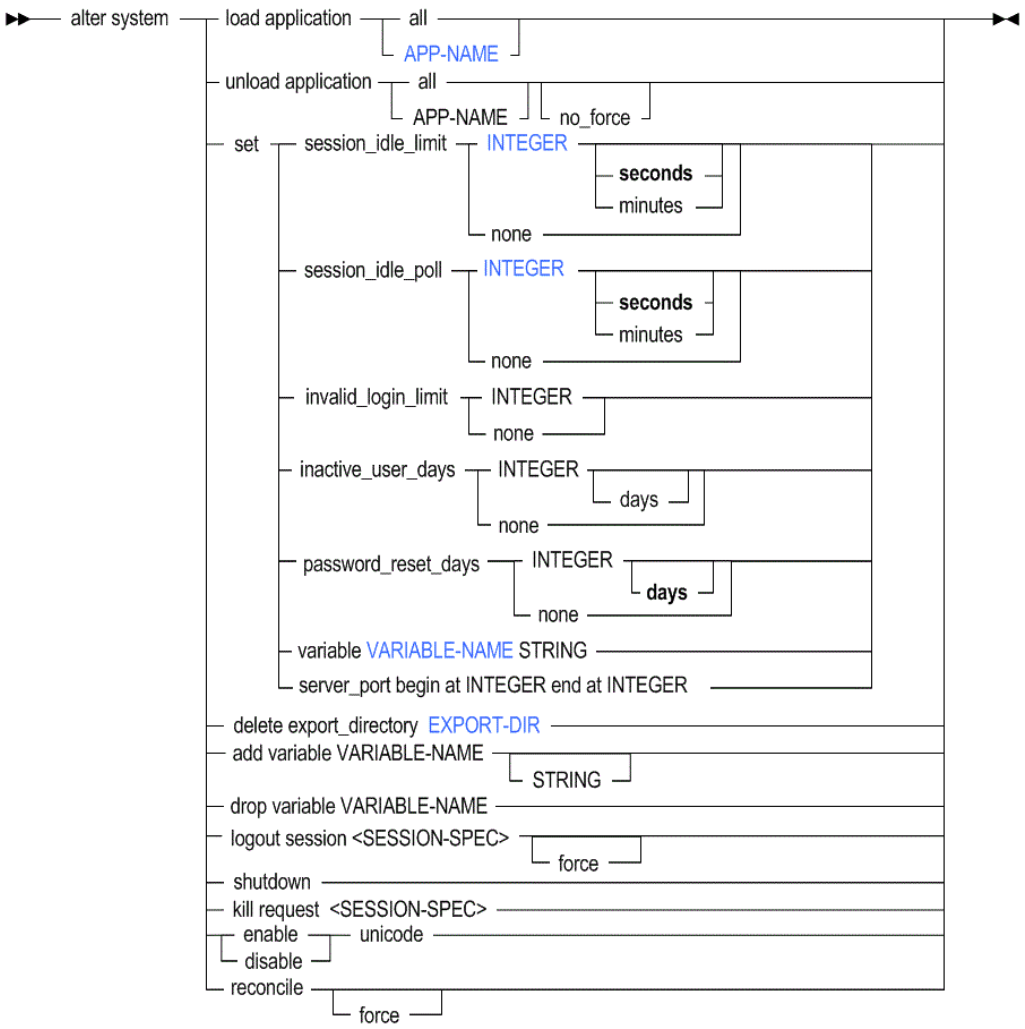
Alter System

Click here for aggregate storage version

Change the state of Essbase Server. Start and stop applications, change system-wide variables, manage password and login activity, disconnect users, end processes, or shut down the server.

Permission required: Administrator.

Syntax



- APP-NAME
- INTEGER
- VARIABLE-NAME
- EXPORT-DIR

Use **alter system** to change the following system-wide settings:

Keywords

load application
Start an application, or start all applications on the Essbase Server.

unload application
Stop an application, or stop all applications on the Essbase Server. Unloading an application cancels all active requests and database connections, and stops the application. If Essbase encounters a problem when trying to cancel active requests and database connections, and stopping the application, an error is logged in the application log.
If you do not want to stop an application if it has active requests and database connections, use the **no_force** grammar. When using **no_force**:

- If the application has active requests and database connections, the application is not stopped; it continues running
- If the application does not have active requests and database connections, the application is stopped, as if you used **unload application** without specifying **no_force**

set session_idle_limit
Set the interval of time permitted for a session to be inactive before Essbase Server logs off the user. The minimum limit that you can set is five minutes (or 300 seconds). When the session idle limit is set to **none**, all users can stay logged on until the Essbase Server is shut down.
The default user idle logout time is 60 minutes. When a user initiates a calculation in the background, after 60 minutes the user is considered idle and is logged out, but the calculation continues in the background.
Because the user may mistakenly assume that the calculation stopped because he or she was logged out, you can do one of the following to correct the user experience:

- Run the calculation in the foreground
- Increase the session idle limit in to a time that exceeds the duration of the calculation, or to none

set session_idle_poll
Set the time interval for inactivity checking and security-backup refreshing. The time interval specified in the session idle poll gives Essbase instructions:

- Tells it how often to check whether user sessions have passed the allowed inactivity interval indicated by `session_idle_limit` in the **alter system** statement.
- Tells it how often to refresh the security backup file. If `session_idle_poll` is set to zero, the security backup file is still refreshed every five minutes.

set invalid_login_limit
Set the number of unsuccessful login attempts allowed by any user before the system disables it. When you change this setting, the counter resets to 0. When the invalid login limit is set to **none**, there is no limit. By default, there is no limit.

set inactive_user_days
Set the number of days a user account may remain inactive before being disabled by the system. The counter resets when the user logs in, is edited, or is activated by an administrator. When the inactive days limit is set to **none**, user accounts remain enabled even if they are not used. By default, there is no limit.

set password_reset_days
Set the number of days users may retain passwords. After the allotted number of days, users are prompted at login to change their passwords. The counter resets for a user when the user changes the password, is edited, or is activated by an administrator. When the password reset days limit is set to **none**, there is no built-in limit for password retention. By default, there is no limit.

set variable
Change the value of an existing substitution variable on the system. The value must not exceed 256 bytes. It may contain any character except a leading ampersand (&).

set server_port
Expand a port range specified Essbase configuration properties. Each Essbase application uses two ports from this range. If no more ports are available, an error message is displayed.

> **Note:**
>
> You can expand port ranges only so that the beginning port range is less than SERVERPORTBEGIN and the ending port range is greater than SERVERPORTEND.

delete export_directory
Delete directories created for linked reporting objects exported from a database to a directory created in *ARBORPATH*`\app`. Use this grammar after the exported LROs are migrated into a database using **import lro**, and the directories containing the exported LRO information are not needed.

> **Note:**
>
> This process works only for directories created in *ARBORPATH*`\app` using the DBS-EXPORT-DIR option of the export lro statement. It does not work for directories created elsewhere using the FULL-EXPORT-DIR option of the **export lro** statement.

To view a list of names of exported linked-reporting-objects directories in *ARBORPATH*`\app`, use `display system export_directory`.

add variable
Create a system-level substitution variable by name, and optionally assign a string value for the variable to represent. You can assign or change the value later using **set variable**. A substitution variable acts as a global placeholder for information that changes regularly. Substitution variables may be referenced by calculations and report scripts.
If substitution variables with the same name exist at server, application, and database levels, the order of precedence for the variables is as follows: a database-level substitution variable supersedes an application-level variable, which supersedes a server-level variable.

drop variable
Remove a substitution variable and its corresponding value from the system.

logout session all
Terminate all user sessions running on the Essbase Server.

logout session...force
Terminate a session (or sessions) even if it is processing a request. The request is allowed to proceed to a safe point, and then the transaction is rolled back.

logout session <session-id>
Terminate a session by its unique session ID number. To see the session ID number, use display session.

logout session by user
Terminate all current sessions by a particular user, either across the entire Essbase Server, or limited to a specific application or database.

logout session by user on application
Terminate all current sessions by a particular user across a specific application.

logout session by user on database
Terminate all current sessions by a particular user across a specific database.

logout session on application
Terminate all current user sessions across a specific application.

logout session on database
Terminate all current user sessions across a specific database.

shutdown
Shut down the Essbase Server.

kill request all
Terminate all current requests on the Essbase Server.

> **Note:**
>
> To terminate your own active request in MaxL Shell, press the ESC key.

kill request <session-id>
Terminate the current request indicated by the session ID. You can obtain session IDs using display session.

kill request by user
Terminate all current requests by the specified user on the Essbase Server.

kill request on application
Terminate all current requests on the specified application.

kill request on database
Terminate all current requests on the specified database.

enable unicode
Set the Essbase Server to allow the creation of Unicode-mode applications and the migration of non-Unicode-mode applications to Unicode-mode applications.

disable unicode
Prevent the Essbase Server from allowing the creation of Unicode-mode applications or the migration of non-Unicode-mode applications to Unicode-mode applications.

reconcile
When Essbase is started using a security backup file (`essbase_timestamp.bak`) instead of `essbase.sec`, reconcile the security file to match the state of Essbase on an external disk. This grammar displays discrepancies in application and database information between the security file and the external disk:

- If an application folder is on the disk but not in the security file, display a message indicating the discrepancy. (Essbase checks for the presence of a *appname*`.app` file in the *ARBORPATH*`/app/`*appname* directory.)

  The *force* option does not apply in this scenario.

- If an application file is in the security file but not on the disk, display a message indicating the discrepancy.

  The *force* option removes the application from the security file.

- If an application database folder is on the disk but not in the security file, display a message indicating the discrepancy. (Essbase checks for the presence of a *dbname*`.otl` file in the *ARBORPATH*`app/`*appname*`/`*dbname*`/cube` directory.)

  The *force* option does not apply in this scenario.

- If an application database file is in the security file but not on the disk, display a message indicating the discrepancy.

  The *force* option removes the database from the security file.

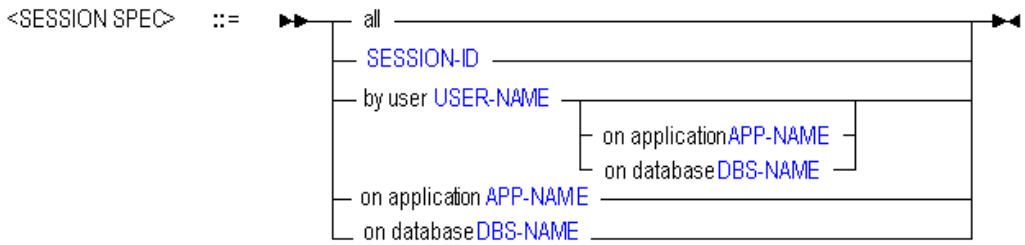
Notes

SESSION SPECIFICATION

A *session* is a single user connection to Essbase Server. The session can be identified by keywords and names indicating context, or by a unique session ID number.

A *request* is a query sent to Essbase Server by a user or by another process; for example, starting an application or restructuring a database outline. Only one request at a time can be processed in each session.

If a session is processing a request at the time that an administrator attempts to terminate the session, the administrator must either terminate the request first, or use the **force** kewyord available with **alter system** to terminate the session *and* the current request.



- SESSION-ID
- USER-NAME
- APP-NAME
- DBS-NAME

Example

```
alter system unload application Sample;
```

Stops the Sample application, if it is currently running.

```
alter system unload application all;
```

Terminates all active requests and stops all applications.

```
alter system unload application Sample no_force;
```

Essbase prepares to unload the Sample application; however, if active requests are running, the application is not stopped.

```
alter system shutdown;
```

Stops all running applications and shuts down Essbase Server.

```
alter system logout session by user Fiona;
```

Disconnects Fiona from any applications or databases to which she is connected.

To log out a user, log out the sessions owned by that user.

```
alter system set password_reset_days 10;
```

Specifies that all users will be prompted after 10 days to change their passwords. The day count for any user is reset when the user changes the password or is edited or reactivated by an administrator.

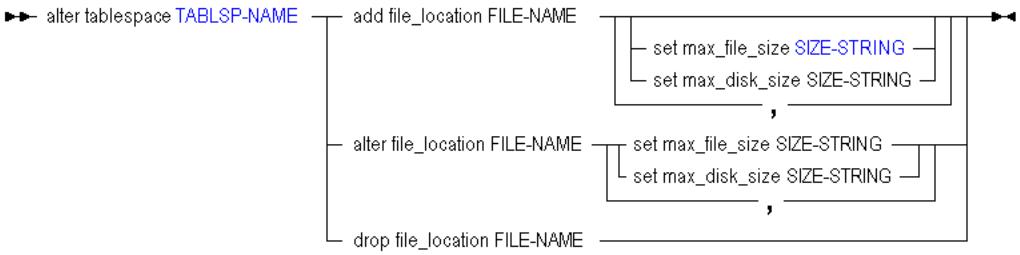
Alter Tablespace (Aggregate Storage)

Change details about a tablespace. To see a list of tablespaces, use display tablespace. You cannot change the location or size of the `metadata` and `log` tablespaces.

Tablespaces are applicable only to aggregate storage databases.

Permission required: Application Manager.

Syntax



- TABLSP-NAME
- SIZE-STRING

Use **alter tablespace** to edit tablespaces in the following ways:

Keywords

add file_location
Add a new file location to the tablespace.

> **Note:**
>
> FILE-NAME is case sensitive in this statement.

alter file_location
Change the maximum file-size or disk-size value for the specified file location.

> **Note:**
>
> FILE-NAME is case sensitive in this statement.

set max_file_size
Specify a value for the maximum size that a data file may attain before Essbase creates a new file.
The largest possible value that the aggregate storage kernel can handle is 134217727 MB. This is also the default value. If operating system limits take effect before this value is reached, the kernel creates a new file. If you enter a value that is larger than 134217727 MB, the kernel ignores the setting and caps file size at 134217727 MB.
The minimum value is 8MB (8388608b), and any values you enter are rounded up to the next 8MB interval.

> **Note:**
>
> Some operating system platforms may enforce a maximum file size limit.

set max_disk_size
Specify the value for the maximum amount of disk space to be allocated to the file location.
The largest possible value that the aggregate storage kernel can handle is 4294967295 MB. This is also the default value. If operating system limits take effect before this value is reached, the kernel attempts to use another file location in the tablespace. If you enter a value that is larger than 4294967295 MB, the kernel ignores the setting and caps disk size at 4294967295 MB.
The minimum value is 8MB (8388608b), and any values you enter are rounded up to the next 8MB interval.

drop file_location
Delete the specified file location from the tablespace. When a file location is deleted, all files in the file location are deleted, as well as the subdirectory containing the files. You cannot delete a file location if it contains data. You cannot delete the tablespace itself.

> **Note:**
>
> FILE-NAME is case sensitive in this statement.

Notes

- This statement requires the application to be started.
- On Windows, you can specify tablespace file locations using Uniform Naming Convention (UNC) syntax, which is `\\ComputerName\SharedFolder\Resource`. Including the escape characters required by MaxL Shell, the UNC file name specification would look like the following:

  ```
  '\\\\ComputerName\\SharedFolder\\Resource'
  ```

Example

```
alter tablespace ASOsamp.'default' add file_location 'C:\\mytablespace'
set max_file_size 50mb;
```

Adds another file location for the `default` tablespace. Now the tablespace `default` is in `C:\mytablespace` in addition to the original location, `C:\Hyperion\products\Essbase\EssbaseServer\app`.

```
alter tablespace ASOsamp.'default' alter file_location 'C:\\Hyperion\
\products\\Essbase\\EssbaseServer\\' set max_file_size 50mb;
```

Changes the maximum file size allowed in the specified location of the default tablespace. Note that the file_location string is case sensitive.

```
alter tablespace ASOsamp.'default' alter file_location '\\\
\ComputerName\\SharedFolder\\Resource' set max_file_size 50mb;
```

Changes the maximum file size allowed in the specified location of the default tablespace. The file_location string is specified using UNC.

Alter Trigger

Enable or disable a trigger created to track state changes over a selected cube area.

Syntax

- TRIGGER-NAME
- DBS-NAME

Use **alter trigger** to edit triggers in the following ways:

Keywords

enable
Essbase monitors the trigger during data load, calculation, or lock and send, and performs the trigger action when the condition is met on the specified cube area.

disable
Essbase does not monitor the trigger.

on database <DBS-NAME> disable
Disables all triggers in the database. A restart of the application or the database following the disable restores the triggers to the same state as before the disable

was issued (all the triggers disabled using `alter trigger on database DBS-NAME disable` are re-enabled).

Example

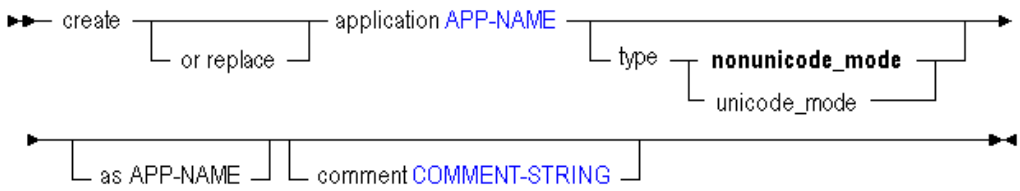
```
alter trigger Sample.Basic.WatchCosts disable;


alter trigger on database sample.basic disable;
```

Create Application

Click here for aggregate storage version

Create or re-create an application, either from scratch or as a copy of another application on the same system. See APP-NAME for information on the maximum length of and special characters that are allowed in an application name. Application names are not case-sensitive.

Syntax



- APP-NAME
- COMMENT-STRING

Use **create application** to create an application in the following ways:

Keywords

create application
Create a new application. Application names are not case-sensitive.

create or replace application
Create an application, or replace an existing application of the same name. Application names are not case-sensitive.

...type nonunicode_mode
Create a Non Unicode-mode application. This is also the default if these keywords are omitted.

...type unicode_mode
Create a Unicode-mode application.

create application as
Create an application as a copy of another application. Application names are not case-sensitive.

comment
Create an application description (optional). The description can contain up to 80 characters.

Example

```
create application Sample comment 'This is a test application.';
```

Creates a new application called Sample with an associated comment.

```
create application Newsamp as Sample;
```

Creates an application called Newsamp which is a copy of the application Sample.

```
create or replace application Sample;
```

Creates an application called Sample. If an application named Sample already exists, it is overwritten.

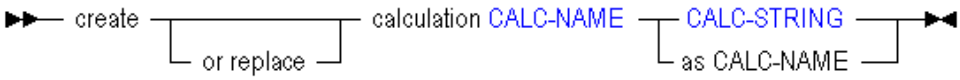
Create Calculation

Create, replace, or copy a stored calculation.

Permissions required:

- Database Manager to create database-level calculations.
- Application Manager to create application-level calculations.

Syntax



- CALC-NAME
- CALC-STRING

Use **create calculation** to create a calculation in the following ways:

Keywords

create calculation
Create a calculation script, the body of which is specified by CALC-STRING.

create or replace calculation
Create a calculation script, the body of which is specified by CALC-STRING. If a calculation script of that name already exists, it is replaced.

create calculation as
Create a calculation as a copy of another stored calculation.

Notes

- When creating database-level calculations, this statement requires the database to be started.
- A stored calculation can be associated with an application/database, or with an application only. To create an application-level calculation, use two tokens for CALC-NAME. To create a database-level calculation, use three tokens. See CALC-NAME for more details.
- Calculations created using MaxL must be valid. For information about calculation syntax, see *Designing and Maintaining Essbase Cubes*.

Example

```
create or replace calculation sample.basic.Accts
'SET UPDATECALC ON;
CALC DIM(Accounts);'
;
```

Creates a calculation named Accts that is associated with sample.basic.

```
create calculation sample.basic.Accts2 as app.db.Accts
```

Creates a calculation named Accts2 on sample.basic that is a copy of another database's calculation named Accts.

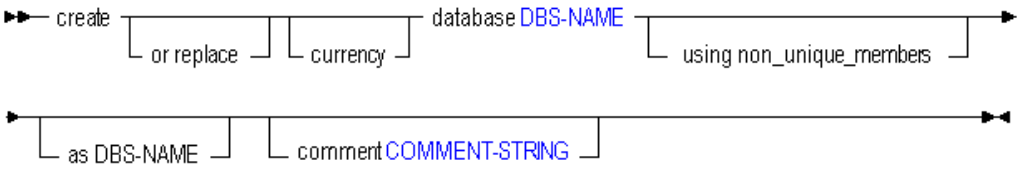
Create Database

Click here for aggregate storage version

Create or re-create a database. Optionally create the database as a copy of another database on the same system. See DBS-NAME for information on the maximum length of and special characters that are allowed in a database name. Database names are not case-sensitive.

Permission required: Application Manager. To copy a database, Manager permission on the source database is additionally required.

Syntax



- DBS-NAME
- COMMENT-STRING

Use **create database** to create a database in the following ways:

Keywords

create database
Create a new database. Database names are not case-sensitive.

create or replace database
Create a database, or replace an existing database of the same name. Database names are not case-sensitive.

create database using non_unique_members
Create a database that supports the use of duplicate member names. Once you have created a database with a duplicate member outline, you cannot convert it back to a unique member outline.
For more information about duplicate member names, see Creating and Working With Duplicate Member Outlines.

create database as
Create a database as a copy of another database. Database names are not case-sensitive.

create currency database
Create or replace a database for currency conversion. Linking a currency database to a main database enables you to convert currency values in a database from one currency into another currency.

comment
Create a database description (optional). The description can contain up to 80 characters.

Example

```
create or replace database Sample.Basic comment 'This is a test.';
```

Creates a database called Basic within the Sample application. If a database named Basic within the Sample application already exists, it is overwritten.

```
create database Sample.New as Sample.Basic;
```

Creates a database called New within the Sample application that is a copy of the database Basic within the Sample application.

```
create currency database Sample.Interntl;
```

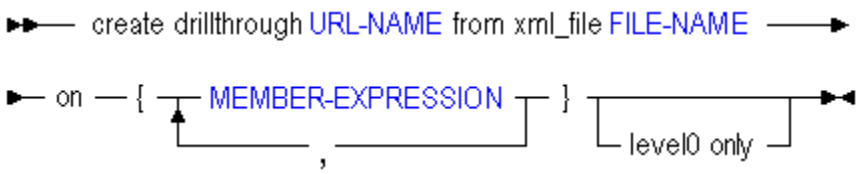
Creates a currency database called Interntl within the Sample application.

Create Drillthrough

Create a drill-through URL within the active cube outline.

For each drillable region of a cube, you can enable drill-through access by means of a URL to Web content hosted on Oracle ERP and EPM applications.

Syntax



- URL-NAME
- FILE-NAME
- MEMBER-EXPRESSION

Use **create drillthrough** to create a drill-through URL definition in the following ways:

Keywords

create drillthrough
Create a drill-through URL as metadata.
The number of drill-through URLs per database is limited to 255.

from xml_file
Indicate the path to the local URL XML file that defines the link information.
The URL XML is created by the ERP or EPM application that deployed the cube. The XML contains the drill-through URL display name and a URL enabling the hyperlink from a cell to a Web interface to occur.
The following is a sample URL XML file:

```
<?xml version="1.0" encoding="UTF-8"?>
<foldercontents path="/">
  <resource name="Assets Drill through GL" description=""
type="application/x-hyperion-applicationbuilder-report">
    <name xml:lang="fr">Rapport de ventes</name>
    <name xml:lang="es">Informe de ventas</name>
```

```
    <action name="Display HTML" description="Launch HTML display of
Content" shortdesc="HTML">
      <url>/fusionapp/
Assetsdrill.jsp?$SSO_TOKEN$&$CONTEXT$&$ATTR(ds,pos,gen,level.edge)$
      </url>
    </action>
  </resource>
</foldercontents>
```

on {<member-expression>,...}
Define the list of drillable regions, using the same member-set calculation language that is used to define security filters. The list of drillable regions must be enclosed in {brackets}.
The number of drillable regions in a drill-through URL is limited to 256. The number of characters per drillable region is limited to 65536.

level0 only
Optional: Restrict the URL definition to level-0 data.

Example

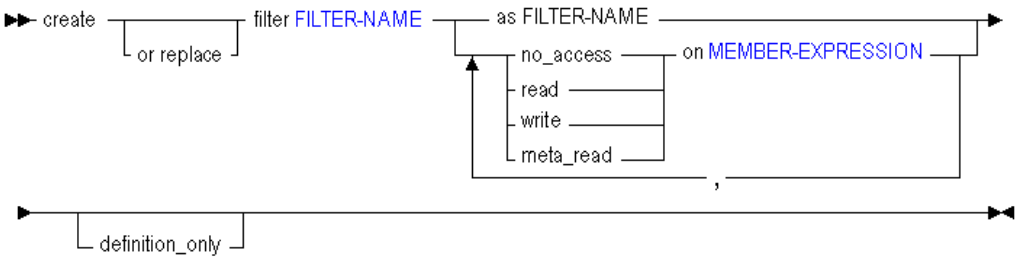
```
create drillthrough sample.basic.myURL from xml_file "myfile1.xml" on
{'@Ichildren(“Qtr1”)', '@Ichildren(“Qtr2”)'} level0 only;
```

Create Filter

Create or re-create a database security filter, either from scratch or as a copy of another filter on the same system. Filters control security for database objects. Use grant to assign filters to users and groups.

Minimum permission required: Database Manager.

Syntax



- FILTER-NAME
- MEMBER-EXPRESSION

Use **create filter** to create a filter in the following ways:

Keywords

create filter
Create a security filter to restrict or permit access to specified database cells.

create or replace filter
Create a security filter or replace an existing security filter of the same name.

create filter ... no_access on <member-expression>
Create a filter blocking access to a specified member combination.

create filter ... read on <member-expression>
Create a filter providing read-only access to a specified member combination.

create filter ... write on <member-expression>
Create a filter providing write access to a specified member combination.

create filter ... meta_read on <member-expression>
Create a filter restricting access to siblings and ancestors of the member expression. In case of a filtering conflict, the MetaRead filtering overrides the other filter permissions. For more information about metatdata filtering, see Metadata Filtering.

create or replace filter ... definition_only;
Updates the filter definition while retaining user associations with the filter. If you replace a filter without using `definition_only`, then the filter must be re-granted to any users to whom it was assigned.

Notes

The member expression must be enclosed in single quotation marks. It can be a comma-separated list.

Example

```
create filter sample.basic.filt1 read on 'Jan, sales', no_access on
'@CHILDREN(Qtr2)';
```

Creates a filter to restrict privileges to Sample.Basic as follows: gives read-only access to the intersection of Jan and sales (sales data for January only); blocks access to children of Qtr2 (April, May, and June).

```
create or replace filter sample.basic.filt1 read on 'Sales,
@ATTRIBUTE(Bottle)';
```

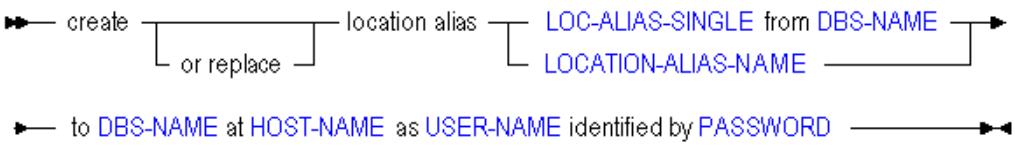
Creates a filter (or changes an existing filter) to restrict privileges to Sample.Basic as follows: gives read-only access to sales data for products packaged in a bottle (product base dimension members associated with the Bottle attribute member).

Create Location Alias

Create on the database a location alias identifying a host name, database, user name, and password. Location aliases provide a shorthand way of referencing login information for other cubes.

Minimum permission required: Database Manager.

Syntax



- LOC-ALIAS-SINGLE
- LOCATION-ALIAS-NAME
- DBS-NAME
- HOST-NAME
- USER-NAME
- PASSWORD

Use **create location alias** to create a location alias in the following ways:

Keywords

create location alias
Create a location alias, identifying a remote host name, database, user name, and password. The location alias can be used by the @XREF function as an abbreviated login to a remote database.

create or replace location alias
Create a location alias, replacing any existing location alias of the same name on the same database.

...from <dbs-name>
Specify the name of the current database (the database on which the location alias is being created).

...to <dbs-name>
Specify the name of the remote database to log in to.

...at <host-name>
Specify where the remote database resides (using discovery URL).

...as <user-name> identified by <password>
Specify a user name and password with which to log in to the remote database.

Notes

- This statement requires the database to be started.
- Location aliases created using MaxL must be valid.
- Location aliases are used by the @XREF calculation function for cross-database calculations.

Example

```
create location alias EasternDB from Sample.Basic to East.Sales at
"https://myEssbase-myDomain.analytics.us2.example.com/essbase/agent" as
smith identified by 'password';
```

Creates a location alias called EasternDB on Sample.Basic that represents the following login information:

- remote host = https://myEssbase-myDomain.analytics.us2.example.com/essbase/agent
- application = East
- database = Sales
- user name = Fiona
- password = sunflower

Create Partition

Create or validate a partition definition between two databases.

Permission required: Database Manager at both sites.

Select the type of partition to create:

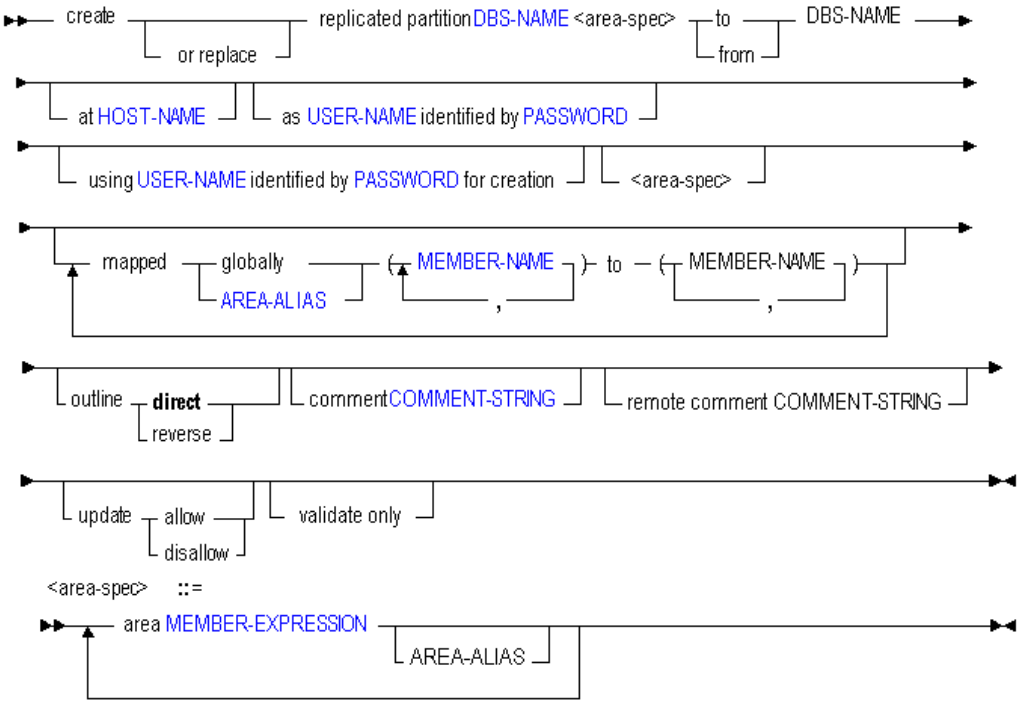
- transparent
- replicated

Partitions created using MaxL must be valid. To validate a partition, use the **validate only** clause. For information about partition definitions, see *Designing and Maintaining Essbase Cubes*.

Create Replicated Partition

Create or validate a replicated partition definition between two cubes. A replicated partition copies a portion of the source (or master) cube to be stored in a target cube. Users can access the target cube as if it were the source. The administrator must periodically refresh the target data from the source data.

Syntax



- DBS-NAME
- HOST-NAME
- USER-NAME
- PASSWORD
- AREA-ALIAS
- MEMBER-NAME
- COMMENT-STRING
- MEMBER-EXPRESSION

Use **create replicated partition** to create a partition in the following ways:

Keywords

create replicated partition
Create a replicated partition. A replicated partition is a copy of a portion of the data source that is stored in the data target.

create or replace ...partition
Create a partition definition, or replace an existing partition definition.

area...
Define the partition areas to share with the other cube. Optionally nickname the area using an area-alias .

to <dbs-name>
Create a partition definition between the current source cube and the second cube (the target).

from <dbs-name>
Create a partition definition between the current target cube and the second cube (the source).

at <host-name>
Specify the discovery URL of the remote cube.

as <user-name> identified by <password>
Provide the name and password of a default partition user who can connect to both cubes. Essbase uses the login information to:

- Transfer data between the source and the target for replicated and transparent partitions. Security filters can be applied to prevent end users from seeing privileged data.
- Synchronize outlines for all partition types.

using <user-name> identified by <password> for creation
Create the partition using a different user than the one being set as the default partition user. This can be useful when you want to specify a read-only user account as the default partition user.

mapped...
Define the member-name mapping for shared sections of both cubes, if member names for sections that map are different in the two cubes.

outline...
Specify the direction in which outline synchronization should proceed, if necessary. The default direction is the same as the data-refresh direction.

update...
Allow or disallow the updating of data in a replicated-type partition target. If you do not specify update allow, by default, the replicated partition cannot be updated.

comment
Create a comment to describe the source half of the partition definition.

remote comment
Create a comment to describe the target half of the partition definition.

validate only
Validate the existing partition definition described by this statement, without actually creating it.

Notes

- Multiple area specifications are allowed, provided they are separated by whitespace. Multiple mappings are allowed, provided they are separated by whitespace. All area aliases used in a mapping should be associated with the target, and the direction of the mapped clause should go from source to target.
- The first DBS-NAME is the local cube, and the second DBS-NAME is the remote cube.
- Creating a partition *to*the remote site means the current cube is the source. Creating a partition *from*the remote site means the current cube is the target.
- Aggregate storage cubes can be the target, but not the source, of a replicated partition.

Example

```
create or replace replicated partition source.source
area 'DimensionA' sourceAreaA
area 'DimensionB' sourceAreaB
to target.target at "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent"
as admin identified by 'password'
area 'ParentMemberA' targetAreaA
area 'ParentMemberB' targetAreaB
mapped targetAreaA (ChildA) to (Child_a)
mapped targetAreaB (ChildB) to (Child_b)
;
```

Creates a partition from cube Source to cube Target where the partitioned areas between them are DimensionA and DimensionB on the source, corresponding to ParentMemberA and ParentMemberB (respectively) on the target. Differences in member names between the two partitioned areas are resolved during the partition creation, using the *mapped* clauses. Area aliases are used after each area specification, so that members can be mapped specifically for each area.

```
create or replace replicated partition sampeast.east
area '@IDESCENDANTS("Eastern Region"), @IDESCENDANTS(Qtr1)'
to samppart.company at "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent"
as partitionuser identified by 'password'
area '@IDESCENDANTS(East) @IDESCENDANTS(Qtr1)'
update disallow;
```

Creates a replicated partition from an area in the source cube, sampeast.east, to an area in the target cube, samppart.company.

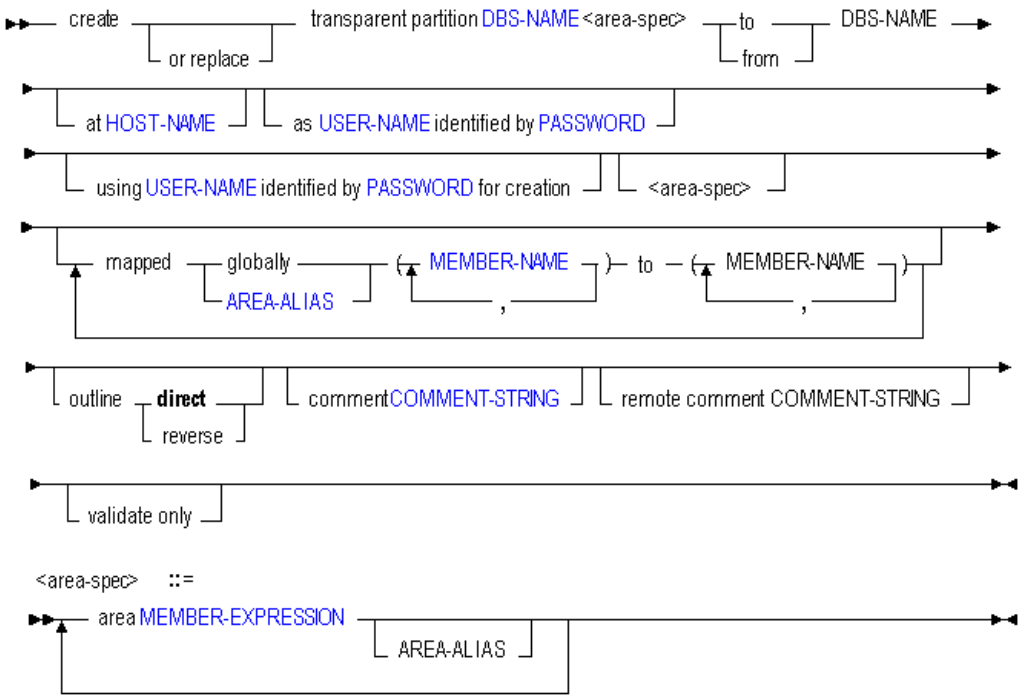
```
create or replace replicated partition sampeast.east
area '@IDESCENDANTS("Eastern Region"), @IDESCENDANTS(Qtr1)'
to samppart.company at "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent"
as admin identified by 'password'
area '@IDESCENDANTS(East) @IDESCENDANTS(Qtr1)' foo
mapped foo (Year) to (Yr)
update allow validate only;
```

Validates the syntax of a replicated partition you might want to create. To create the partition after checking validity, simply remove the *validate only* phrase. For an explanation of *foo* as used above, see the definition for AREA-ALIAS .

Create Transparent Partition

Create or validate a transparent partition definition between two cubes. A transparent partition allows users to manipulate data that is stored in a target cube as if it were part of the source cube. The remote data is retrieved from the data source each time that users at the data target request it.

Syntax



- DBS-NAME
- HOST-NAME
- USER-NAME
- PASSWORD
- AREA-ALIAS
- MEMBER-NAME
- COMMENT-STRING
- MEMBER-EXPRESSION

Use **create transparent partition** to create a partition in the following ways:

Keywords

create transparent partition
Create a transparent partition. A transparent partition enables users to access data from the data source as though it were stored in the data target. The data is, however,

stored at the data source, which can be in another application, in another cube, or on another Essbase instance.

create or replace ...partition
Create a partition definition, or replace an existing partition definition.

area...
Define the partition areas to share with the other database. Optionally nickname the area using an area-alias.

to <dbs-name>
Create a partition definition between the current source cube and the second cube (the target).

from <dbs-name>
Create a partition definition between the current target cube and the second cube (the source).

at <host-name>
Specify the discovery URL of the remote cube.

as <user-name> identified by <password>
Provide the name and password of a default partition user who can connect to both cubes. Essbase uses the login information to:

- Transfer data between the source and the target for replicated and transparent partitions. Security filters can be applied to prevent end users from seeing privileged data.
- Synchronize outlines for all partition types.

using <user-name> identified by <password> for creation
Create the partition using a different user than the one being set as the default partition user. This can be useful when you want to specify a read-only user account as the default partition user.

mapped...
Define the member-name mapping for shared sections of both cubes, if member names for sections that map are different in the two cubes.

outline...
Specify the direction in which outline synchronization should proceed, if necessary. The default direction is the same as the data-refresh direction.

comment
Create a comment to describe the source half of the partition definition.

remote comment
Create a comment to describe the target half of the partition definition.

validate only
Validate the existing partition definition described by this statement, without actually creating it.

Notes

- Multiple area specifications are allowed, provided they are separated by whitespace. Multiple mappings are allowed, provided they are separated by

whitespace. All area aliases used in a mapping should be associated with the target, and the direction of the mapped clause should go from source to target.

- The first DBS-NAME is the local cube, and the second DBS-NAME is the remote cube.
- Creating a partition *to* the remote site means the current cube is the source. Creating a partition *from* the remote site means the current cube is the target.
- Aggregate storage cubes can be the source, the target, or the source and target of a transparent partition. Outline synchronization (**refresh outline** statement) is not currently enabled for partitions that involve aggregate storage cubes.

Example

```
create or replace transparent partition sampeast.east
        area '@CHILDREN("Eastern Region"), @CHILDREN(Qtr1)' sourceArea
to samppart.company at "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent"
as partitionuser identified by 'password'
        area '@CHILDREN(East) @CHILDREN(Qtr1)' targetArea;
```

Creates a transparent partition between the source, sampeast.east, and the target, samppart.company. The partition is defined only for the areas specified by the area aliases `sourceArea` and `targetArea`.

```
create or replace transparent partition source.source
        area 'DimensionA' sourceAreaA
        area 'DimensionB' sourceAreaB
to target.target at "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent"
as smith identified by 'password'
        area 'ParentMemberA' targetAreaA
        area 'ParentMemberB' targetAreaB
        mapped targetAreaA (ChildA) to (Child_a)
        mapped targetAreaB (ChildB) to (Child_b)
        ;
```

Creates a partition from cube Source to cube Target where the partitioned areas between them are DimensionA and DimensionB on the source, corresponding to ParentMemberA and ParentMemberB (respectively) on the target. Differences in member names between the two partitioned areas are resolved during the partition creation, using the *mapped* clauses. Area aliases are used after each area specification, so that members can be mapped specifically for each area.

Create Trigger

Create or replace a trigger to track state changes over a selected cube area.

Select the type of trigger to create:

- on-update
- after-update

Create After-Update Trigger

Create or replace a trigger to track state changes over a selected cube area.

Triggers help you track whether designated constraints are violated during updates (events) in the area, and allow you to specify resultant actions to execute if violations are detected.

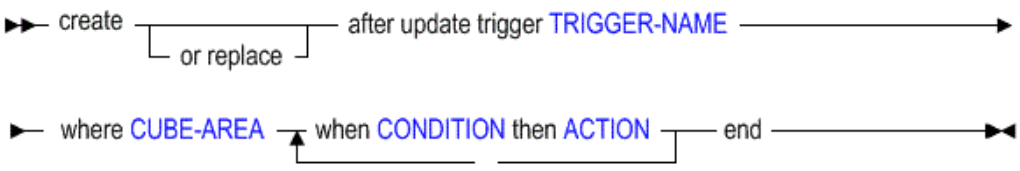
Minimum permission required: Database Manager.

Create an *after-update* trigger if you want the trigger to be activated after the entire data update operation is completed. This is the only type of trigger supported in aggregate storage mode. When after-update triggers are used, the trigger fires when an update operation on level-0 data cells is complete, and the update operation as a whole has met any condition specified for the cube area.

> **Note:**
>
> You cannot create or replace a trigger during a calculation, data update, or data load.

> **Note:**
>
> If a calculation assigns the same value to a given cell as was already present before the calculation, then triggers for that cell will not activate. In other words, if cell values are not changed, blocks are not marked as dirty, and triggers for those blocks are not activated, even if the trigger condition was otherwise met.

Syntax



- TRIGGER-NAME
- CUBE-AREA
- CONDITION
- ACTION

Use **create after update trigger** to create a trigger in the following ways:

Keywords

create after update trigger
Create a new after-update trigger.

create or replace after update trigger
Create an after-update trigger, or replace an existing trigger of the same name.

where <cube area>
Define the area of the database to be tracked. Use a valid, symmetric MDX slicer specification.

when <condition>
Define the condition to be tested for using the keyword WHEN followed by a valid MDX conditional expression.

then <action>
Define the action to be taken if the WHEN condition is met. See examples in Examples of Triggers.

end
The END keyword must terminate every create trigger statement.

Example

```
create or replace after update trigger Sample.Basic.EastColas
where (Jan, Sales, Actual, [100], East)
when Jan > 20 then spool EastColas_Fail end;
```

Logs a message in the EastColas_Fail file in the database directory.

Create On-Update Trigger

Create or replace an on-update trigger to track state changes over a selected cube area.

Triggers help you track whether designated constraints are violated during updates (events) in the area, and allow you to specify resultant actions to execute if violations are detected.

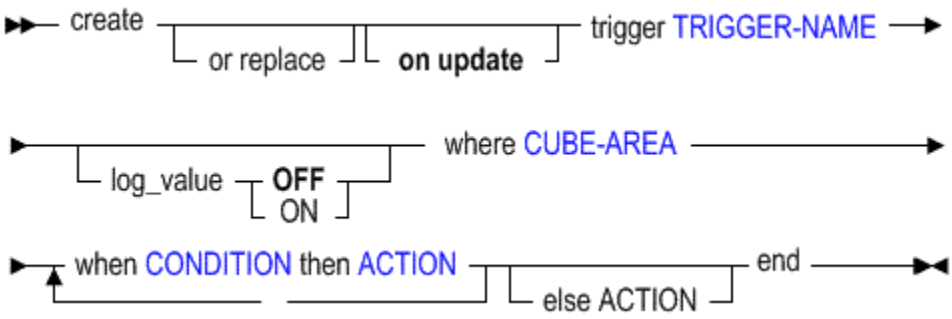
Minimum permission required: Database Manager.

An *on-update* trigger is the default type of trigger, even if no type is specified. During a data update process, any cell update that meets a condition specified for the cube area will immediately activate the trigger. On-update triggers are not supported in aggregate storage databases. If you are using an aggregate storage database, you can create after-update triggers.

> **Note:**
>
> If a calculation assigns the same value to a given cell as was already present before the calculation, then triggers for that cell will not activate. In other words, if cell values are not changed, blocks are not marked as dirty, and triggers for those blocks are not activated, even if the trigger condition was otherwise met.

> **Note:**
>
> You cannot create or replace a trigger during a calculation, data update, or data load.

Syntax



- TRIGGER-NAME
- CUBE-AREA
- CONDITION
- ACTION

Use **create on update trigger** to create a trigger in the following ways:

Keywords

create [on update] trigger
Create a new on-update trigger. The **on update** keywords are optional; an on-update trigger is created by default.

create or replace [on update] trigger
Create an on-update trigger, or replace an existing trigger of the same name.

log_value OFF
Optional. Log no data values to the trigger spool file. This is the default.

log_value ON
Optional. Log new and old data values to the trigger spool file.

where <cube area>
Define the area of the database to be tracked. Use a valid, symmetric MDX slicer specification.

when <condition>
Define the condition to be tested for using the keyword WHEN followed by a valid MDX conditional expression.

then <action>
Define the action to be taken if the WHEN condition is met. See examples in Examples of Triggers.

else <action>
Optional. Define an action to be taken if the WHEN condition is *not* met. See examples in Examples of Triggers.

end
The END keyword must terminate every create trigger statement.

Example

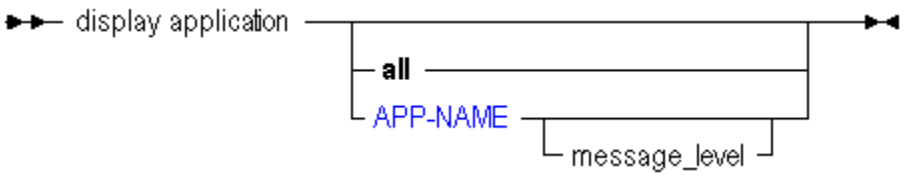
```
create or replace on update trigger Sample.Basic.EastColas
where (Jan, Sales, Actual, [100], East)
when Jan > 20 then spool EastColas_Fail end;
```

Logs a message in the EastColas_Fail file in the database directory.

Display Application

View information about current application-wide settings.

Syntax



APP-NAME

Use **display application** to display application information in the following ways:

Keywords

all
Display all applications on the system.

<app-name>
Display the named application.

<app-name> message_level
Display the message-level settings for the named application.
Sample output:

```
component           message_level
+-------------------+-------------------+
Sample              info
```

Output Columns

application
String. Name of the application.

comment
String. Optional description of the application.

startup
TRUE or FALSE. Whether all users who have at least read permission can start the application.

autostartup
TRUE or FALSE. Whether the application starts when Essbase Server starts.

minimum permission
String. Minimum level of permission all users can have to databases in the application.

connects
TRUE or FALSE. Whether any user with a permission lower than Application Manager can make connections to the databases in this application which would require the databases to be started.

commands
TRUE or FALSE. Whether users with sufficient permissions can make read requests (or higher) to databases in the application.

updates
TRUE or FALSE. Whether users with sufficient permissions can make write requests (or higher) to databases in the application.

security
TRUE or FALSE. If FALSE, the Essbase security settings are disabled for the application, and all users are treated as Application Managers.

lock_timeout
Number. Maximum time interval (in seconds) that locks on data blocks can be held by clients.

max_lro_file_size
Number. If 0, there is no limit on the size of LRO attachments. All other sizes are displayed in kilobytes.

application_type
The type of encoding for the application.

```
0       Unspecified encoding type. The application was created using a
pre-Release 7.0 version of Essbase.
1       This value is not in use.
2       Non-Unicode-mode application
3       Unicode-mode application
```

application_locale
The language of the character set in use by the application.

server
The name of the computer hosting the Essbase Server.

application_status

```
0       Not Loaded
1       Loading
2       Loaded
3       Unloading
```

elapsed_time
How long the application has been loaded.

users_connected
The number of users currently connected to the application.

storage_type
The data storage type of the application.

```
0       Default data storage (same as 1)
1       Block storage (multidimensional)
4       Aggregate storage
```

number_of_databases
The number of databases in the application namespace.

Example

```
display application;
```

Displays information about all applications on the system.

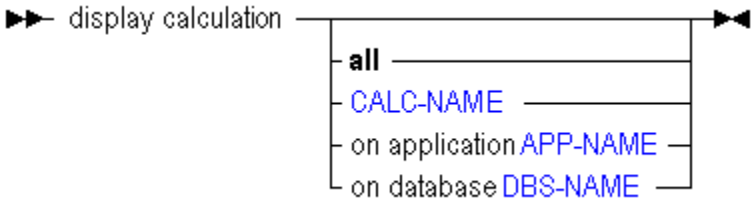
```
display application Sample;
```

Displays information about the Sample application.

Display Calculation

View a list of stored calculations on the system.

Syntax



- CALC-NAME
- DBS-NAME
- APP-NAME

Use **display calculation** to display calculations in the following ways:

Keywords

all
Display all stored calculations on the system.

<calc-name>
Display the named calculation.

on application
Display all calculations on the specified application.

on database
Display all calculations on the specified database.

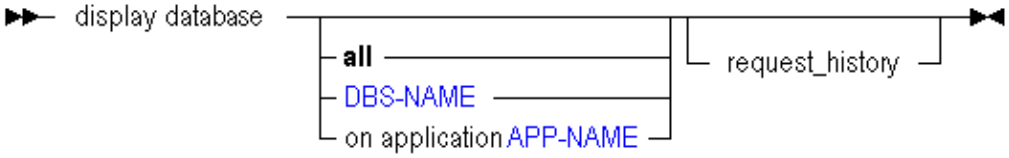
Example

```
display calculation;
```

Display Database

View information about current database-wide state and settings.

Syntax



- DBS-NAME
- APP-NAME

Use **display database** to display database information in the following ways:

Keywords

all
Display information for all databases on the system.

<dbs-name>
Display information about the specified database.

on application
Display information about all databases on the specified application.

request_history
Display information about recent requests for the database. Information about the last three requests is returned.

Output Columns

application
Name of the application

database
Name of the database

comment
Text of the database comment, if present

startup
Whether the database is set to start when a user attempts retrievals against it

autostartup
Whether the database is set to start when the application starts

minimum permission
Minimum permission setting for the database.

aggregate_missing
Whether Essbase aggregates missing values during database calculations

two_pass_calc
Whether Two-Pass calculation is enabled

create_blocks
Whether create blocks on equations is enabled

data_cache_size
The size setting of the data cache for holding uncompressed data blocks

file_cache_size
The size setting of the file cache

index_cache_size
The size setting of the index cache, a buffer in memory that holds index pages

index_page_size
The size setting for the index page, a subdivision of an index file that contains index entries that point to data blocks. This setting is not changeable

cache_pinning
Whether cache memory locking is enabled (no longer supported)

compression
Compression type. Field values are numeric, and translate as follows:

```
1               Run-length encoding
2               Bitmap
3               (no longer supported)
```

retrieve_buffer_size
The size of the retrieval buffer, used to process and optimize retrievals from grid clients

retrieve_sort_buffer_size
The size of the retrieval sort buffer, used to hold data to be sorted during retrievals

io_access_mode
The current I/O access mode. Only buffered I/O is supported.

pending_io_access_mode
Values are numeric, and translate as follows:

```
0               Invalid / Error
1               Buffered
2               Direct /* no longer supported
```

no_wait
Whether Essbase is set to wait to acquire a lock on data blocks that are locked by another transaction

committed_mode
Whether Essbase is set to enable transactions to hold read/write locks on all data blocks involved with a transaction until the transaction completes and commits

pre_image_access
Whether Essbase is set to allow users read-only access to data blocks that are locked for the duration of another concurrent transaction

lock_timeout
The maximum number of minutes that data blocks can be locked by users

commit_blocks
The number of data blocks updated before Essbase performs a commit (The default is 3000)

commit_rows
The number of rows of a data file processed during a data load before Essbase performs a commit (The default is 0)

currency_database
Name of a linked currency database, if one exists

currency_member
The member to use as a default value in currency conversions

currency_conversion
The method of currency conversion.
Values are numeric, and translate as follows:

```
1               division
2               multiplication
```

note
Annotation accessible from the login dialog box

db_type
Database type. Values are numeric, and translate as follows:

```
0               Normal
1               Currency /* no longer supported
```

read_only_mode
Values are numeric, and translate as follows:

```
0               Not read only
1               Read only
```

db_status
Running status of the database. Values are numeric, and translate as follows:

```
0               Not Loaded
1               Loading
2               Loaded
3               Unloading
```

elapsed_time
How long the database has been running, in hours:minutes:seconds

users_connected
Number of connected users

blocks_locked
How many data blocks are locked

number_dimensions
Number of dimensions

number_disk_volume
Number of disk volumes

data_status
Values are numeric, and translate as follows:

```
0               No Data
1               Data Loaded without Calculation
2               Data is Calculated
```

current_data_cache
Current size of the data cache

current_file_cache
Current size of the file cache

current_index_cache
Current size of the index cache

current_index_page
Current size of the index page

currency_country_dim
For currency databases, the country dimension

currency_time_dim
For currency databases, the time dimension

currency_category_dim
For currency databases, the accounts dimension where currency categories are defined

currency_type_dim
For currency databases, the currency type dimension, which contains members that identify various currency scenarios

request_type_n / request_user_n / request_start_n / request_end_n
If you use the **request_history** keyword, information about the last three requests is returned under columns *request_type_n*, *request_user_n*, *request_start_n*, and *request_end_n*, where *n* is 1, 2, and 3. The *request_user* fields return the names of the users who made the requests. The *request_start* and *request_end* fields return the date and time of the requests.
request_type field values are numeric, and translate as follows:

```
0               Data Load
1               Calculation
2               Outline Update
                Unknown
```

Example

```
display database;
```

Displays information about all databases on the system.

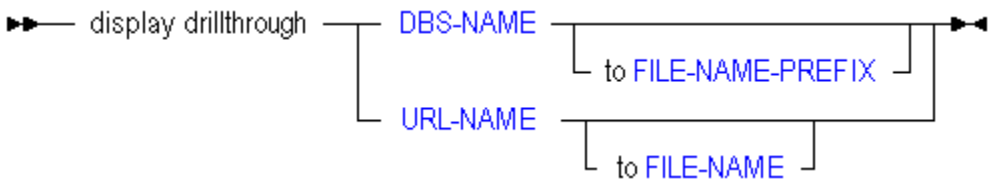
```
display database Sample.Basic;
```

Displays information about the Sample.Basic database.

Display Drillthrough

View drill-through URL definitions used to link to content hosted on Oracle ERP and EPM applications.

Syntax



- DBS-NAME
- FILE-NAME-PREFIX
- URL-NAME
- FILE-NAME

Use **display drillthrough** to display URL information in the following ways:

Keywords

<dbs-name>
Display all drill-through URL definitions on the database.
The number of drill-through URLs per database is limited to 255.

<dbs-name> to <file-name-prefix>
Display all drill-through URL definitions on the database, writing the URL XML content to file names prefixed with the string given as input for FILE-NAME-PREFIX.

<url-name>
Display the specified drill-through URL definition.
The number of drillable regions in a drill-through URL is limited to 256. The number of characters per drillable region is limited to 65536.

<url-name> to <file-name>
Display the specified drill-through URL definition, writing the URL XML content to the specified file name.

Example

```
display drillthrough sample.basic;
```

Displays all drill-through URL definitions on Sample.Basic.

```
display drillthrough sample.basic to "urlxmls";
```

Displays all drill-through URL definitions on Sample.Basic, writing the URL XML content to file names prefixed with `urlxmls`.

```
display drillthrough sample.basic."Drill through To EPMI";
```

Displays the drill-through URL definition named `Drill through To EPMI`.

```
display drillthrough sample.basic."Drill through To EPMI" to "c:/temp/
drillthrough.xml";
```

Displays the drill-through URL definition named `Drill through To EPMI`, writing the URL XML content to the file `drillthrough.xml`.

Display Filter

View a specific filter or a list of all filters on the system.

Syntax

- FILTER-NAME
- DBS-NAME

Use **display filter** to display filters in the following ways. Use display filter row to display the contents of filters.

Keywords

all
Display all filters on the system.

<filter-name>
Display a filter by name.

on database
Display all filters associated with the specified database.

Example

```
display filter;
```

Displays the names of all filters on the system.

Display Filter Row

View the filter rows which define database access within a specific filter or all filters.

Syntax

- FILTER-NAME
- DBS-NAME

You can display filter contents in the following ways using **display filter row**.

Keywords

all
Display all filters (and their contents) defined on the system.

<filter-name>
Display a filter and its contents by name.

on database
Display all filters (and their contents) associated with the specified database.

Example

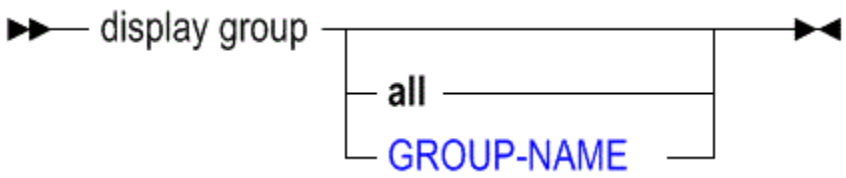
```
display filter row sample.basic.filt2;
```

Displays the row-by-row definition of a filter named filt2 which is associated with Sample.Basic.

Display Group

View a specific group or a list of all groups on the system. To view group membership information, use display user.

Syntax



GROUP-NAME

Use **display group** to display groups in the following ways:

Keywords

all
Display all security groups on the system.

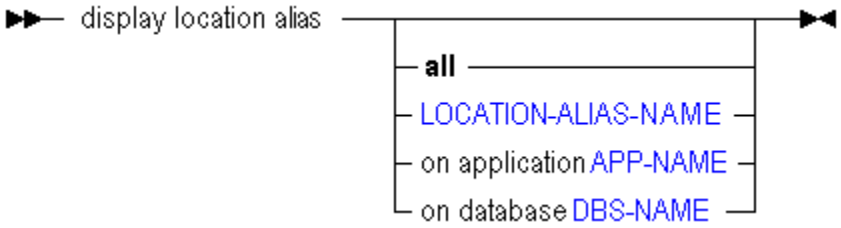
> **Note:**
>
> This MaxL grammar is deprecated. Oracle recommends using Java API or Oracle Enterprise Manager to get a list of all groups.

<group-name>
Display a security group by name.

Display Location Alias

View a specific location alias or a list of all location aliases defined on the system.

Syntax



- LOCATION-ALIAS-NAME
- APP-NAME
- DBS-NAME

You can display location aliases in the following ways using **display location alias**.

Keywords

all
Display all location aliases defined on the system.

<location-alias-name>
Display a location alias by name.

on application
Display all location aliases defined for the specified application.

on database
Display all location aliases defined for the specified database.

Example

```
display location alias all;
```

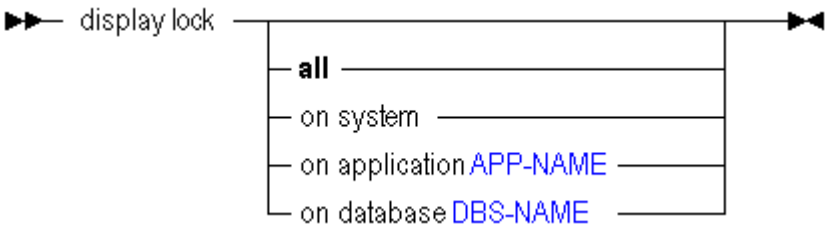
Displays a list of location aliases defined on the system.

Display Lock

View information about locks currently held by users or processes on data blocks.

> **Note:**
>
> Data locks do not apply to aggregate storage applications.

Syntax



- APP-NAME
- DBS-NAME

You can display locks in the following ways using **display lock**.

Keywords

all
Display all locks on the specified scope. If **all** is omitted, this is the default.

on system
Display all locks on the system.

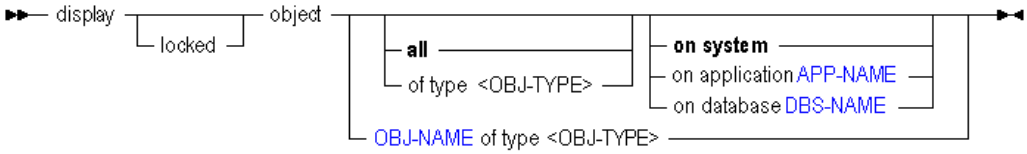
on application
Display all locks associated with the specified application.

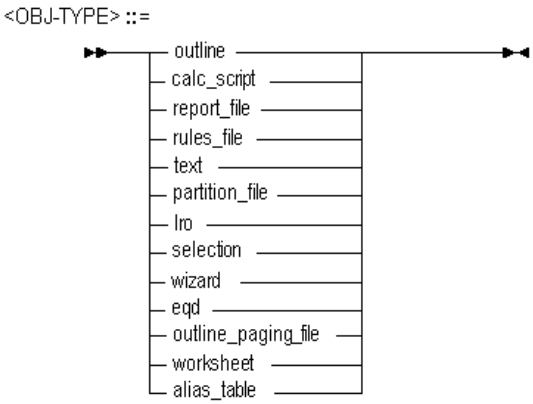
on database
Display all locks associated with the specified database.

Display Object

View a list of database-related file objects stored in database directories.

Syntax





- APP-NAME
- DBS-NAME
- OBJ-NAME

You can display objects in the following ways using **display object**.

Keywords

all
Display all stored objects on the specified scope.

locked
Display only locked objects on the specified scope.

of type...
Display only the objects of type specified by `OBJ-TYPE ::=`.

OBJ-NAME of OBJ-TYPE
Display a specific object by name and type.

on system
Display all stored objects on the system.

on application
Display all objects associated with the specified application.

on database
Display all objects associated with the specified database.

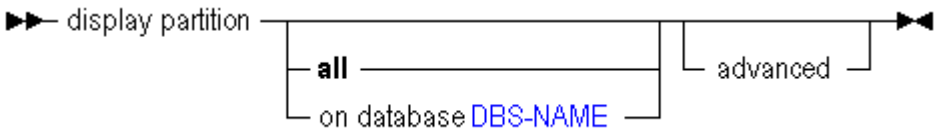
Example

```
MAXL> display object sample.basic.Calcdat of type text;

 applicati database  object_na object_ty locked      locked_by
locked_time
+---------+---------+---------+---------+-----------+-----------
+-----------
 Sample    Basic     Calcdat           9       FALSE N/A         N/A
```

Display Partition

View information about a specific partitioned database or all partitioned databases on the system. Only displays partition information for applications which are currently started.

Syntax



DBS-NAME

You can display partition information in the following ways using **display partition**.

Keywords

all
Display all partitions defined on the system.

on database
Display all partitions associated with the specified database.

advanced
Display full information including areas and member mappings for local and remote pieces of partitions.

Notes

If a partition definition is invalid, the same partition may be displayed twice, one time for each half. Each half will show the connection information of the other half.

Example

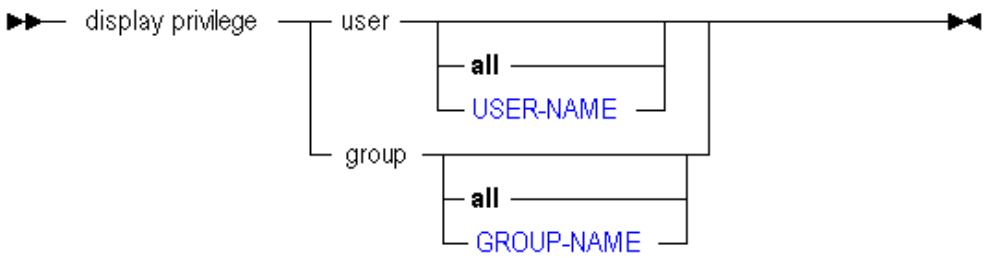
```
display partition all;
```

Displays information about all partitioned databases defined on the system.

Display Privilege

View a list of privileges, calculations, or filters held by users or groups.

Syntax



- USER-NAME
- GROUP-NAME

You can display security permissions in the following ways using **display privilege**.

Keywords

user...
Display security permissions for all users, or for a specified user.

group...
Display security permissions for all groups, or for a specified group.

The values returned for the *type* field are numeric, and translate as follows:

Table 3-7 Output Columns

Column	Description
1	System-level system privileges (no longer supported in MaxL)
2	System-level system roles (no longer supported in MaxL)

Table 3-7 (Cont.) Output Columns

Column	Description
3	Execute calculation
4	Filter

Example

```
display privilege user Fiona;
```

Displays the privileges user Fiona has on each database object, including any calculations or filters granted to Fiona.

```
display privilege group;
```

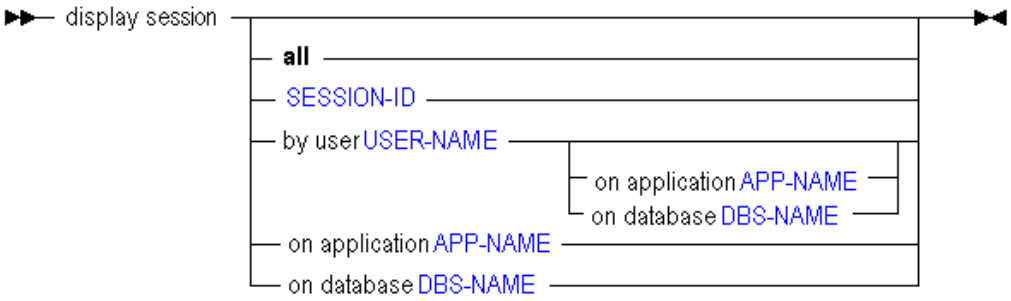
Displays privileges held by all groups on the system to all applications and databases on the system.

Display Session

View active login sessions on the current server, application, or database, including:

- The user that owns each session
- A session ID for each session
- How long the sessions have been active
- Information about outstanding requests (description, time started, name of computer originating the request, and status).

Syntax



- APP-NAME
- DBS-NAME
- USER-NAME
- SESSION-ID

You can display login and request information in the following ways using **display session**.

Keywords

all
Display information about all current user sessions and active requests.

<session-id>
Display information about a particular user session, indicated by the numeric session ID.

by user
Display information about all current sessions by a particular user.

by user on application
Display information about all current sessions by a particular user on the specified application.

by user on database
Display information about all current sessions by a particular user on the specified database.

on application
Display information about all current sessions on the specified application.

on database
Display information about all current sessions on the specified database.

Table 3-8 Display Session: Output Columns

Column	Description	Example
user	Logged in user name	`powerusr`
session	Numeric session id	`865075202`
login_time	Number of seconds ago the session began	`192`
application	Name of active application	`Sample`
database	Name of active database	`Basic`
db_connect_time	Number of seconds ago the database was set active	`11879`
request	Type of active request in progress; for example, calculation, data load, or restructure. This information can help you get details about what is occurring during lengthy sessions.	`BuildDimXml : Index Only`
request_time	Number of milliseconds the active request has been running	`1503869494621`
connection_source	Host name of the connected service	`example.com`

Table 3-8 (Cont.) Display Session: Output Columns

Column	Description	Example
connection_ip	IP address of the connected service	`192.0.2.123`
request_state	The status of the active request	`in_progress`

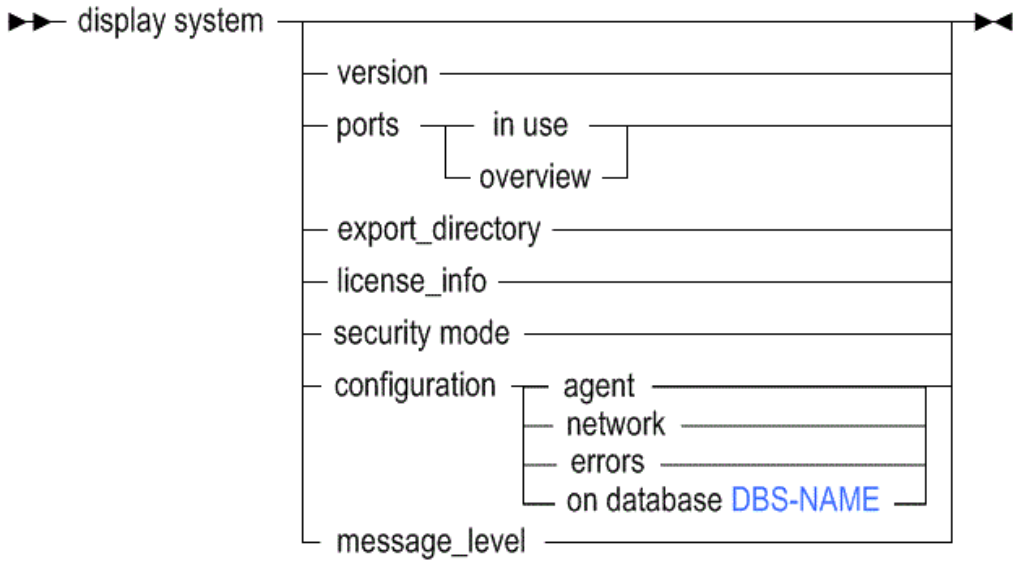
Example

```
display session;


display session on database sample.basic;
```

Display System

View information about current system-wide settings.

Syntax



DBS-NAME

You can display server-wide information in the following ways using **display system**.

Keywords

display system
Display current connections and system-wide settings.

configuration field values are numeric, and translate as follows:

```
1  Non-Unicode mode
2  Unicode mode
```

display system version
Display the server software version number.

display system ports in use
Display information about ports currently in use on the system.

display system ports overview
Display the number of ports that are available and in use on the system.

display system export_directory
Display names of directories created for linked-reporting objects exported from a database to a directory created in `$ARBORPATH\app`.
If you used **export lro** and gave a full path to a directory for export files, those directories are not listed. Only export directories created in the `ARBORPATH\App` directory using the following **export lro** method are listed:
`export database DBS-NAME lro to <server or local> directory DBS-EXPORT-DIR;`
where DBS-EXPORT-DIR is a suffix (for example, `dir1`) for the name of a directory created by MaxL in `$ARBORPATH\App`. MaxL creates the directory with a prefix of `appname-dbsname-`. For example, **display system export_directory** would list the following directories existing under `$ARBORPATH\App`:

```
sample-basic-dir1


sample-basic-dir2
```

but it would not list export directories created elsewhere by providing a full directory path when using the **export lro** statement, such as:`c:\MyExports\MyExportDir`

display system license_info
Display information about the license settings implemented on the system.

display system security mode
The type of security in use: native or OPSS mode.
security_mode field values are numeric, and translate as follows:

```
1  Native Essbase security (no longer supported)
2  OPSS security
```

display system configuration agent
Display current Essbase Agent configuration properties.
Permission required: Administrator.

display system configuration network
Display current Essbase configuration properties applicable to the network.
Permission required: Administrator.

display system configuration errors
Display Essbase configuration properties that contain errors: an error is any line entry that is not a comment *and* results in nothing being set.
Permission required: Administrator.

display system configuration on database DBS-NAME
Display Essbase configuration properties applicable to the named database.
Permission required: Administrator.

message_level
Display the values that are set for the system message level.
Sample output:

```
component           message_level
+-------------------+-------------------+
system              info
```

Example

```
display system;
```

Displays current password and session management settings.

```
display system configuration agent;
```

Displays current Essbase configuration properties applicable to the Essbase Agent.

Sample Outputs for Display System Configuration

```
MAXL> set column_width 40;

MAXL> display system configuration agent;
 KEYWORDS                          SETTINGS
+---------------------------------+----------------
 AGENTTHREADS                     50
 MAXLOGINS                        100000
 PORTUSAGELOGINTERVAL             600

 OK/INFO - 1241044 - Records returned: [3].
MAXL> display system configuration network;

 KEYWORDS                                  SETTINGS
+----------------------------------------
+----------------------------------------
 NETDELAY                                  1500
 NETRETRYCOUNT                             2000

 OK/INFO - 1241044 - Records returned: [2].
```

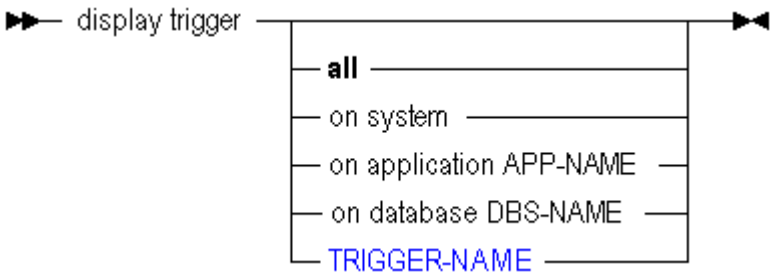
```
MAXL> display system configuration on database democfg.basic;

 KEYWORDS                                SETTINGS
+---------------------------------------
+---------------------------------------
 CALCCACHE                               TRUE
 CALCCACHEDEFAULT                        1250000
 CALCCACHEHIGH                           1750000
 CALCCACHELOW                            40000
 DLSINGLETHREADPERSTAGE                  FALSE
 DLTHREADSPREPARE                        4
 DLTHREADSWRITE                          4
 DYNCALCCACHEMAXSIZE                     DB[41943040], SV[41943040]
 SSPROCROWLIMIT                          250000

 OK/INFO - 1241044 - Records returned: [9].
```

Display Trigger

View details about a trigger created to track state changes over a selected cube area.

> **Note:**
>
> The application containing the trigger must be started in order to use display trigger.

Syntax



APP-NAME

Table 3-9 Output Columns

Column	Description
application	The name of the application that contains the database.
database	The name of the database that contains the trigger. Essbase lists only databases that contain triggers.

Table 3-9 (Cont.) Output Columns

Column	Description
name	The name of the trigger.
definition	The MaxL trigger statement (for example, create or replace trigger)
enabled	Whether Essbase is set to monitor the trigger. Values: TRUE or FALSE. To change the value, use **alter trigger**.

Example

```
display trigger on database Sample.Basic;
```

This example displays the output columns:

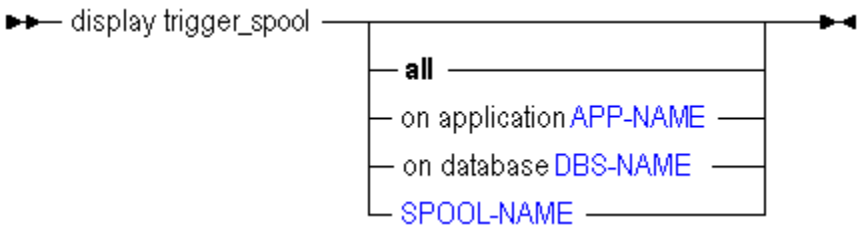
Table 3-10 Display Trigger MaxL Output

application	database	name	definition	enabled
Sample	Basic	WatchCosts	create or replace trigger	TRUE

Display Trigger Spool

View the log file created by a trigger. Triggers track state changes over a selected cube area. For more information about triggers, see Examples of Triggers.

Syntax

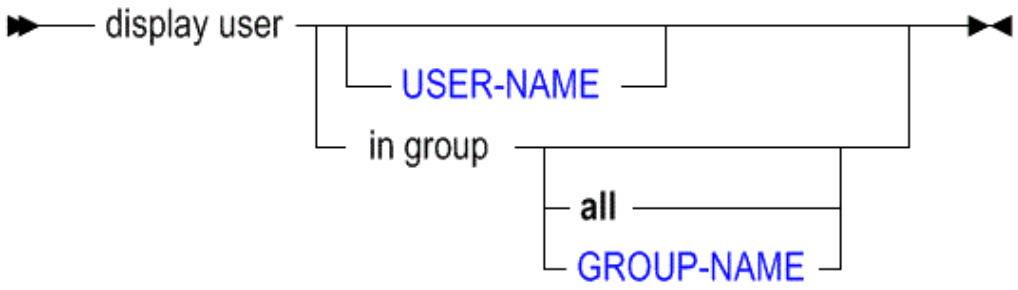


- APP-NAME
- DBS-NAME
- SPOOL-NAME

Display User

View a specific user or a list of all users defined on the system. View account and group membership information.

Syntax



- USER-NAME
- GROUP-NAME

You can display user information in the following ways using **display user**.

Keywords

all
Display information about all users on the system.

> **Note:**
>
> This MaxL grammar is deprecated. Oracle recommends using Java API or Oracle Enterprise Manager to get a list of all users.

<user-name>
Display information about the specified user.

in group all
Display membership information for all groups on the system.

in group <group-name>
Display membership information for the specified group.

Output Columns

user
String. Name of the user.

description
No longer supported.

logged in
Values: TRUE or FALSE.

password_reset_days
Integer. The number of days before the password expires, or 0 if no expiration is set.

enabled
Values: TRUE if the user account is active, or FALSE if the account has been disabled by an administrator.

change_password
Values: TRUE if the user must change the password at the next login; FALSE otherwise.

type
Values:

```
0       User is set up using native Essbase security (no longer
supported)
1       No longer used.
3       User is externally authenticated.
```

protocol
If the user is externally authenticated, this field contains the value `OPSS`. This field is blank if the `type` field is 0 (the user is not externally authenticated).

conn param
This field is blank.

application_access_type
Values:

```
0       No access
1       Essbase access
2       Planning access
3       Essbase and Planning access     (requires 2 licenses)
```

See also Descriptions section.

Example

```
display user;
```

Displays all users on the system and shows whether they are logged in, whether their accounts are enabled, and whether their passwords are set to expire.

```
display user in group;
```

Displays the membership information of all groups on the system.

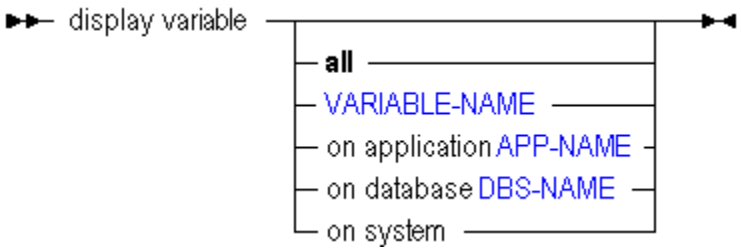
```
display user in group big_group;
```

Displays the membership information for a group called big_group.

Display Variable

View a list of substitution variables defined on the system.

Syntax



- VARIABLE-NAME
- APP-NAME
- DBS-NAME

You can display substitution variables in the following ways using **display variable**.

Keywords

all
Display all substitution variables defined on the Essbase Server, including those associated with applications and databases.

<variable-name>
Display a substitution variable by name.
Permission required:

- Read access for the applicable database or application.
- Administrator for system-defined variables.

on application
Display only substitution variables defined on the specified application.
Permission required: Read access for the application.

on database
Display only substitution variables defined on the specified database.
Permission required: Read access for the database.

on system
Display only the substitution variables associated with the Essbase Server.
Permission required: Administrator.

Notes

To manage substitution variables, use alter database (containing add, drop, and set variable).

Example

```
display variable;
```

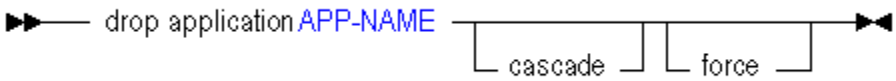
Displays a list of all substitution variables on the Essbase Server.

Drop Application

Delete an empty application from the system. To remove an application with databases, use **cascade**. To remove an application that has locked objects in a constituent database, you can use **force**.

Minimum permission required: Application Manager.

Syntax



APP-NAME

You can delete applications in the following ways using **drop application**.

Keywords

cascade
Delete an application along with its constituent databases.

force
Delete an application that may have locked objects in a constituent database.

Drop Calculation

Delete a stored calculation from a database.

Minimum permission required: Database Manager.

Syntax

You can delete calculations using **drop calculation**.

CALC-NAME

Keywords

drop calculation <calc-name>
Delete the specified calculation.

Example

```
drop calculation Sample.basic.calcname;
```

Deletes a calculation from Sample.basic.

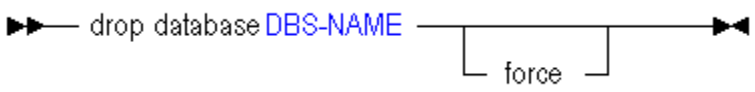
Drop Database

Delete a database from the sytsem. If the database has outstanding locks, clear them first, or use **force** to drop with locks.

Minimum permission required: Database Manager.

Syntax

You can delete databases using **drop database**.



DBS-NAME

Keywords

force
Delete a database that may have locked objects.

Example

```
drop database Sample.Basic force;
```

Deletes the database Sample.Basic, even if client users have outstanding locks on Sample.Basic.

Drop Drillthrough

Delete a drill-through URL definition used to link to content hosted on Oracle ERP and EPM applications.

Syntax

URL-NAME

Example

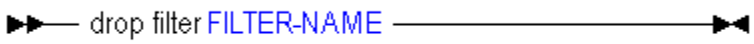
```
drop drillthrough sample.basic.myURL;
```

Drop Filter

Delete a security filter from the database.

Minimum permission required: Database Manager.

Syntax



FILTER-NAME

You can delete filters using **drop filter**.

Keywords

drop filter <filter-name>
Delete a filter by name.

Example

```
drop filter sample.basic.filter1;
```

Deletes the filter called filter1 from the sample.basic database.

Drop Location Alias

Delete from the database a location alias identifying a host name, application, database, user name, and password.

Minimum permission required: Database Manager.

Syntax

LOCATION-ALIAS-NAME

You can delete location aliases using **drop location alias**.

Keywords

drop location alias <location-alias-name>
Delete a location-alias definition.

Example

```
drop location alias Main.Sales.EasternDB;
```

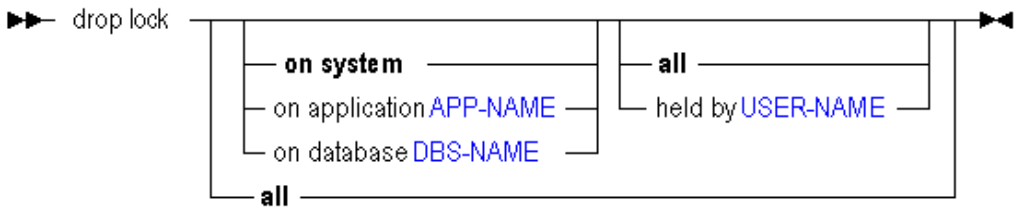
Drops the location alias called EasternDB in the Main.Sales database.

Drop Lock

Remove locks acquired through a grid client operation.

> **Note:**
>
> Data locks do not apply to aggregate storage applications.

Syntax



- APP-NAME
- USER-NAME
- DBS-NAME

Keywords

drop lock on system all
Drops all locks by all users, for all databases on the system.

drop lock all
Same as "drop lock on system all"

drop lock on system
Same as "drop lock on system all"

drop lock
Same as "drop lock on system all"

drop lock on application APP-NAME
Drops all locks on the application, for all users.

drop lock on application APP-NAME held by USER-NAME
Drops locks on the application which are held by a specific user.

drop lock on database DBS-NAME
Drops all locks on the database, for all users.

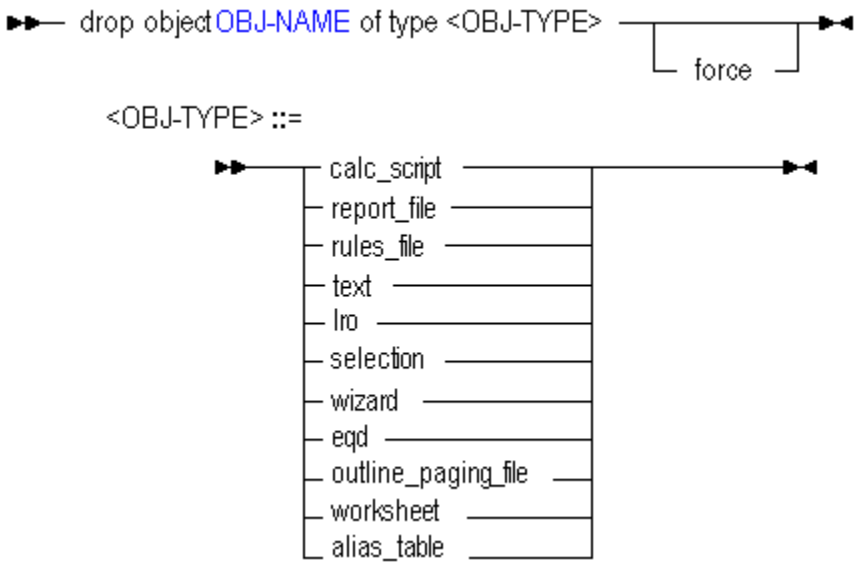
drop lock on database DBS-NAME held by USER-NAME
Drops locks on the database which are held by a specific user.

drop lock held by USER-NAME
Drops all locks held by a specific user, on any application or database.

Drop Object

Remove database-related file objects stored in database directories.

Syntax



OBJ-NAME

Keywords

...force
If the object is locked by a user or proecess, unlock it and delete it.

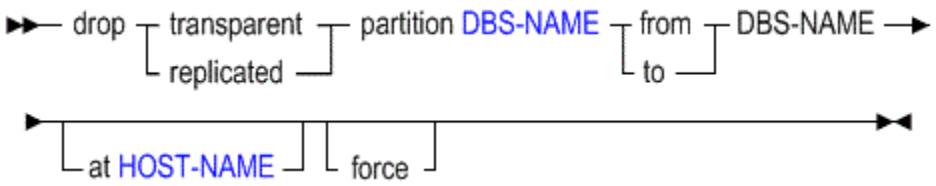
Notes

To drop a partition, use drop partition.

Drop Partition

Delete from the system a partition definition between two cubes. Database Manager permission for each cube is required.

Syntax



- DBS-NAME
- HOST-NAME

You can delete partition definitions in the following ways using **drop partition**.

Keywords

drop...partition...from
Remove a partition definition between the current target cube and a source cube.

drop...partition...to
Remove a partition definition between the current source cube and a target cube.

at <host-name>
Optionally specify the host location, if removing a partition definition associated with a remote instance.
Use the discovery URL to indicate the location. For example, "`https://myEssbase-myDomain.analytics.us2.example.com/essbase/agent`".

force
Specify that the source half of a partition definition should be dropped regardless of whether the target half is missing or invalid. For more information, see Forcing Deletion of Partitions.

Notes

If the **create partition** statement used was of the format:

```
create partition SOURCE to TARGET;
```

Then the only permutations of the **drop partition** statement that will have effect are:

```
drop partition SOURCE to TARGET;
drop partition TARGET from SOURCE;
```

Example

```
create or replace replicated partition
sampeast.east area '@IDESCENDANTS("Eastern Region"),
@IDESCENDANTS(Qtr1)' to samppart.company at "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent";

drop replicated partition Samppart.Company from Sampeast.East;
```

Drop Trigger

Remove a trigger created to track state changes over a selected cube area.

Syntax

TRIGGER-NAME

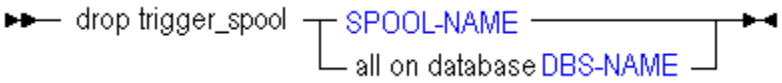
Example

```
drop trigger Sample.Basic.WatchCosts ;
```

Drop Trigger Spool

Delete the log file created by a trigger. Triggers track state changes over a selected cube area. For more information about triggers, see Examples of Triggers.

Syntax



- SPOOL-NAME
- DBS-NAME

Execute Calculation

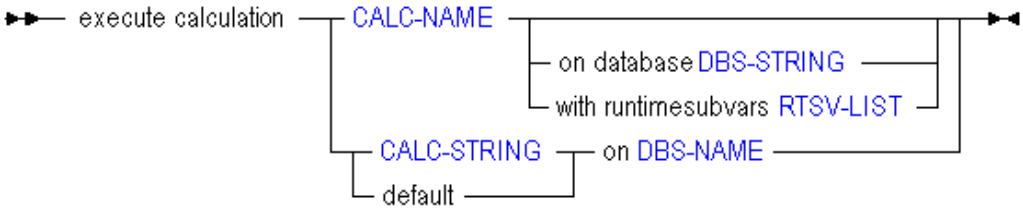
Click here for aggregate storage version

Execute a stored calculation, the stored default calculation (determined by alter database), or an anonymous (non-stored) calculation string.

Minimum permissions required:

- For stored calculations (CALC-NAME): Granted access to the calculation.
- For anonymous calculations (CALC-STRING) and the default calculation: Database Update

Syntax



- CALC-NAME
- DBS-STRING
- RTSV-LIST
- CALC-STRING
- DBS-NAME

You can run calculations in the following ways using execute calculation.

Keywords

execute calculation <calc-name>
Run the specified stored calculation script.

<calc-name> on database
Run the specified stored calculation script against the specified database.

<calc-string> on <dbs-name>
Run an anonymous calculation, whose body is contained in <*calc-string*>, against the specified database.

default on <dbs-name>
Run the default calculation against the specified database.

<calc-name> with runtimesubvars <rtsv-list>
Run the specified stored calculation script with the runtime substitution variables specified in *RTSV-LIST*, which is a string of runtime substitution variables specified as key/value pairs. The string must be enclosed with single quotation marks, and the key/value pairs must be separated by a semicolon, including a semicolon after the last runtime substitution variable in the string and before the terminal single quotation mark. In this example of a runtime substitution variable string, the name and value of four runtime substitution variables are specified (for example, the value of the runtime substitution variable named "a" is 100):

```
'a=100;b=@CHILDREN("100");c="Actual"->"Final";d="New York";'
```

The string of runtime substitution variables cannot exceed 64 KB.

> **Note:**
>
> Runtime substitution variables used in a calculation script must be declared in the SET RUNTIMESUBVARS calculation command, with a name and default value. If a different value is declared in the *RTSV-LIST*, the default value is overwritten at runtime.
> If you include a runtime substitution variable in *RTSV-LIST* that has not been declared in SET RUNTIMESUBVARS, Essbase ignores the undeclared runtime substitution variable (no warnings or exceptions are generated).
> Runtime substitution variables that are used in a calculation script can be logged in the application log file, using the ENABLERTSVLOGGING configuration setting. See "Logging Runtime Substitution Variables" in the *Designing and Maintaining Essbase Cubes*.
> If the name of a runtime substitution variable that is declared in the SET RUNTIMESUBVARS calculation command is the same as a runtime substitution variable declared in *RTSV-LIST*, the value specified in *RTSV-LIST* overwrites the default value in SET RUNTIMESUBVARS.

Notes

- A stored calculation can be associated with a specific database in an application (database level), or with an application only (application level). To execute a calculation stored at the application level, you must specify which database in the application to calculate using the **on database STRING** grammar.
- A calculation script can reference runtime substitution variables using the **with runtimesubvars** grammar.

Example

```
execute calculation Sample.Basic.Calc1;
```

Calculates the Sample.Basic database using the stored calculation script file named Calc1, which is associated with the database.

```
execute calculation Sample.Calc2 on database Basic;
```

Calculates the Sample.Basic database using the stored calculation script file named Calc2, which is associated with the Sample application.

```
execute calculation
  'SET MSG ERROR;
   CALC ALL;'
on Sample.basic;
```

Calculates the Sample.Basic database using an anonymous (unstored) calculation string.

```
execute calculation Sample.Basic.Calc3 with runtimesubvars
‘a=100;b=50;’;
```

Calculates the Sample.Basic database using the stored calculation script file named Calc3, which is associated with the database, and the specified runtime substitution variables, in which the value of the runtime substitution variable named "a" is 100 and the value of "b" is 50.

Execute Aggregate Process (Aggregate Storage)

Perform an aggregation, optionally specifying the maximum disk space for the resulting files, and optionally basing the view selection on user querying patterns.

This statement applies to aggregate storage databases only.

This statement enables you to build aggregate views with a minimum of settings. If greater control is needed, you can combine the following statements:

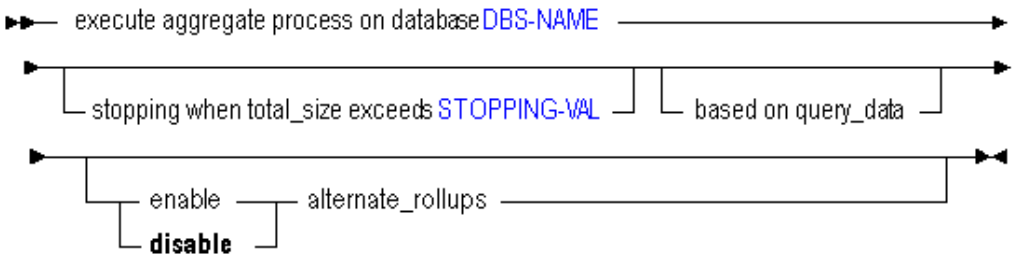
- Execute Aggregate Selection
- Execute Aggregate Build

This statement causes Essbase to:

1. Select 0 or more aggregate views based on the stopping value and/or on querying patterns, if given.
2. Build the views that were selected.

You can also configure Essbase to generate aggregate views automatically. For more information about aggregate views, see Aggregating an Aggregate Storage Database.

Syntax



- DBS-NAME
- STOPPING-VAL

You can aggregate an aggregate storage database in the following ways using **execute aggregate process**.

Keywords

stopping when total_size exceeds...
Aggregate whichever views Essbase selects, with the exception that the maximum growth of the aggregated database must not exceed the given ratio. For example, if the size of a database 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.

based on query_data
Aggregate whichever views Essbase selects, based on collected user querying patterns. This option is only available if query tracking is turned on, using alter database with the **enable query_tracking** grammar.

enable|disable alternate_rollups
If enabled, secondary hierarchies (with default level usage) are considered for view selection. Default: disabled (no secondary hierarchies are considered).

Notes

View selection (step 1) can be performed independently of aggregation by using execute aggregate selection. Aggregation (step 2) can be performed without built-in view selection by using execute aggregate build.

Example

```
execute aggregate process on database ASOsamp.Sample
stopping when total_size exceeds 1.3;
```

Selects and builds an aggregation of the ASOsamp.Sample database that permits the database to grow by no more than 30% as a result of the aggregation.

```
execute aggregate process on database ASOsamp.Sample based on
query_data;
```

Selects and builds an aggregation of the ASOsamp.Sample database, where the views that Essbase selects for aggregation are based on the most frequently queried areas of the database.

Related Topics

- Execute Aggregate Build
- Execute Aggregate Selection

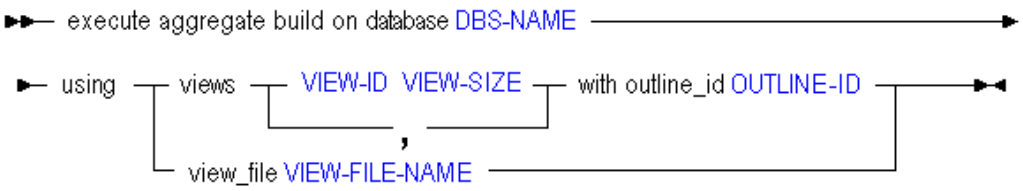
Execute Aggregate Build

Performs an aggregation based on the views selected by the execute aggregate selection statement.

The views to build must either be identified by their view IDs, obtained previously using execute aggregate selection, or by a view selection saved in an aggregation script.

You can also configure Essbase to generate aggregate views automatically. For more information about aggregate views, see Aggregating an Aggregate Storage Database.

Syntax



- DBS-NAME
- VIEW-ID
- VIEW-SIZE
- OUTLINE-ID
- VIEW-FILE-NAME

You can materialize aggregations in the following ways using **execute aggregate build**.

Keywords

using views...
Builds an aggregation based on a previously selected view (or views) and the associated outline ID.

using view_file...
Builds an aggregation based on a saved view selection stored in an aggregation script.
Omit the `.csc` file extension from the view file name when you issue the **execute aggregate build** statement.

Notes

- Although it is possible to pass arbitrary view-id and view-size arguments, this practice is not supported.
- Passing view-size arguments other than those returned by the **execute aggregate selection** command may cause unpredictable results.

Example

```
execute aggregate build on database Sample.Basic using views 711
0.00375 with outline_ID 4142187876;
```

Builds an aggregation of the Sample.Basic database. The build is based on the view of an aggregate storage outline

(identified as 4142187876) having the view ID 711, and a view size of 0.00375.

```
execute aggregate build on database Sample.Basic using view_file myView;
```

Builds an aggregation of the Sample.Basic database based on the view saved in the aggregation script `myView.csc`.

Related Topics

- Execute Aggregate Process (Aggregate Storage)
- Execute Aggregate Selection

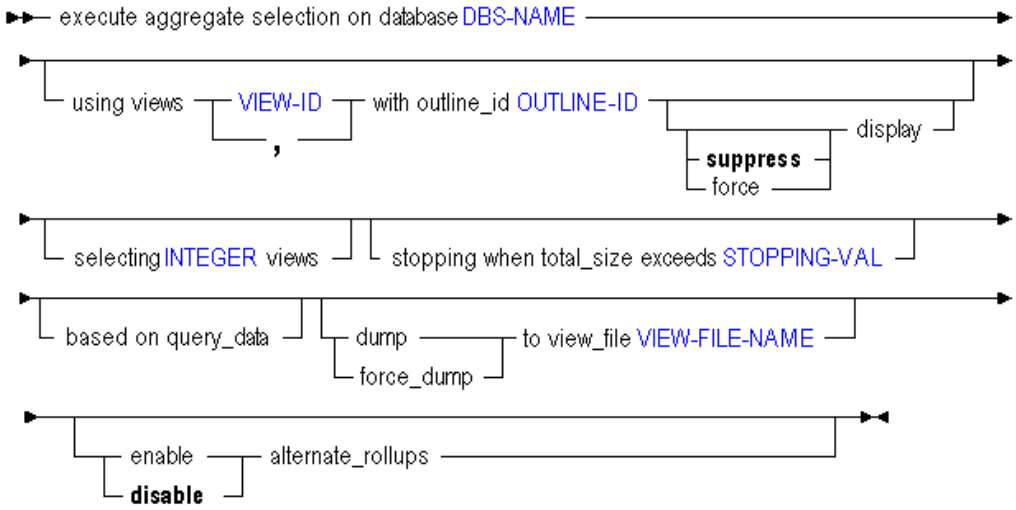
Execute Aggregate Selection

Select views of an aggregate storage database based on various selection criteria, and return the results in the form of a table or aggregation script. Next, use the tabular information or aggregation script to build an aggregation (materialize a view) using execute aggregate build.

> **Note:**
>
> View selection and aggregation can be performed by Essbase in a single step by using execute aggregate process. However, the use of the two separate statements **execute aggregate selection** and **execute aggregate build** enables you more control of the selection criteria.

You can also configure Essbase to generate aggregate views automatically. For more information about aggregate views, see Aggregating an Aggregate Storage Database.

Syntax



- DBS-NAME
- OUTLINE-ID
- VIEW-ID
- INTEGER
- STOPPING-VAL
- VIEW-FILE-NAME

You can select views in the following ways using **execute aggregate selection**.

Keywords

using views...with outline_ID
Selects views based on pre-selected view IDs. The view IDs are obtained from previous executions of the statement.

using views...with outline_ID...force display
Selects views based on pre-selected view IDs, including the pre-selected views IDs themselves.

using views...with outline_ID...suppress display
Selects views based on pre-selected view IDs, skipping the pre-selected views IDs themselves. This is the default behavior even if the **suppress** keyword is omitted.

selecting <INTEGER> views
Selects the number of views based on whether the number of views specified in <INTEGER> is greater than or equal to, or less than, the recommended number of default views that are returned by the **execute aggregate selection** statement. By default, Essbase determines the recommended number of default views.
Assume that <RECNUM> represents the recommended number of default views:

- If the value of <INTEGER> is greater than or equal to the value of <RECNUM>, the selected number of views equals <RECNUM>.

  For example, if <INTEGER> equals 20 and <RECNUM> equals 15, the number of selected number of views equals 15.

- If the value of <INTEGER> is less than the value of <RECNUM>, the number of views that are selected equals <INTEGER>.

  If you want the number of views that are selected to equal the value of <INTEGER>, use the **stopping when total_size exceeds <STOPPING-VAL>** grammar to change the number of recommended default views that are returned by the **execute aggregate selection** statement. Define the <STOPPING-VAL> factor large enough so that the number of default views that are returned by **execute aggregate selection** is greater than the value of <INTEGER>.

  For example, if <INTEGER> equals 20 and <RECNUM> equals 50, the number of selected number of views equals 20.

> **Note:**
>
> This parameter does not create views.

stopping when total_size exceeds <STOPPING-VAL>
Selects views, specifying a storage stopping value in terms of a factor times the size of the unaggregated input (level 0) values. For example, a stopping value of 1.5 means that the view selection should permit the database to grow by no more than 50% as a result of the aggregation.

based on query_data
Selects views based on previously collected query-tracking data. You must have already enabled query tracking. After enabling query tracking, allow sufficient time to collect user data-retrieval patterns before performing an aggregate selection based on query data.
Query tracking records information about every query executed on the database, so that it can be used as a basis for view selection. Query-based view selection helps to improve query performance when the distribution of user queries is skewed.
For every level combination, the cost of retrieving cells is recorded. The recording continues until the application is shut down or until the recording is explicitly turned off using **alter database <dbs-name> disable query_tracking**. In both cases, all the query cost data is discarded, and the recording stops (and will not continue when the application starts again).
All query cost data becomes invalid when additional views are built.
To create views based on tracked query patterns,

1. Enable query tracking using **alter database <dbs-name> enable query_tracking**.
2. Run all production queries once, and then select the first set of views based on the query cost data. To select the views, run this MaxL statement (**execute aggregate selection…based on query_data…**).
3. Build the selected aggregate view using **execute aggregate build**.
4. Repeat the previous two steps at least twice. Selecting and building multiple views iteratively helps ensure there are enough usage-tracking data to form a pattern. Each new view you build decreases the rate at which query costs grow.

dump to view_file
Saves the view selection to an aggregation script. If the specified script name already exists, an error is returned. To overwrite an existing script, use the **force_dump** keyword.
The aggregation script contains information derived during the aggregate view selection. You can materialize the aggregation at a different time by running the aggregation script. For example:`execute aggregate build on database <dbs-name> using view_file <view-file-name>`

force_dump to view_file
Saves the view selection to an aggregation script. If the specified script name already exists, the **force_dump** keyword causes it to be overwritten.

enable|disable alternate_rollups
If enabled, secondary hierarchies (with default level usage) are considered for view selection. Default: disabled (no secondary hierarchies are considered).

Example

```
execute aggregate selection on database ASOsamp.Sample;
```

Performs the default view selection for ASOsamp Sample. This statement selects the same views as `execute aggregate process on database ASOsamp.Sample` would build.

```
execute aggregate selection on database ASOsamp.Sample using views 711,
8941 with outline_ID 4142187876;
```

Selects views based on the pre-selected view IDs. The view IDs are obtained from previous executions of the statement.

```
execute aggregate selection on database ASOsamp.Sample using views 711,
8941 with outline_ID 4142187876 force display;
```

Selects views based on the pre-selected view IDs. **force display** is used to include the pre-selected views (711 and 8941) in the new selection.

```
execute aggregate selection on database ASOsamp.Sample stopping when
total_size exceeds 1.2;
```

Selects an aggregation of the ASOsamp Sample database that, when built, would permit the database to grow by no more than 20% as a result of the aggregation.

```
execute aggregate selection on database ASOsamp.Sample based on
query_data;
```

Selects views based on previously collected query-tracking data. You must have enabled query tracking using **alter database <dbs-name> enable query_tracking**.

```
execute aggregate selection on database ASOsamp.Sample
dump to view_file myView;
```

Selects a default aggregation of the ASOsamp Sample database, saving the selection to `APP\DB\myView.csc`. You can materialize the view later by running the aggregation script `myView.csc`. For example:

```
execute aggregate build on database ASOsamp.Sample using view_file
'myView.csc';
```

Related Topics

- Execute Aggregate Build

- Execute Aggregate Process (Aggregate Storage)

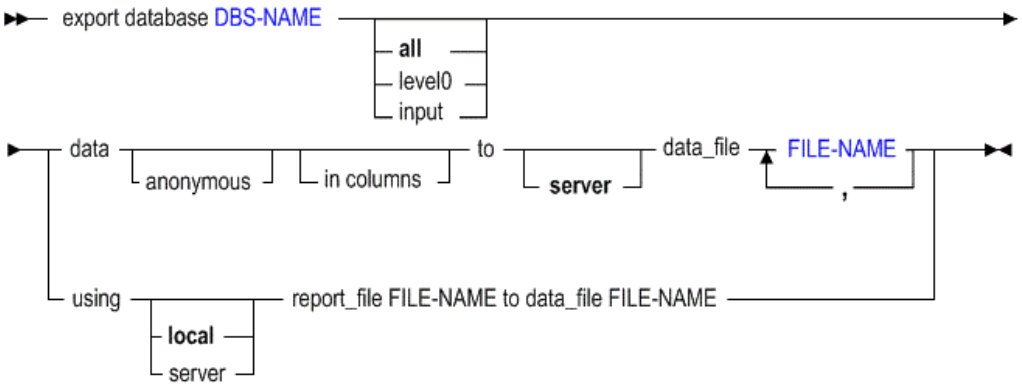
Export Data

Click here for aggregate storage version

Export all data, level-0 data, or input-level data, which does not include calculated values. Export data files are written to the application/cube directory. To use Report Writer, export the data using a report file.

Minimum permission required: Read.

Syntax



- DBS-NAME
- FILE-NAME

You can export data from a database in the following ways using **export data**.

Keywords

export database <dbs-name> all data...
Export all data in the specified cube to the application/cube directory.

> **Note:**
>
> Exporting data does not clear the data from the database.

export database <dbs-name> level0 data...
Export level-0 data blocks only (blocks containing only level-0 sparse member combinations. Note that these blocks may contain data for upper level dense dimension members.) A level-0 block is created for sparse member combinations when all of the members of the sparse combination are at the bottom of dimension branches.

> **Note:**
>
> Exporting data does not clear the data from the cube.

export database <dbs-name> input data...
Export only blocks of data where the block contains at least one data value that was loaded (imported), rather than created as the result of a calculation.

export database <dbs-name> ... data in columns
Export data in columns, to facilitate loading the exported data into a relational database. In each row, the columnar format displays a member name from every dimension. Names can be repeated from row to row.
Columnar format provides a structure to the exported data, so that it can be used for further data processing by applications other than Essbase tools. In non-columnar format, sparse members identifying a data block are included only once for the block. Because the export file in non-columnar format is smaller than in columnar format, reloading a file in non-columnar format is faster.

export database <dbs-name> ... data anonymous
Export data in anonymized format. Anonymization removes the risk of sensitive data disclosure, and can be used in case sample data needs to be provided for technical support. Essbase replaces real data values with incremental values beginning with 0, increasing by 1 for each value in the block.

export database <dbs-name> ...using...report_file...
Run a stored report script, exporting a subset of the database.

Notes

- This statement requires the database to be started.
- To export data in parallel, specify a comma-separated list of export files, up to a maximum of 1024 file names. The number of file names determines the number of export threads. The number of available block-address ranges limits the number of export threads that Essbase actually uses. Essbase divides the number of actual data blocks by the specified number of file names (export threads). If there are fewer actual data blocks than the specified number of export threads, the number of export threads that are created is based on the number of actual data blocks. For example, if the block storage database is very small, with only 100 data blocks, Essbase will use only 100 threads, even if you specify a higher number. This approach results in a more even distribution of data blocks between export threads.

  > **Note:**
  >
  > In specifying the number of export files, it is important to consider the number of available CPU cores and I/O bandwidth on the computer on which Essbase Server runs. Specifying too large a number can result in poor performance.

  If the data for a thread exceeds 2 GB, Essbase may divide the export data into multiple files with numbers appended to the file names.

The naming convention for additional export files is as follows: `_1`, `_2`, etc. are appended to the additional file names. If the specified output file name contains a period, the numbers are appended before the period. Otherwise, they are appended at the end of the file name.

For example, if the given file name is `exportfile.txt`, the next additional file is `exportfile_1.txt`.

- To export data in column format, use the optional "in columns" grammar.
- During a data export, the export process allows users to connect and perform read-only operations.
- When MaxL exports data from a Unicode-mode application, the export file is encoded in UTF-8. You cannot use UTF-8-encoded export files from a Unicode-mode application to import data to a non-Unicode-mode application.
- MaxL cannot export databases with names containing hyphens (-).

Example

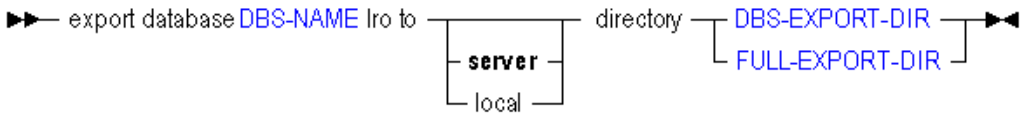
```
export database Sample.Basic data to server data_file 'myfilesamp.txt'
```

Exports data to `Sample/Basic/myfilesamp.txt`.

Export LRO

Export linked-reporting-object information, and binary files if the database has file-type LROs, to a directory.

Syntax



- DBS-NAME
- DBS-EXPORT-DIR
- FULL-EXPORT-DIR

You can export LRO information from a database in the following ways using **export lro**.

Keywords

to server directory
Export the LRO information to a directory you specify on the Essbase Server to which you are connected.

to local directory
Export the LRO information to a directory you specify.

Notes

- This statement requires the database to be started.
- MaxL creates exactly one export directory; it does not create a directory *structure*. For example, if `c:\temp` exists, MaxL will create `c:\temp\exports`, but not `c:\temp\exports\to\this\long\path`.
- If the specified export directory already exists, the export LRO statement will fail. This is a safeguard against overwriting existing export directories.
- If you do not specify a *full path* for an export directory to be created on the client or server, MaxL uses your short directory specification (DBS-EXPORT-DIR) as a suffix, and creates the destination export-directory in the `ARBORPATH\app` directory with a prefix of `appname-dbname-`. If you do specify a full path, MaxL creates whatever directory you specify.
- When MaxL exports LROs from a database, if the database is from a Unicode-mode application, the exported LRO-catalog file is encoded in UTF-8. You cannot use UTF-8-encoded export files from a Unicode-mode application to import LROs to a non-Unicode mode application.

Example

```
export database sample.basic lro to server directory '../home/temp/
lros';
```

Exports LRO-catalog information, and binary files if the database has file-type LROs, to a server directory called `home/temp/lros`. The directory contains file-type LROs, if applicable, and the LRO-catalog export file `lros.exp`. These can be brought back into a database using import lro.

```
export database sample.basic lro to server directory 'exportedLROs';
```

Exports LRO-catalog information, and binary files if the database has file-type LROs, to a server directory `$ARBORPATH/app/sample-basic-exportedLROs`. The directory contains file-type LROs, if applicable, and the LRO-catalog export file named `sample-basic-exportedLROs.exp`. These can be brought back into a database using import lro.

```
export database sample.basic lro to server directory 'D:\\MaxL\
\LROexports\\dir';
```

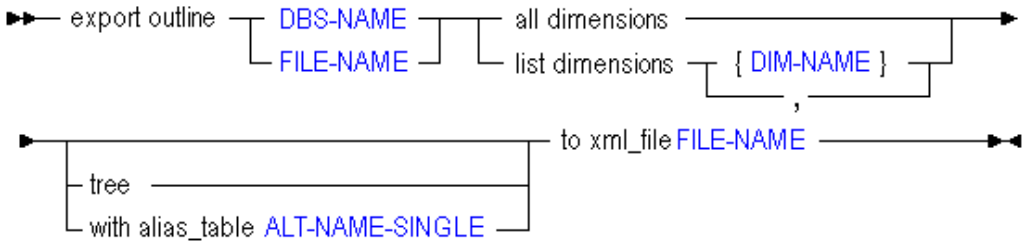
On Windows, exports LRO-catalog information to a new directory `dir` under the existing directory structure `D:\MaxL\LROexports`. The double backslashes (\\) must be used because a single backslash is an escape character to MaxL.

Export Outline

Export metadata, either from the active database outline or an input outline file, to a specified XML file. Export outline files must be written to a location on the Essbase Server or client computer on which the **export outline** MaxL statement is run.

Permission required: Database Manager.

Syntax



- DBS-NAME
- FILE-NAME
- DIM-NAME
- ALT-NAME-SINGLE

You can export metadata information from a database in the following ways using **export outline**.

Keywords

DBS-NAME
Specify the database name instead of the outline file path.

FILE-NAME
Specify the outline file path instead of the database name.

all dimensions
Export information about all dimensions in the database.

list dimensions
Export information about only the listed dimensions. Specify each dimension name within curly braces, and separated by commas.

tree
Export only the member names in the hierarchy, omitting full metadata details.

with alias_table
Export using only the member names indicated in the specified alias table.

to xml_file
Specify the full path to the output XML file.

Notes

- This statement requires the database to be started.
- The following general outline information is included in the XML export:
  - Case sensitiveness
  - Outline Type
  - Duplicate Member Names allowed
  - Typed Measures Enabled
  - Date Format
  - Varying Attributes Enabled
  - Alias Table count and list
  - Active Alias Table
  - Attribute information
  - Auto configure
  - Text list definitions
  - Universal member comments
  - Locale, if it exists
  - Query hint list (if aggregate storage)
- The following dimension information is included in the XML export:
  - Name
  - Two pass calc
  - Type
  - Text list, if text typed
  - Formula
  - Format String
  - Comment
  - Extended member comment
  - Dimension category
  - Attribute type
  - Data Storage
  - Dimension Storage
  - Alias Names, if any
  - UDAs, if any
  - Consolidation
  - Attribute dimension associated
  - Independent dimensions, if any
  - Time balance

  - Skip options
  - Variance reporting
  - Currency conversion
  - Currency conversion member
  - Dynamic Time Series enabled list
  - Attachment level, if linked attribute dimension
  - Dimension solve order
  - Is Non Unique dimension?
  - Hierarchy type
  - Level usage for aggregation (for aggregate storage hierarchies)
  - Is Compression dimension? (if aggregate storage)
  - Storage category
- The following member information is included in the XML export:
  - Name
  - Two pass calc
  - Type
  - Text list, if text typed
  - Is shared?
  - Shared member name, if shared
  - Formula
  - Format string
  - Comment
  - Extended member comment
  - Attribute type
  - Data storage
  - Dimension storage
  - Alias names, if any
  - UDAs, if any
  - Consolidation
  - Attribute member associated
  - Validity sets, if any
  - Time balance
  - Skip options
  - Variance reporting
  - Currency conversion
  - Currency conversion member
  - Member solve order (if aggregate storage)

- Level usage for aggregation (for aggregate storage hierarchy members)

Example

```
export outline sample.basic all dimensions to xml_file "c:/temp/
basic.xml";
```

Exports all outline information from Sample.Basic to the specified XML file, `basic.xml`.

```
export outline sample.basic list dimensions {"Product", "Market"} tree
to xml_file "c:/temp/basic.xml";
```

Exports information about Product and Market dimensions from Sample.Basic to the XML file.

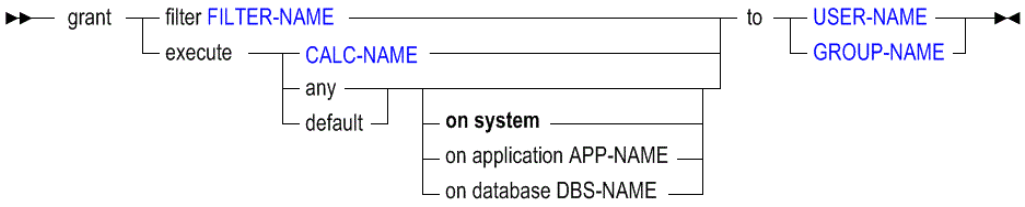
```
Export outline "c:/temp/basic.otl" all dimensions with alias_table
"Default" to xml_file "c:/temp/basic.xml";
```

Exports information about all dimensions in Sample.Basic from the specified outline file to the XML file, using only default alias names.

Grant

Grant a a filter or a stored calculation to a user or a group.

Syntax



- FILTER-NAME
- USER-NAME
- GROUP-NAME
- CALC-NAME

You can grant permissions to users and groups in the following ways using **grant**.

Keywords

filter <filter-name> to...
Assign a filter to a user or group that grants or denies permissions to the specified database at a data-value level of detail.

execute <calc-name> to...
Grant the user or group permission to run the specified stored calculation script.

execute any on system to...
Grant the user or group permission to run any calculation against any database on the Essbase Server.

execute any on application...to...
Grant the user or group permission to run any calculation against any databases in the specified application.

execute any on database...to...
Grant the user or group permission to run any calculation against the specified database.

execute default on system to...
Grant the user or group permission to run the default calculation against any database on the Essbase Server.

execute default on application...to...
Grant the user or group permission to run the default calculation against any databases in the specified application.

execute default on database...to...
Grant the user or group permission to run the default calculation against the specified database. The default calculation is typically 'CALC ALL;', but it can be changed using **alter application set default calculation**.

Notes

Granting filters:

Users may be granted multiple filters per database.

Granting calculations:

A user or group may have any number of calculations per database. Therefore, granting a calculation adds it to the user or group's list of calculations. **Grant execute any** gives the user or group permission to execute all calculations, including the default calculation.

Example

```
grant filter Sample.basic.filter8 to Fiona;
```

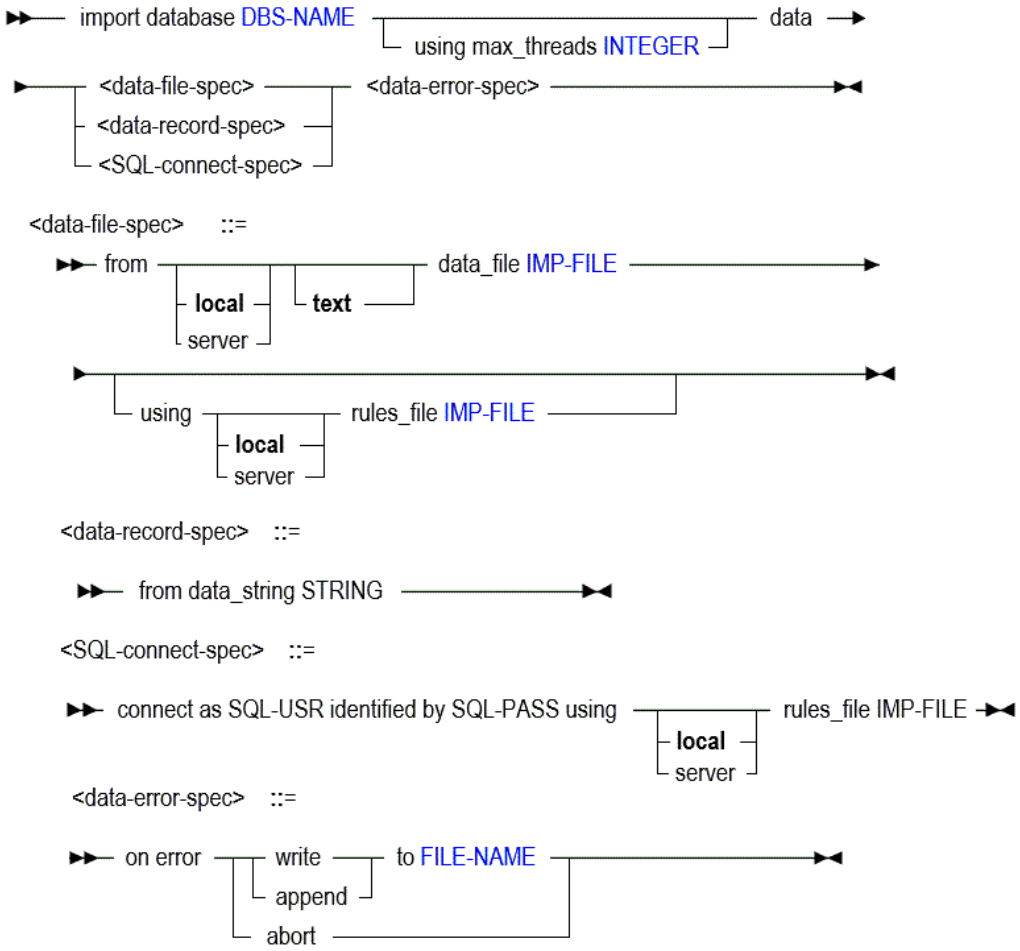
Import Data

Click here for aggregate storage version

Import data from data files, with or without a rules file.

Minimum permission required: Write.

Syntax



- DBS-NAME
- INTEGER
- IMP-FILE
- FILE-NAME

You can import data to a database in the following ways using **import data**.

Keywords

...using max_threads INTEGER
Optionally specify a maximum number of threads to use, if this is a parallel data load. If this clause is omitted for a parallel data load, Essbase uses a number of pipelines equal to the lesser of number of files, or half the number of CPU cores.

import database <dbs-name> data from...
Specify whether the data import file(s) are local or on the server, and specify the type of import file(s).

To import from multiple files in parallel, use the wildcard characters * and/or ? in the IMP-FILE name so that all intended import files are matched.

- * substitutes any number of characters, and can be used anywhere in the pattern. For example, `day*.txt` matches an entire set of import files ranging from `day1.txt - day9.txt`.
- ?* substitutes one occurrence of any character, and can be used anywhere in the pattern. For example, `0?-*-2011.txt` matches data source files named by date, for the single-digit months (Jan to Sept).

...using ... rules_file
Import data into the database using a specified rules file. If you are using a rules file for a parallel data load, all the data files in the load must be able to use the same rules file.

...<data error spec> (on error...)
Required. Tell Essbase what to do in case of errors during the data load: abort the operation, or write or append to a specified error log.

...<data record spec> from data_string
Load a single data record into the selected database. The string following **data_string** must be a contiguous line, without newline characters.

...<SQL connect spec> (connect as...)
If you are importing data from an SQL source, provide your SQL user name and password. You must always use a rules file when you load SQL data sources.

Notes

- This statement requires the database to be started.
- When using the import statement, you must specify what should happen in case of an error.
- To import from a SQL data source, you must connect as the relational user name, and use a rules file.

Example

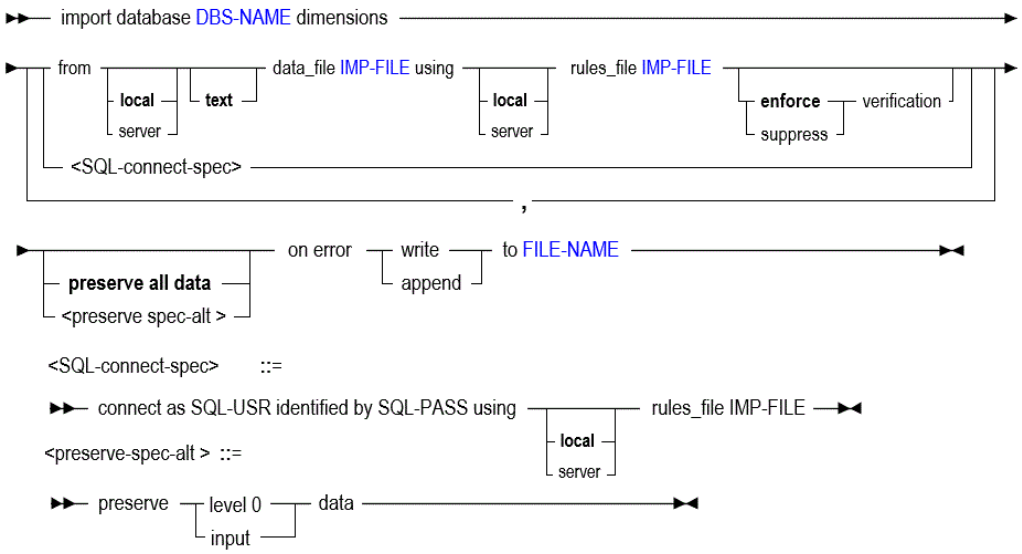
```
import database Sample.Basic data from server data_file 'expsamp.txt'
on error abort;


import database Sample.Basic data from server data_file '/Sample/Basic/
expsamp.txt' on error abort;
```

Import Dimensions

Import dimensions from data files, using a rules file.

Minimum permission required: Write.

Syntax



- DBS-NAME
- IMP-FILE
- FILE-NAME

You can import dimensions to a database in the following ways using **import dimensions**.

Keywords

import database <dbs-name> dimensions from...
Specify whether the dimension import is from a local or server file, and what type of file to import the dimension from.

...using ... rules_file
Import dimensions into the database outline using a specified rules file.

...enforce verification
Verify the outline resulting from the dimension build. This is the default behavior.

...suppress verification
Do not verify the outline resulting from the dimension build.

> **Caution:**
>
> Using this option defers restructuring.

...preserve all data
If you need to preserve all data when importing dimensions, specify that here.

...on error...
Tell Essbase what to do in case of errors during the dimension build: abort the operation, or write or append to an error log.

...<SQL connect spec> (connect as...)
If you are importing dimensions from an SQL source, provide your SQL user name and password. You must always use a rules file when you load SQL data sources.

...<preserve spec alt> (preserve...data)
If you need to preserve level-0 or input data when importing dimensions, specify that here.

Notes

- This statement requires the database to be started.
- When using the import statement, you must specify how error logs should be handled.
- When multiple files are included in the same statement, restructure is deferred until all files have been processed. The deferred-restructure type of dimension build has been called an incremental dimension build.
- When the **suppress verification** option is used, restructure is deferred.
- When multiple files are included in the same statement, **be sure verification is enforced for the last file**.
- To import from a SQL data source, you must connect as the relational user name, and use a rules file.

Example

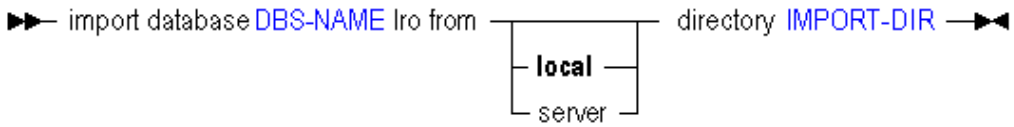
```
import database sample.basic dimensions
from data_file 'dims.txt' using rules_file 'rulesfile.rul'
on error append to 'dimbuild.log';
```

Import LRO

Import Linked Reporting Objects (LROs) from the specified output directory created by export lro. The directory contains an ASCII `.exp` file containing LRO-catalog information, and LRO binary files (if the database from which LROs were exported contained file-type LROs).

Minimum permission required: Write.

Syntax



- DBS-NAME
- IMPORT-DIR

You can import exported LRO information to a database using **import lro**.

Keywords

import database <dbs-name> lro...
Import Linked Reporting Objects (LROs) from the specified export directory on the local computer or on a remote server where the Essbase Server resides.

Notes

- This statement requires the database to be started.
- The specified import directory must come from the results of the export lro operation. The exported LRO-catalog file contains a record of the LRO file locations, cell notes, or URL text, and database index locations to use for re-importing to the correct data blocks.
- In the paths in the second two examples, double quotation marks are used to allow variable expansion in the string IMPORT-DIR, and single quotation marks are required because there are special characters (see MaxL Syntax Notes) in the path name.

Example

Windows Example

```
import database sample.basic lro
from server directory 'C:\\Hyperion\\products\\Essbase\\EssbaseServer\
\app\\sample-basic-lros';


import database sample.basic lro
from directory "'$ARBORPATH\\app\\sample-basic-lros'";
```

UNIX Example

```
import database sample.basic lro

  from server directory "'$ARBORPATH/app/sample-basic-lros'";
```

From the subdirectory created by **export lro** in the app directory on the server, both the Windows and UNIX example statements above re-import the LRO-catalog information (and file-type LROs if applicable) that were exported to that location.

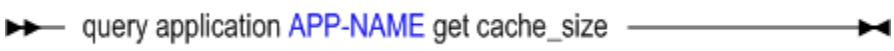
Query Application

Click here for aggregate storage version

Get information about the current state of the application.

This statement requires the application to be started.

Syntax



APP-NAME

You can query application state information using keywords.

Keywords

get cache_size
Check the current maximum size setting to which the application cache may grow. The application cache grows dynamically until it reaches this limit. The application cache is used for hybrid aggregation in block storage databases, and can help you manage memory usage for retrievals.

Example

The following MaxL statement:

```
query application sample get cache_size;
```

returns the maximum size (in kilobytes) to which the application cache may grow.

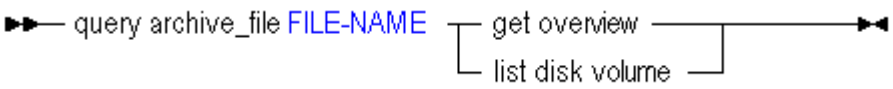
Query Archive_File

Retrieve information about the database backup archive file.

Minimum permission required: Read.

The database must be running.

Syntax



FILE-NAME

You can query archive file information using keywords.

Keywords

get overview
Retrieve the following overview information:

- Application name
- Database name
- Time when the archive was performed

list disk volume
Retrieve a list of disk volume names.
On Windows, Essbase adds the default `ARBORPATH` drive (for example, the `C:` drive) as a disk volume, even if the database that you backed up does not store data on that disk volume.

Example

```
query archive_file /Hyperion/samplebasic.arc get overview;
```

Retrieves overview information about the `samplebasic.arc` backup archive file.

```
query archive_file /Hyperion/samplebasic.arc list disk volume;
```

Retrieves disk volume information about the `samplebasic.arc` backup archive file.

Query Database

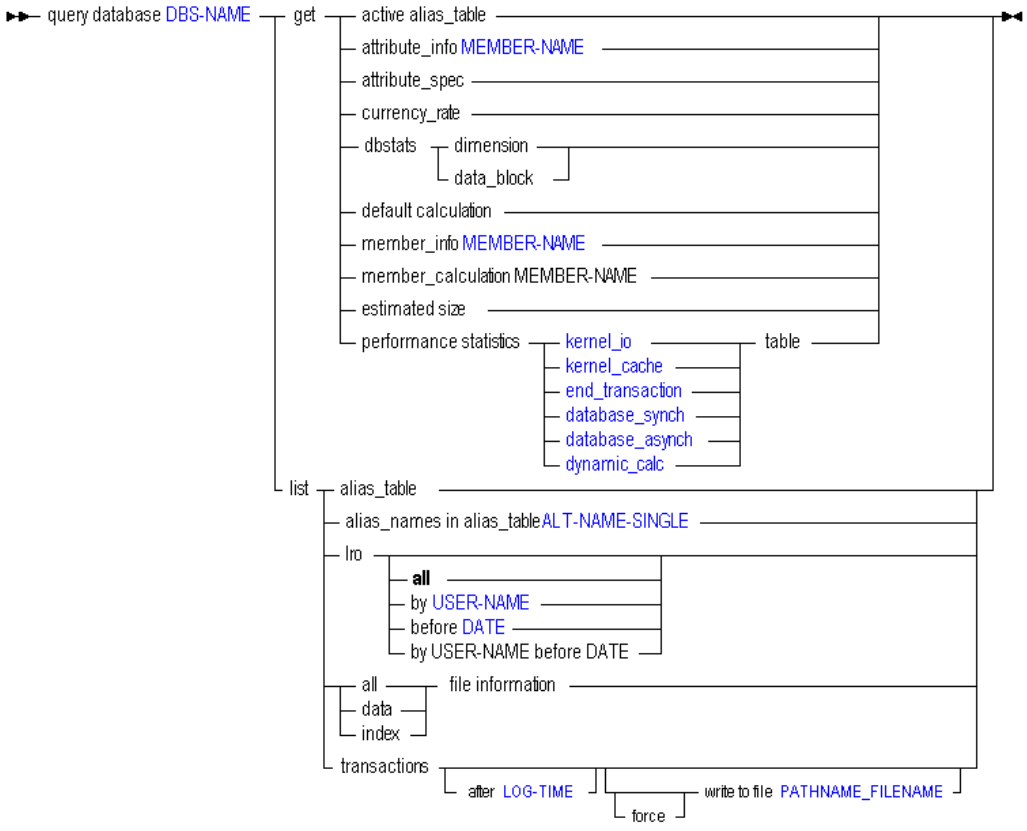
Click here for aggregate storage version

Get advanced information about the current state of the database.

Minimum permission required: Read.

This statement requires the database to be started.

Syntax



- DBS-NAME
- MEMBER-NAME
- kernel_io
- kernel_cache
- end_transaction
- database_synch
- database_asynch
- dynamic_calc
- ALT-NAME-SINGLE
- USER-NAME
- DATE
- LOG-TIME
- PATHNAME_FILENAME

You can query for database information in the following ways using **query database**.

Keywords

get active alias_table
Display the active alias table for the user issuing the statement.

get attribute_info
Get attribute member, dimension, and name information for the specified attribute member.

get attribute_spec
Display the current attribute specifications for the database. These specifications include attribute member name format, Attribute Calculation dimension member names, Boolean and date member names, and numeric range specifications.

get currency_rate
Display the currency rate for every currency partition.

get dbstats dimension
Get information about dimensions.
Output
The **index_type** field values are numeric, and translate as follows:

```
0   Dense
1               Sparse
3               None (database is aggregate storage)
```

get dbstats data_block
Get information about data blocks. The information returned has little relevance to aggregate storage databases.
Output

The type field values are numeric, and translate as follows:

```
0               Array
1               AVL (or "B+ Tree")
```

get default calculation
View the contents of the calculation designated as default for the database. The default calculation refers to either the relations defined in the database outline (CALC ALL) or to the set of calculation strings defined as the default database calculation.

get member_info MEMBER-NAME
Get information on a specific member.
Output
The **unary_type** field values are numeric, and translate as follows:

```
0               Add
1               Subtract
2               Multiply
3               Divide
4               Percent
5               NoRollUp
```

The **member_tag_type** field values translate as follows:

```
0               SkipNone
16384           SkipMissing
32768           SkipZero
49152           SkipBoth
1               BalFirst
2               BalLast
4               TwoPass
8               Average
64              Expense
```

Variations are possible. The field value consists of one of the first four "skip" values plus any/all/none of the last five values. Some examples:

```
0               SkipNone
77              SkipNone, BalFirst, TwoPass, Average, Expense
16385           SkipMissing and BalFirst
```

The first four "skip" values are base values, and added to them are combinations of 1, 2, 4, 8, and 64.
The **status** field values are hexadecimal, and translate as follows:

```
0               Normal
1               Never Share
2               Label
4               Refer Share
8               Refer Share (with different name)
16              Implicit share
```

```
32              Virtual Member (stored)
64              Virtual Member (not stored)
2048            Attribute
32768           Referred
```

get member_calculation MEMBER-NAME
View the formula associated with the selected member.

get estimated size
Display an estimate of the number of blocks a database will create after full calculation (CALC ALL), based on the number of blocks that exist before calculation. The database can have all data loaded, or it can have a random sampling of data loaded. Outlines that contain sparse formulas of any type or top-down formulas are not supported. Results of the estimation on such databases may be invalid.

performance statistics...table
Display one of several choices of performance statistics tables. Before you can use this statement, you must enable performance statistics gathering, using alter database DBS-NAME set performance statistics enabled.

list alias_table
Get a list of alias tables that are defined for the database.

list alias_names in alias_table
List the alias names defined in an alias table. Alias tables contain sets of aliases for member names and are stored in the database outline. Use this grammar to see a list of alias names defined in the specified table.

list lro
Get information about linked objects, including the object type, name, and description, based on criteria you specify. If you specify both a user name and modification date, objects matching both criteria are listed. If you specify no user name or date, a list of all linked objects in the database is displayed.

list...file information
Get accurate index and data file information. Provides index and data file names, counts, sizes, and totals, and indicates whether or not each file is presently opened by Essbase. The file size information is accurate. Note that the file size information provided by the Windows operating system for index and data files that reside on NTFS volumes may not be accurate.

list transactions
Display, in the MaxL Shell window, database transactions that were logged after the time when the last replay request was originally executed or after the last restored backup's time (which ever occurred later).

list transactions after LOG-TIME
Display, in the MaxL Shell window, database transactions that were logged after the specified time. Enclose the TIME value in quotation marks; for example: '11_20_2007:12:20:00'

list transactions after LOG-TIME write to file PATHNAME_FILENAME
Write the list of database transactions to the specified file. The list output is written to a comma-separated file on the Essbase Server computer.

Provide the full pathname to an existing directory and the name of the output file. If only the output file name is provided, Essbase writes the file to the *ARBORPATH*`/app` directory.
When writing to an output file that already exists, you must use the **force** grammar to overwrite the file.

list transactions force write to file PATHNAME_FILENAME
Overwrite the contents of an existing output file.

list transactions after TIME...write to file PATHNAME_FILENAME
Write the list of database transactions that were logged after the specified time to the specified file.

Example

Example 1

```
query database Sample.Basic list transactions;
```

Displays, in the MaxL Shell window, Sample.Basic database transactions that were logged after the time when the last replay request was originally executed or after the last restored backup's time (which ever occurred later).

Example 2

```
query database Sample.Basic list transactions after
'11_20_2007:12:20:00'
write to file 'C:\\Hyperion\\products\\Essbase\\EssbaseServer\\app\
\Sample\\Basic\\listoutput.csv';
```

Writes the transactions in the Sample.Basic database that were logged after November 20, 2007 at 12:20:00 to a CSV file in the Sample.Basic database directory.

Example 3

```
query database sample.basic get member_calculation 'Profit per Ounce';
```

Displays the formula associated with the 'Profit per Ounce' member.

Example 4

```
query database sample.basic list lro before '06_16_2008';
```

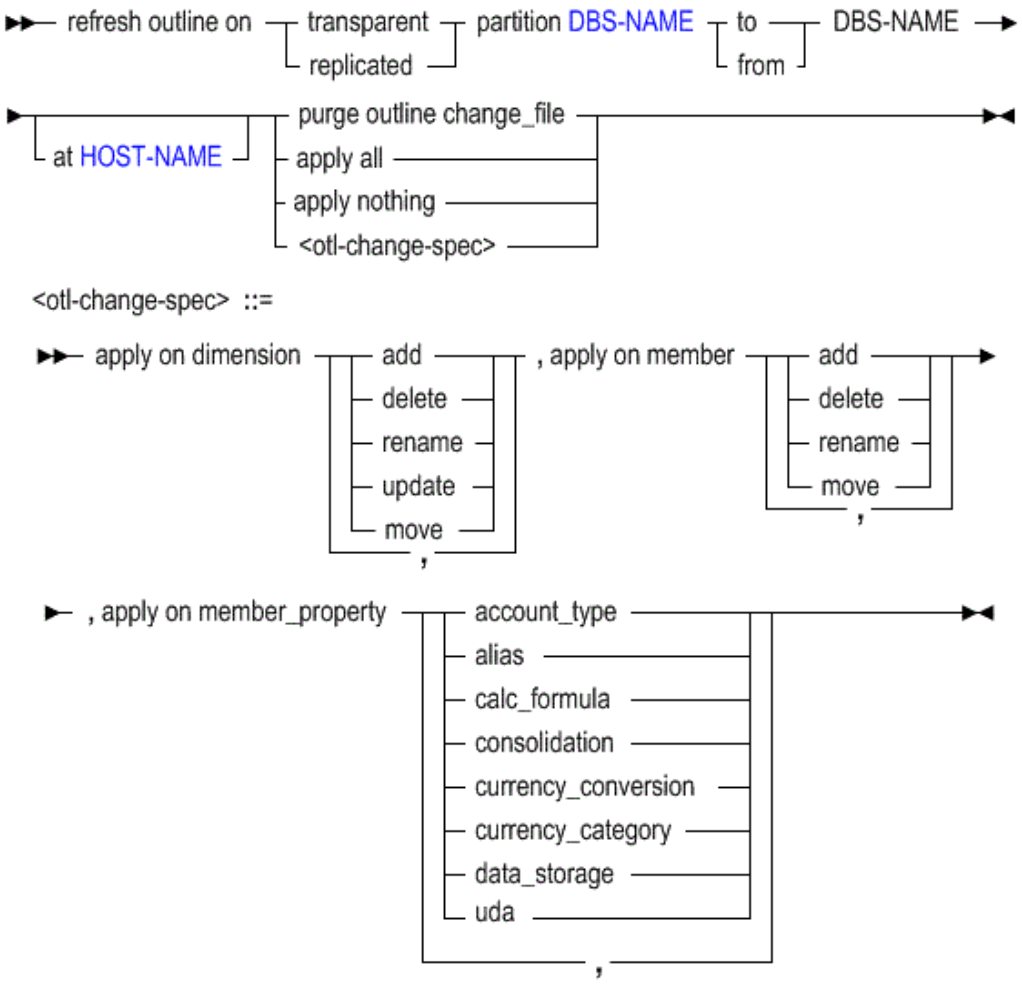
Displays information about linked objects, in the Sample.Basic database, that were modified before the specified time.

Refresh Outline

Synchronize the outlines between partitioned databases. Use this in the event that one outline has undergone changes to dimensions, members, or member properties, and you wish to propagate those changes to the partitioned database.

Outline synchronization is not currently enabled for partitions that involve aggregate storage databases.

Syntax



- DBS-NAME
- HOST-NAME

You can synchronize the outlines between partitioned databases using **refresh outline**.

Keywords

...to...
Use the current source outline to refresh the remote target outline.

...from...
Refresh the current target outline using the remote source outline.

purge outline change_file
Clear any source outline changes that have already been applied to the target outline or have been rejected. Source outline changes that have not been applied or rejected are not deleted from the outline change file.

apply all
Refresh all aspects of the target outline, including dimension changes, member changes, and member property changes made to the source outline. This is the recommended method for refreshing outlines, because if you choose to omit some changes, those changes cannot be applied later.

apply nothing
Do not apply source outline changes to any aspects of the target outline. The target outline will be considered synchronized to the source, and the timestamp will be updated, although source changes were not actually applied to the target.

apply on dimension...
Refresh the target outline with all or some dimension changes made to the source outline.

- **add**: Refresh with added dimensions.
- **delete**: Refresh by deleting dimensions.
- **rename**: Refresh with renamed dimensions.
- **update**: Refresh with dimensions that have member updates (required if the statement will also use **apply on member**).
- **move**: Refresh the order of dimensions in the outline.

Use commas to separate the types of source dimension changes to refresh on the target. For example, to refresh only with added or moved dimensions, use the following phrase: `apply on dimension add, move`.

apply on member...
Refresh the target outline with all or some physical member changes made to the source outline. Requires **apply on dimension update**.

- **add**: Refresh dimensions with added members.
- **delete**: Refresh dimensions by deleting members.
- **rename**: Refresh dimensions with renamed members.
- **move**: Refresh the order or hierarchy of members in the dimension.

Use commas to separate the types of source member changes to refresh on the target. For example, to refresh only with added or moved members, use the following phrase: `apply on dimension update, apply on member add, move`.

apply on member_property...
Refresh the target outline with all or some member property changes made to the source outline. Requires **apply on dimension update**.

- **account_type**: Refresh with changes in account type.
- **alias**: Refresh with changes to aliases.
- **calc_formula**: Refresh with changes to member formulas.
- **consolidation**: Refresh with changes to consolidation tags.
- **currency_conversion**: Refresh with changes to currency conversion flags.
- **currency_category**: Refresh with changes to currency categories.
- **data_storage**: Refresh with changes to data storage tags.
- **uda**: Refresh with changes to UDAs.

Use commas to separate the types of source member-property changes to refresh on the target. For example, to refresh only with updated member formulas, use the following phrase: `apply on dimension update, apply on member_property calc_formula`.

Example

```
refresh outline on replicated partition sampeast.east to
samppart.company
 apply all;
```

Refreshes the target outline (for Samppart.company database) with any and all changes made to the source outline (Sampeast.east).

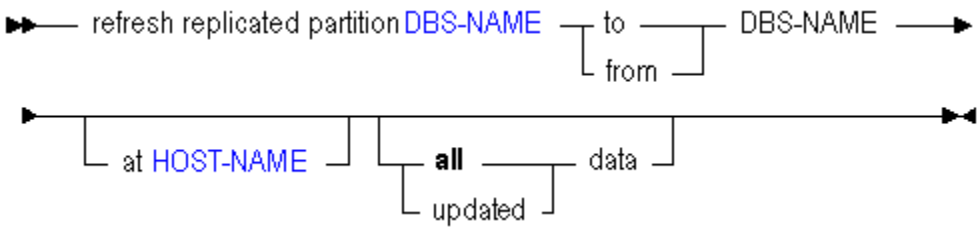
```
refresh outline on replicated partition Sampeast.east to
Samppart.company
apply on dimension update, apply on member rename, apply on
member_property  account_type;
```

Refreshes the target outline (for Samppart.company database) with changes made to the source outline (Sampeast.east), reflecting the following update to a dimension: a member tagged Accounts was renamed.

Refresh Replicated Partition

Refresh the current replicated-partition target cube from the remote (second DBS-NAME) source partition. Database Manager permission for each cube is required.

Syntax



- DBS-NAME
- HOST-NAME

You can update a replicated-partition using **refresh replicated partition**.

Keywords

...to...
Use the current replicated-partition source cube to refresh the remote target partition.

...from...
Refresh the current replicated-partition target cube from the remote source partition.

...updated data
Refresh a replicated-partition cube only with data that has been updated since the last refresh.

...all data
Refresh a replicated-partition cube with all data, regardless of the last refresh.

Example

```
refresh replicated partition sampeast.east to samppart.company
at "https://myEssbase-myDomain.analytics.us2.example.com/essbase/agent"
all data;
```

MaxL Definitions

This section contains the following topics:

- MaxL Syntax Notes
- Numbers in MaxL Syntax
- Terminals
- Privileges and Roles
- Quoting and Special Characters Rules for MaxL Language

MaxL Syntax Notes

The following syntax scheme applies to the creation of MaxL statements.

A MaxL **statement** corresponds to a sentence telling Essbase what to do with users and database objects. In this documentation, the grammar of MaxL statements is illustrated using railroad diagrams.

When issued via the MaxL Shell (essmsh), statements must be terminated by semicolons. Semicolons are used only to tell the shell when to terminate the statement; semicolons are not part of the MaxL language itself. Therefore, when issuing MaxL statements programmatically external programs, *do not* terminate with a semicolon.

A **token** is a delimited sequence of characters recognized by MaxL as a single readable unit. Tokens may be singleton names, keywords, strings, or numbers. Names can have one, two, or three tokens, delimited by periods. The space delimiting tokens can be any white space: spaces, tabs, new lines, or blank lines.

A **keyword** is a sequence of alphabetic characters that is part of the MaxL grammar. Each keyword is recognized as one token. To be recognized as keywords, keywords cannot be enclosed in quotation marks. However, if you wish to use MaxL keywords outside of the grammar as *terminals* (for example, as database names or passwords), they must be enclosed in single or double quotation marks.

A **terminal** is something referenced in the grammar for which you provide the correct name or definition. Terminals can be names, numbers, or strings. Examples: user-name, filter-name, size-string.

A **name** is a string which can be quoted or unquoted. Unquoted names must begin with an alphabetic character. Quoted names can consist of any sequence of characters. Names in MaxL are used to uniquely identify databases and database objects, such as users, applications, or filters.

Names in MaxL may be one of three types:

- *singletons*, which are names with one token (example: `Sample`). Use a singleton name for objects that have a system-wide context: for example, applications.
- *doubles*, which are names with two tokens. A double is two names connected by a period (example: `Sample.basic`). Use doubles to name objects with application-wide contexts, such as databases.
- *triples*, which are names with three tokens. A triple is three names connected by two periods (example: `Sample.Basic.Calcname`). Use triples to name objects having database-wide contexts, such as filters.

A **string** is unquoted or quoted. An unquoted string can be any sequence of non-special characters. A quoted string can be any sequence of characters (special, alphabetic, or numeric) in the MaxL Alphabet, enclosed in single or double quotation marks.

A **number** is one kind of token which may be passed to Essbase by MaxL. To have meaning, the number must be in the correct format for the Essbase value it represents. In the MaxL grammar documentation, labels for numbers indicate whether the allowed number is positive, negative, an integer, or a real. See Numbers in MaxL Syntax.

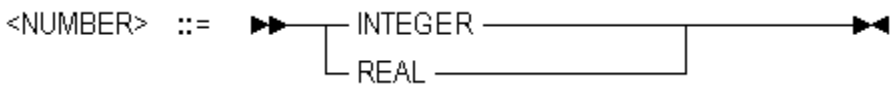
The MaxL **alphabet** consists of the following elements:

Table 3-11 MaxL Alphabet Elements

Element	Description
Special characters	Valid special characters: **. , ; :** % $ " ' SPACE TAB * + - = < > [ ] { } ( ) ? ! **/ \ | ~** ` # & @ **^** When using special characters in MaxL terminals, note the quoting rules (see Quoting and Special Characters Rules for MaxL Language).
Non-special characters	Alphabetic characters and numbers.
Alphabetic characters	Letters of the alphabet, and the underscore. [a-z, A-Z, _]
Numbers	See Numbers in MaxL Syntax

Numbers in MaxL Syntax

Numbers in MaxL statements fit into one of the following categories.



- INTEGER—Zero or a positive integer. Decimals and scientific notation are permitted. Examples: 0, 1, 1000, 1.3e4
- REAL—Zero or a positive real number. Decimals and scientific notation are permitted. Examples: 0.0, 1, 1000, 1000.4, 13.1e-4

Terminals

The following sections describe terminals in alphabetical order.

ACTION

The required action if a data-monitoring trigger is activated.

Syntax

```
mail [smtp],[sender],[receiver1,reciever2,...],[subject]
spool FILE-NAME
```

- mail - sends an email from the specified sender, to a specified email address or addresses, with the specified subject line (optional). Enclose email addresses containing special characters in square brackets ([]). The mail action is not supported for after-update triggers, which are the only triggers available for use with aggregate storage cubes.
- spool - logs a message in a specified file in the `\trig` folder under the cube directory.

Type

string (see MaxL Syntax Notes)

Example

```
mail manager.sales.com, [mktdir@CC.com, Monitor@acnts.com]

spool "trgmonitor"
```

Referenced By

create trigger

drop trigger

ALT-NAME-SINGLE

The name of an alias table. If the name contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Type

name (see MaxL Syntax Notes)

Example

```
Region
'Long Names'
```

Referenced By

alter database

query database

APP-NAME

The name of the application.

The application name must not exceed 8 bytes (non-Unicode-mode applications) or 30 characters (Unicode-mode applications). Avoid using spaces. Application names are not case-sensitive.

If the name contains any allowed special characters, it must be enclosed in single or double quotation marks. Only the following special characters are allowed by Essbase within application names:

```
% (percent sign)

$ (dollar sign)

- (minus sign)

{ (open brace)

} (close brace)

( (open parenthesis)

) (close parenthesis)

! (exclamation mark)

~ (tilde)

` (accent mark)

# (pound sign)

& (ampersand)

@ (at sign)

^ (caret)
```

Type

name (see MaxL Syntax Notes)

Example

```
Sample
```

Referenced By

alter application

alter partition

alter system

create application

display application

display calculation

display database

display location alias

display lock

display object

display session

display trigger spool

drop application

drop lock

grant

query application

AREA-ALIAS

A shorthand name used in the in the create partition statement for referring to an already-specified member expression that designates which areas of the databases should be partitioned.

Type

name (see MaxL Syntax Notes)

Example

In the create partition statement below, "foo" is an area-alias for the member expression specified in the area specification. To create area-aliases, enter the alias names after the member expression in each area specification. To specify which area is relevant when mapping members (if applicable), refer to its alias name in the **mapped** phrase.

In the example below, the alias name as *created* is shown in this color, and it specifies which area (in other words, it refers to the entire member expression string, `'@IDESCENDANTS(East) @IDESCENDANTS(Qtr1)'`). The alias name as *referenced* is shown in this color.

```
create or replace replicated partition sampeast.east
        area '@IDESCENDANTS("Eastern Region"), @IDESCENDANTS(Qtr1)'
to samppart.company at aspen
```

```
as admin identified by 'password'
        area '@IDESCENDANTS(East) @IDESCENDANTS(Qtr1)' foo
        mapped foo (Year) to (Yr)
update allow validate only;
```

> **Note:**
>
> All area aliases used in a mapping should be associated with the target (as in the example above), and the direction of member names listed in the mapped clause should go from source to target.

Referenced By

create partition

BUFFER-ID

A number between 1 and 999,999 inclusive. To destroy a buffer before a data load is complete, you must use the same BUFFER-ID number that was used to initialize the buffer.

Type

number (see MaxL Syntax Notes)

Referenced By

alter database

CALC-NAME

A stored calculation.

Syntax

- **Syntax for database-level calculation:**

  name1.name2.name3

- **Syntax for application-level calculation:**

  name1.name3

- *name1*—Application name.
- *name2*—Database name (not required for application-level calcs).
- *name3*—Calculation script name.

Type

name (see MaxL Syntax Notes)

For calculations associated with databases, three tokens are required, to indicate application and database context and the calculation name.

Example

```
Sample.basic.'alloc.csc'
```

For application-level calculations, two tokens are required, indicating application context and the calculation name. When executing application-level calculations, you must specify which database to calculate using the syntax 'on database STRING.'

Example

- `Sample.'alloc.csc'` is the application-level CALC-NAME.
- `execute calculation Sample.'alloc.csc' on database Basic;` is a way to execute the application-level calculation on a database.

If any part of the name contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Referenced By

create calculation

display calculation

drop calculation

execute calculation

grant

CALC-NAME-SINGLE

A stored calculation name that is the third token of a database-level CALC-NAME.

If any part of the name contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Type

name (see MaxL Syntax Notes)

Example

If the full database-level calc name is `sample.basic.'alloc.csc'`, then CALC-NAME-SINGLE is `'alloc.csc'`.

Referenced By

alter database

CALC-STRING

A calculation string. The body of an anonymous (unstored) calculation, or the string used to specify the body of a stored calculation at create time.

Because calculations are terminated with a semicolon, and semicolons are special characters to MaxL, CALC-STRING should be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Example

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

Referenced By

alter database

execute calculation

COLUMN-WIDTH

A number (at least 8) representing character-width of columns; or, the keyword **default**, representing 20 characters wide.

Type

number (see MaxL Syntax Notes) or **default**

Example

```
set display column width 80
```

```
set display column width default
```

Referenced By

Set Display Column Width

COMMENT-STRING

A string of user-defined informational text. If the string contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Example

```
'This is a comment.'
```

Referenced By

alter application

alter database

create application

create database

create partition

CONDITION

A numeric-value-expression developed in MDX. Must be enclosed in double quotation marks. Enclose strings containing special characters in square brackets ([]).

Type

string (see MaxL Syntax Notes)

Example

```
"Jan>20"
```

Referenced By

create trigger

CUBE-AREA or MDX-SET

A cube area or other specification developed in MDX as a symmetric, syntactically-valid set. The area specification must be static, for example it cannot contain Dynamic Calc members or runtime functions such as Filter, TopSum, or BottomSum. Enclose strings containing special characters in square brackets ([]).

Type

string (see MaxL Syntax Notes)

Examples

The following is a set of siblings.

```
'{[Jan 2000], [Feb 2000], [Mar 2000]}'
```

The following is a crossjoined set.

```
'{([Qtr1], [New York]), ([Qtr1], [California]),
 ([Qtr2], [New York]), ([Qtr2], [California])}'
```

The following set is also a tuple.

```
'{(Jun, FY2011, Actual)}'
```

The following statement clears data from a region of ASOsamp.Sample. The region is defined using a CUBE-AREA expressed in MDX.

```
alter database ASOsamp.sample clear data in region '{(Coupon, [Prev
Year], South)}' physical;
```

Referenced By

create trigger

alter database (aggregate storage)

execute allocation

execute calculation (aggregate storage)

DATE

A valid date string formatted according to these rules:

- `MM/DD/YYYY` or `MM/DD/YY`
- Any character can be used as a separator; for example, `MM~DD~YY` is valid.

If the string contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Example

```
'04/16/03'
'04.16.2003'
04_16_2003
```

Referenced By

alter database

query database

DBS-EXPORT-DIR

Suffix for the name of a cube directory to contain export files, to be created (upon export lro) in the application directory as `appname-dbname-suffix`.

After LRO export, the directory contains file-type LRO binary files (if applicable to the database), and the LRO-catalog export file with file-extension *.exp*.

If for a Sample.Basic export, DBS-EXPORT-DIR is given as `lros`, then the `sample-basic-lros` directory is created in the application directory. The `sample-basic-lros` directory contains file-type LRO binary files and the LRO-catalog export file `'sample-basic-lros.exp'`.

Notes:

- MaxL creates exactly one export directory; it does not create a directory *structure*.
- If the specified export directory already exists, the export LRO statement fails, as a safeguard against overwriting.

Type

string (see MaxL Syntax Notes)

Referenced By

export lro

DBS-NAME

The name of a database. Two tokens are required, to indicate application context.

Syntax

name1.name2

- *name1*—The name of the application containing the database.

  The application name must not exceed 8 bytes (non-Unicode-mode applications) or 30 characters (Unicode-mode applications). Avoid using spaces.
- *name2*—The name of the database.

  The database name must not exceed 8 bytes (non-Unicode-mode applications) or 30 characters (Unicode-mode applications). Avoid using spaces.

Database names are not case-sensitive.

If the name contains any allowed special characters, it must be enclosed in single or double quotation marks. Only following special characters are allowed by Essbase within database names:

```
% (percent sign)
$ (dollar sign)
- (minus sign)
{ (open brace)
} (close brace)
( (open parenthesis)
) (close parenthesis)
! (exclamation mark)
~ (tilde)
` (accent mark)
# (pound sign)
& (ampersand)
@ (at sign)
^ (caret)
```

Type

name (see MaxL Syntax Notes)

Example

```
Sample.basic
```

Referenced By

alter database

alter partition

alter system

alter trigger

create database

create location alias

create outline

create partition

display database

display filter

display filter row

display location alias

display lock

display object

display partition

display session

display trigger spool

display variable

drop database

drop lock

drop partition

drop trigger spool

execute aggregate build

execute aggregate process

execute aggregate selection

export data

grant

import data

import dimensions

import lro

query database

refresh outline

refresh replicated partition

DBS-STRING

The second token of DBS-NAME. Limit 8 characters.

If the name contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Example

```
basic
```

Referenced By

alter application

alter database

alter partition

execute calculation

DIM-NAME

The name of a database dimension.

If the string contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Example

```
Year
Market
```

Referenced By

query database

EXPORT-DIR

Type

string (see MaxL Syntax Notes)

Example

```
'sample-basic-out'
```

Referenced By

alter system

FILE-NAME

A file name or path. If the string contains special characters, it must be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Example

- `file01`
- `"errors.txt"`
- `'/Sample/Basic/expsamp.txt'`

Referenced By

alter database

export data

import data

import dimensions

FILE-NAME-PREFIX

Prefix for one or more file names to be created (upon **display drillthrough DBS-NAME to FILE-NAME-PREFIX**) on the client in the working directory of MaxL execution.

These display output files contain the URL XML content of URL drill-through definitions used to link to content hosted on ERP and EPM applications.

If the string contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Example

```
urlxmls
```

Referenced By

display drillthrough

FILTER-NAME

The name of a security filter. Three tokens are required, to indicate application and database context.

The following special characters are not permitted:

```
! @ # $ % ^ & * () _ + - = {} [] | ; ' : " < > ? , . / ~ `
```

Syntax

name1.name2.name3

- *name1*—Application name.
- *name2*—Database name.
- *name3*—Filter name.

Type

name (see MaxL Syntax Notes)

Example

```
Sample.basic.filt1
```

Referenced By

alter filter

create filter

display filter

display filter row

drop filter

grant

FULL-EXPORT-DIR

Full path for the name of a directory for LRO export files,to be created (upon export lro) anywhere on the client or server.

After export lro, the directory contains file-type LRO binary files (if applicable to the database), and the LRO-catalog export file named in the format *directoryname*`.exp`.

For example, if for a Sample.Basic export, FULL-EXPORT-DIR is given as `home/temp/lros`, then the `lros` directory structure is created under `home/temp` if `home/temp` exists.

The `lros` subdirectory contains file-type LRO binary files and the LRO-catalog export file `'lros.exp'`.

Notes:

- MaxL creates exactly one export directory; it does not create a directory *structure*. In the above example, if the `home/temp` directory structure exists, MaxL creates the `lros` directory as a subdirectory of `home/temp`, but if `home/temp` does not exist, MaxL will not create `home/temp/lros`.
- If the specified export directory already exists, the export LRO statement will fail. This is a safeguard against overwriting existing export directories.

Type

string (see MaxL Syntax Notes)

Referenced By

Export LRO

GROUP-NAME

The name of the Essbase security group.

Group name guidelines:

- Non-Unicode application limit: 256 bytes
- Unicode-mode application limit: 256 characters
- Group names must start with a letter or a number
- The following special characters are not permitted:

  ```
  . ; , = + * ? [ ] |< > \ " ' / [Space] [Tab]
  ```

- If the group name contains any special characters (see MaxL Syntax Notes), the name must be enclosed in single or double quotation marks.

Types

- name (see MaxL Syntax Notes)
- name@provider
- WITH IDENTITY ID-STRING

> **Note:**
>
> If a user or group name includes the @ character, you must specify the provider as well. For example, if you want to log in user admin@msad which is on a Native Directory provider, you must specify 'admin@msad@Native Directory'.

Examples

```
Sales010
```

```
Sales010@Native Directory
```

```
with identity "native://
nvid=f0ed2a6d7fb07688:5a342200:1265973105c:-7f46?GROUP"
```

Referenced By

alter application

display privilege

grant

HOST-NAME

Use the discovery URL instead of a host name. A discovery URL is the URL provided by your Service Administrator, with `/agent` appended to the end. For example, `https://myEssbase2.oraclecloud.com/essbase/agent`.

Leading or trailing spaces will be trimmed off. Maximum length is 1024 bytes (non-Unicode application) or characters (Unicode application).

ID-RANGE

A comma-separated list of sequence ID ranges for logged sequential transactions. A range can consist of:

- A single transaction: *n* to *n*; for example, 1 to 1
- Multiple transactions: *x* to *y*; for example, 20 to 100

Type

string (see MaxL Syntax Notes)

Example

```
1 to 10,20 to 100
```

Referenced By

alter database

ID-STRING

Unique identity attribute identifying a user or group in a directory.

To find the identities of existing users or groups, use display user or display group.

Example

```
native://nvid=f0ed2a6d7fb07688:5a342200:1265973105c:-7f46?USER
```

Referenced By

USER-NAME

GROUP-NAME

IMPORT-DIR

A string representing the full path to the directory used in the **export lro** statement.

> **Note:**
>
> If importing lros from a server directory (using **from server** syntax of **import lro**), you can give just the full directory name instead of the full path, as specified by EXPORT-DIR.

The string must be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Example

```
'home/exports/temp/sample-basic-lros'
```

For information about how IMPORT-DIR is created, see the grammar and definitions for export lro.

Referenced By

import lro

IMP-FILE

A name or path to a server-side rules file or data file, used for import data and import dimension statements.

If the data or rules file is specified to be on the server, the following rules apply. If the data or rules file is specified to be local (or left unspecified, in which case it is also local), skip the following and use FILE-NAME.

If you are using **server data_file** or **server rules_file**, you can get the file from any application (not just the current application) by starting the IMP-FILE string using the following pattern:

```
FILE_SEP AppName FILE_SEP DbName FILE_SEP rest_of_file_name
```

where `FILE_SEP` must be `/`.

Type

name (see MaxL Syntax Notes)

Examples

Consider the MaxL statement:

```
import database demo.basic data
from server rules_file 'IMP-FILE'
on error abort;
```

If IMP-FILE is `'calcdat.txt'` or `'/Demo/Basic/calcdat.txt'`, the file will be looked for in the Demo.Basic cube directory.

```
import database demo.basic data
from server file '/Sample/Basic/Calcdat.txt'
on error abort;
```

Essbase looks for calcdat.txt inside the Sample.Basic cube directory, and loads the data to Demo.Basic.

Referenced By

import data

import dimensions

LOCATION-ALIAS-NAME

The name of a location alias referencing another database.

Syntax

`name1.name2.name3`

- *name1*—Application name.
- *name2*—Database name.
- *name3*—Location alias name.

Type

name (see MaxL Syntax Notes)

Example

```
Sample.Basic.EasternDB
```

Referenced By

create location alias

display location alias

LOC-ALIAS-SINGLE

The single form of a location alias name. Use if you are creating a new location alias.

Type

name (see MaxL Syntax Notes)

Example

```
EasternDB
```

Referenced By

alter database

create location alias

LOG-TIME

A specific log time after which to replay subsequent transactions. Enclose the value in quotation marks.

Type

string (see MaxL Syntax Notes)

Example

```
'11_20_2007:12:20:00'
```

Referenced By

alter database

ALLOC-NUMERIC

An MDX numeric value expression used to specify the amount for an allocation source. The amount value is allocated to cells in the target region. The allocation numeric is one of the following:

- An MDX tuple
- A number
- An arithmetic expression using member names, with the following restrictions:
  - All members in the expression must be from the same dimension.
  - Tuples cannot be used.
  - Only arithmetic operators (+, -, /, and *) can be used.
  - MDX functions (such as Avg and Parent) are not allowed.

Type

string (see MaxL Syntax Notes)

Examples

- `(Acc_1000, Jan_2009)`
- `100.00`
- `(Acc_1000 + Acc_2000)/2`
- `AcctA + AcctB`
- `Balance * 1.1`

Referenced By

execute allocation

MEMBER-EXPRESSION

Outline member specification of members from one or more dimensions, member combinations separated by commas, or member sets defined with functions. Must be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Example

```
'@ANCESTORS(Qtr2)'
```

If MEMBER-EXPRESSION contains MEMBER-NAMES that begin with numbers or contain special characters, enclose those member names in double quotation marks, and the entire MEMBER EXPRESSION in single quotation marks. For example:

- `create or replace filter demo.basic.numfilt no_access on '"2"';`
- `'@DESCENDANTS("Eastern Region"), @CHILDREN(Qtr1)'`

The following example shows how create drillthrough uses a member expression to define the list of drillable regions.

```
create drillthrough sample.basic.myURL from xml_file "temp.xml" on
{'@Ichildren("Qtr1")', '@Ichildren("Qtr2")'} level0 only;
```

Referenced By

alter filter

create filter

create partition

create drillthrough

alter drillthrough

MEMBER-NAME

The name of a database outline member.

If the name contains special characters (see MaxL Syntax Notes), it must be enclosed in single quotation marks.

Type

name (see MaxL Syntax Notes)

Example

```
Jan


'New York'
```

If MEMBER-NAME is part of MEMBER-EXPRESSION and MEMBER-NAME begins with a number or contains special characters (see MaxL Syntax Notes), enclose MEMBER-NAME in double quotation marks and enclose MEMBER-EXPRESSION in single quotation marks.

Referenced By

alter database

create partition

query database

OBJ-NAME

The name of a database object. Three tokens are required, to indicate application and database context.

Syntax

name1.name2.name3

- *name1*—Application name.
- *name2*—Database name.
- *name3*—Object name.

Type

name (see MaxL Syntax Notes)

Example

```
Sample.basic.Calcdat
```

Referenced By

alter object

drop object

OBJ-NAME-SINGLE

A stored database object name that is the third token of a database-level OBJ-NAME.

If any part of the name contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Type

name (see MaxL Syntax Notes)

Example

If the full database object name is `sample.basic.calcdat`, then OBJ-NAME-SINGLE is `calcdat`.

Referenced By

alter object

OUTLINE-ID

The numeric identification of an aggregate storage outline associated with a view. The outline ID is returned by the execute aggregate selection statement. The **execute aggregate selection** statement returns a set of views, including the outline ID for the views it returns.

Type

number (see MaxL Syntax Notes)

Example

```
4142187876
```

Referenced By

execute aggregate selection

execute aggregate build

PASSWORD

A user's password. Not applicable for externally authenticated users.

Password guidelines:

- Non-Unicode application limit: 100 bytes
- Unicode-mode application limit: 100 characters

- If the string contains special characters (see MaxL Syntax Notes), the password must be enclosed in single or double quotation marks
- Leading or trailing spaces are illegal and will be trimmed off

Type

string (see MaxL Syntax Notes)

Referenced By

alter partition

create location alias

create outline

create partition

MaxL Shell Invocation

PATHNAME_FILENAME

A path to a file. If the string contains special characters (see MaxL Syntax Notes), it must be enclosed in single or double quotation marks.

Type

string (see MaxL Syntax Notes)

Referenced By

query database

PRECISION-DIGITS

An integer between 0 and 15, inclusive.

Type

number (see MaxL Syntax Notes)

Referenced By

alter session

PROPS

Aggregate storage data load properties that determine how missing and zero values, duplicate values, and multiple values for the same cell in the data source are processed.

- `ignore_missing_values`: Ignore missing values in the data source.
- `ignore_zero_values`: Ignore zeros in the data source.
- `aggregate_use_last`: Combine duplicate cells by using the value of the cell that was loaded last into the data load buffer. When using this option, data loads are significantly slower, even if there are not any duplicate values.

> **Caution:**
>
> The `aggregate_use_last` method has significant performance impact, and is not intended for large data loads. If your data load is larger than one million cells, consider separating the numeric data into a separate data load process (from any typed measure data). The separate data load can use `aggregate_sum` instead.

- `aggregate_sum`: (Default) Add values when the buffer contains multiple values for the same cell.

If you use multiple properties and any conflict occurs, the last property listed takes precedence.

Type

string (see MaxL Syntax Notes)

Referenced By

alter database (aggregate storage)

RNUM

Resource usage specification for temporary aggregate storage data load buffer.

Must be a number between .01 and 1.0 inclusive. If not specified, the default value is 1.0. Only two digits after the decimal point are significant (for example, 0.029 is interpreted as 0.02). The total resource usage of all load buffers created on a database cannot exceed 1.0 (for example, if a buffer of size 0.9 exists, you cannot create another buffer of a size greater than 0.1). Send operations internally create load buffers of size 0.2; therefore, a load buffer of the default size of 1.0 will cause send operations to fail because of insufficient load buffer resources.

Type

number (see MaxL Syntax Notes)

Example

```
0.02
```

Referenced By

alter database (aggregate storage)

RTSV-LIST

A string of runtime substitution variables that can be used in calculation scripts. Runtime substitution variables are specified as key/value pairs. The string must be enclosed with single quotation marks, and key/value pairs must be separated by a semicolon, including a semicolon after the last runtime substitution variable in the string and before the terminal single quotation mark.

Runtime substitution variables—name and default value—must be declared in the SET RUNTIMESUBVARS calculation command. If you include a runtime substitution variable in RTSV-LIST that has not been declared in SET RUNTIMESUBVARS, Essbase ignores the undeclared runtime substitution variable (no warnings or exceptions are generated).

Type

string (see MaxL Syntax Notes)

Example

In this example of a runtime substitution variable string, the name and value of four runtime substitution variables are specified (for example, the value of the runtime substitution variable named "a" is 100):

```
'a=100;b=@CHILDREN("100");c="Actual"->"Final";d="New York";'
```

Referenced By

execute calculation (block storage only)

RULE-FILE-NAME

A comma separated list of strings of rules-file names. Each rules-file name should be an 8-character object file name with no extension. The rule files must reside on the Essbase server.

Type

string (see MaxL Syntax Notes)

Example

```
'h1h1h1' , 'h1h1h2'
```

Referenced By

import data (aggregate storage)

SESSION-ID

The unique session ID. This ID can be used to logout a user session, or kill the current request in that session.

Type

number (see MaxL Syntax Notes)

Example

```
3310545319
```

Referenced By

alter system

display session

query database

SIZE-STRING

Syntax

number units

OR

number

- *number*—Any positive number. Decimals and scientific notation are permitted. Whitespace between *number* and *units* is optional.
- *units*—One of the following: b, kb, mb, gb, tb (case-insensitive).If units are unspecified, bytes are assumed.

Type

number (see MaxL Syntax Notes)

Examples

```
51040b
51040 b
11MB
11000kb
12.34gb
1234e-2gb
```

Referenced By

alter application

alter database

alter tablespace

SPOOL-NAME

The name of a trigger's output file, as specified in the THEN or ELSE section of the create trigger statement.

Syntax

name1.name2.name3

Type

name (see MaxL Syntax Notes)

Example

In the following create trigger statement, the **bold** section is the spool name.

```
create or replace trigger Sample.Basic.Trigger_Jan_20
where "(Jan,Sales,[100],East,Actual)"
when Jan > 20 and is(Year.currentmember,Jan) then
spool Trigger_Jan_20
end;
```

Referenced By

display trigger spool

drop trigger spool

STOPPING-VAL

Optional stopping value for the execute aggregate process statement. Use this value to give the ratio of the growth size you want to allow during the materialization of an aggregate storage database, versus the pre-aggregation size of the database (Before an aggregation is materialized, the database contains only level 0 input-level data.)

Type

number (see MaxL Syntax Notes)

Example

A stopping value of 1.5 means that during the materialization of the aggregation, the aggregate cells are allowed to occupy up to 50% of the disk space occupied by the level-0 data.

Referenced By

execute aggregate selection

execute aggregate process

TABLSP-NAME

The name of a tablespace. Tablespaces are applicable only to aggregate storage databases. For this release, possible names for tablespaces you can alter are `default` and `temp`. Other tablespace names reserved by the system are `metadata` and `log`.

Syntax

name1.name2

- *name1*—Application name.
- *name2*—Tablespace name.

Type

name (see MaxL Syntax Notes)

Example

```
temp
```

Referenced By

alter tablespace

display tablespace

TRIGGER-NAME

The name of the trigger device created to track and respond to database updates. Trigger names must be triple names, specifying application name, database name, and trigger name (if you rename the application or database, the trigger is invalidated). Trigger names are case-insensitive, are a maximum of 30 bytes, and cannot contain special characters.

Syntax

name1.name2.name3

- *name1*—Application name.
- *name2*—Database name.
- *name3*—The name of the trigger.

Type

name (see MaxL Syntax Notes)

Example

```
Sample.Basic.MyTrigger
```

Referenced By

alter trigger

create trigger

display trigger

drop trigger

URL-NAME

The name of a drill-through URL definition used to link to content hosted on Oracle ERP and EPM applications.

Syntax

name1.name2.name3

- *name1*—Application name
- *name2*—Database name
- *name3*—URL name

Type

name (see MaxL Syntax Notes)

Example

```
Sample.basic.MyURL
```

If any part of the name contains special characters (see MaxL Syntax Notes), the name must be enclosed in single or double quotation marks.

Referenced By

create drillthrough

alter drillthrough

display drillthrough

drop drillthrough

USER-NAME

The name of the user.

User name guidelines:

- Non-Unicode application limit: 256 bytes
- Unicode-mode application limit: 256 characters
- The following special characters are not permitted:

  ```
  ; , = + * ? [ ] |< > \ " ' / [Space] [Tab]
  ```

- If the user name contains any special characters (see MaxL Syntax Notes), the name must be enclosed in single or double quotation marks.

Types

- name (see MaxL Syntax Notes)
- name@provider
- WITH IDENTITY ID-STRING

> **Note:**
>
> If a user or group name includes the @ character, you must specify the provider as well. For example, if you want to log in user admin@msad which is on a Native Directory provider, you must specify 'admin@msad@Native Directory'.

Examples

```
JWSmith


JWSmith@Native Directory


with identity "native://
nvid=f0ed2a6d7fb07688:5a342200:1265973105c:-7f46?USER"
```

Referenced By

alter application

alter database

alter partition

alter system

create location alias

create outline

create partition

display privilege

drop lock

grant

query database

MaxL Shell Invocation

VARIABLE-NAME

The name of the substitution variable. The name can only contain alphanumeric characters and the underscore: `(a-z A-Z 0-9 _)`.

Type

name (see MaxL Syntax Notes)

Example

```
curmonth
```

Referenced By

alter application

alter database

alter system

display variable

VIEW-FILE-NAME

An aggregation script containing information derived during aggregate view selection.

The file is created in the cube directory, with a `.csc` extension.

Aggregation scripts are valid as long as the dimension level structure in the outline has not changed.

Executing an aggregation script (using execute aggregate build) materializes the aggregate views specified within it.

The `.csc` extension is optional when executing the script.

The file name can be a maximum of 8 characters in length (excluding the extension) and must not contain any of the following characters, or whitespace: `:;.,=+*?[]|<>"'\/`

Type

string (see MaxL Syntax Notes)

Referenced By

execute aggregate selection

execute aggregate build

query database

VIEW-ID

The numeric identification of an aggregate view, returned by the execute aggregate selection statement. The concept of views applies only to aggregate storage databases.

VIEW-IDs persist only as long as their associated OUTLINE-IDs. OUTLINE-IDs change when changes are made to the outline.

Type

number (see MaxL Syntax Notes)

Example

```
8941
```

Referenced By

execute aggregate selection

execute aggregate build

VIEW-SIZE

Approximate view size as a fraction of input data size. For example, a view size of `0.5` means that the view is 2X smaller than the input-level view. The concept of views applies only to aggregate storage databases.

Type

number (see MaxL Syntax Notes)

Referenced By

execute aggregate build

Privileges and Roles

Essbase system privileges are indivisible database access types. In MaxL, privileges are grouped together to form permission-sets called *roles*. Privileges themselves are not grantable using MaxL; you typically grant roles, which are the equivalent of privilege levels. The scope of a role can be the system, the application, or the database.

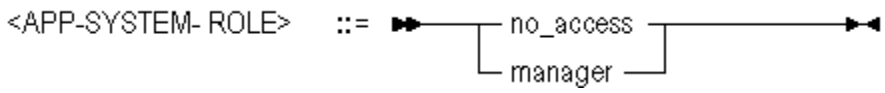
While one privilege does not imply another, roles are hierarchical. The following table illustrates the Essbase system privileges that are contained in each MaxL system role.

Table 3-12 Privileges and Roles

Privileges and Roles	read	write	calculate	manage database	create database	start application	manage application	create/ drop application
no access	.	.	.	.	.	.	.	.
read	■	.	.	.	.	.	.	.
write	■	■	.	.	.	.	.	.
execute	■	■	■	.	.	.	.	.
manager (database)	■	■	■	■	.	.	.	.

Application-Level System Roles

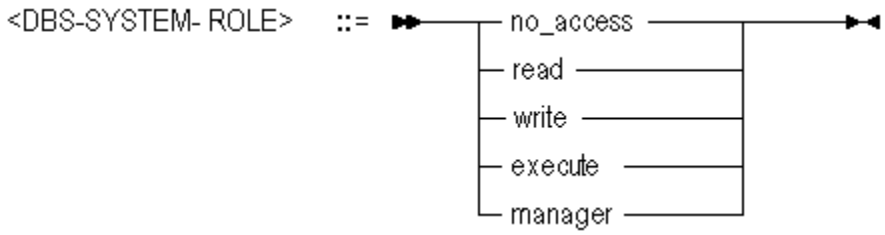
Application-level system roles are applicable to an application. The following roles may have an application-wide scope:



- no_access—No access to the application or any databases within it.
- manager—Manager access to the application and any databases within it. Manager access means ability to create, delete, and modify databases within the application, in addition to having Read, Write, and Execute access for that application.

Database-Level System Roles

Database-level system roles are minimum access permissions you can set for databases. The following roles have a database-wide scope and are available when assigning minimum database permissions:



- no_access—No access to the database (if assigned using alter database) or to any databases in the application (if assigned using alter application).
- read—Read-only access to the database (if assigned using alter database) or to all databases in the application (if assigned using alter application). Read access means ability to view files, retrieve data values, and run report scripts.
- write—Write access to the database (if assigned using alter database) or to all databases in the application (if assigned using alter application). Write access means ability to update data values, in addition to having Read access.
- execute—Calculate access to the database (if assigned using alter database) or to all databases in the application (if assigned using alter application). Calculate access means ability to update data values, in addition to having Read and Write access.
- manager—Manager access to the database (if assigned using alter database) or to all databases in the application (if assigned using alter application). Manager access means ability to modify database outlines, in addition to having Read and Write access.

Quoting and Special Characters Rules for MaxL Language

These rules apply to terminals of MaxL statements; for example, USER-NAME or FILE-NAME. Rules for MaxL Shell also apply (see MaxL Shell Syntax Rules and Variables).

Tokens enclosed in Single Quotation Marks

Contents are preserved as literal, with the following exceptions:

- One backslash is ignored; two are treated as one.
- Apostrophe must be escaped using one backslash (\').

Example: `export database sample.basic data to data_file 'D:\\export.txt';`

Result: Exports data to `D:\export.txt`.

Example: `display user 'O'Brien';`

Result: Error.

Example: `display user 'O\'Brien';`

Result: User `O'Brien` is displayed.

Tokens Enclosed in Double Quotation Marks

Contents are preserved as literal, with the following exceptions:

- Variables are expanded.
- One backslash is ignored; two are treated as one.
- Apostrophe must be escaped using one backslash (\').

Example: `export database sample.basic data to data_file "D:\\export.txt";`

Result: Exports data to `D:\export.txt`.

Example: `export database sample.basic data to data_file "$ARBORPATH\\App\
\Sample\\Basic\\export.txt";`

Result: Exports data to
`C:\Hyperion\products\Essbase\EssbaseServer\App\Sample\Basic\export.txt`.

Example: `display user "O'Brien";`

Result: Error.

Example: `display user "O\'Brien";`

Result: User O'Brien is displayed.

Use of Backslashes in MaxL

Ignored unless preceded by another backslash (the escape character). Must use single or double quotation marks around the token containing the two backslashes.

`create application 'finance\\budget';`

Result: Application `finance\budget` is created.

Example (Windows):

```
export database sample.basic using report_file
'EssbaseServer\\App\\Sample\\Basic\\asym.rep'
to data_file 'c:\\home\\month2.rpt';
```

Result: The Windows file paths are interpreted correctly as `EssbaseServer\App\Sample\Basic\asym.rep` and `c:\home\month2.rpt`.

Use of Apostrophes (Single Quotation Marks)

Syntax error returned, unless preceded by a backslash (the escape character) and enclosed in single or double quotation marks.

Example:`display user 'O\'Brien';`

Result: User `O'Brien` is displayed.

> **Note:**
>
> Use sparingly. Apostrophes are permitted by Essbase in user and group names, but not in application or database names.

Use of Dollar Signs

Syntax error returned, unless preceded by a backslash (the escape character) and enclosed in single quotation marks. Dollar signs ($) intended literally need to be escaped by the backslash so that they are not considered variable indicators.

Example:`create application '\$App1';`

Result: Application `$App1` is created.

MaxL Shell Commands

The MaxL Shell (`essmsh`) is a pre-parser mechanism for entering MaxL statements. The MaxL Shell has a separate set of useful commands, independent of the MaxL language itself. Before using any of the following MaxL Shell commands, you need to log in (see MaxL Shell Invocation).

- Spool on/off
- Set Display Column Width
- Set Message Level
- Set Timestamp
- Echo
- Nesting
- Error Checking and Branching
- Version

- Logout
- Exit

Overview of MaxL Shell

The MaxL Client is a utility through which you execute MaxL statements or scripts.

This section contains the following topics:

- Invocation and Login
- Syntax Rules and Variables
- Shell Commands

MaxL Shell Invocation

The MaxL Shell (`essmsh`) is a pre-parser mechanism for entering MaxL statements.

You can start the shell to be used interactively, to read input from a file, or to read stream-oriented input (standard input from another process). You can log in after you start the shell, interactively or using a login statement in the input file. You can also log in at invocation time, by using the -l flag (see -l Flag: Login).

To start the `essmsh` shell, do not invoke it directly. In order for the environment to be set correctly, you must start `essmsh` using `startMAXL.bat` (Windows) or `startMAXL.sh` (UNIX).

- Prerequisites for Using MaxL
- MaxL Invocation Summary
- Interactive Input
- File Input
- Standard Input
- Login
- LoginAs
- Encryption
- Query Cancellation

Prerequisites for Using MaxL

Before the Essbase Server can receive MaxL statements,

1. The Essbase Server must be running.
2. The MaxL Shell (`essmsh`) must be invoked (see MaxL Invocation Summary), if you are using the shell.
3. You must log in (see Login) to the Essbase Server from the MaxL Shell. If you are running a MaxL script, the first line of your script must be a login statement.

You must use a semicolon (;) to terminate each MaxL statement.

MaxL Invocation Summary

The following MaxL Shell help page summarizes invocation options. This help is also available at the operating-system command prompt if you type `startMAXL.bat -h | more`.

> **Note:**
>
> The following help text is for `essmsh` shell; however, in order for the environment to be set correctly, you must start `essmsh` using `startMAXL.bat` (Windows) or `startMAXL.sh` (UNIX). You can pass the same arguments to `startMAXL` as you would formerly pass to `essmsh`. For example, instead of `essmsh -l username password`, you should now use `startMAXL.bat -l username password`.

```
essmsh(1)

NAME
     essmsh -- MaxL Shell

SYNOPSIS
     essmsh [-hlsmup] [-a | -i | file] [arguments...]

DESCRIPTION
     This document describes ways to invoke the MaxL Shell.
     The shell, invoked and nicknamed essmsh, takes input in the
following
     ways: interactively (from the keyboard), standard input (piped
from another
     program), or file input (taken from file specified on the command
line).
     The MaxL Shell also accepts any number of command-line arguments,
     which can be used to represent any name.

OPTIONS
     essmsh accepts the following options on the command line:

     -h
      Prints this help.

     -l <user> <pwd>
      Logs in a user name and password to the local Essbase Server
instance.

     -u <user>
      Specifies a user to be logged in to an Essbase Server instance.
      If omitted but the '-p' or '-s' flags are used, essmsh will
      prompt for the username.

     -p <pwd>
```

```
      Specifies a password of the user set by the '-u' option to
      be logged in to an Essbase Server instance. If omitted, essmsh
      will prompt for the password, and the password will be hidden
      on the screen.

     -s <server>
      Used after -l, or with [-u -p], logs the specified user into a
named
      server. When omitted, localhost is implied.

     -m <msglevel>
      Sets the level of messages returned by the shell. Values for
<msglevel>
      are: all (the default), warning, error, and fatal.

     -i
      Starts a MaxL session which reads from <STDIN>, piped in from
another program.
      The end of the session is signalled by the EOF character in that
program.

     -a
      Allows a string of command-line arguments to be referenced from
within the
      subsequent INTERACTIVE session. These arguments can be referenced
with positional
      parameters, such as $1, $2, $3, etc. Note: omit the -a when using
arguments with
      a file-input session.

NOTES

     No option is required to pass a filename to essmsh.

     Arguments passed to essmsh can represent anything: for example, a
user name, an
     application name, or a filter name. Arguments must appear at the
end of the
     invocation line, following '-a', '-i', or filename.

EXAMPLES

     Interactive session, simplest case:
          essmsh

     Interactive session, logging in a user:
          essmsh -l user pwd

     Interactive session, logging user in to a server:
          essmsh -l user pwd -s server

     Interactive session, logging in with two command-line arguments
     (referenced thereafter at the keyboard as $1 and $2):
           essmsh -l user pwd -a argument1 argument2
```

```
Interactive session, with setting the message level:
 essmsh -m error

Interactive session, hiding the password:
 essmsh -u user1
 Enter Password > ******

File-input session, simplest case:
     essmsh filename

File-input session, with three command-line arguments
(referenced anonymously in the file as $1, $2, and $3):
     essmsh filename argument1 argument2 argument3


Session reading from <STDIN>, logging into a server with two
command-line arguments:
     essmsh -l user pwd -s server -i argument1 argument2
```

Interactive Input

You can log into the MaxL Shell for interactive use (typing statements at the keyboard) in the following ways. See MaxL Invocation Summary for more descriptions of login flags.

No Flag

-a Flag: Arguments

-l Flag: Login

-u, -p, and -s Flags: Login Prompts and Hostname Selection

-m Flag: Message Level

No Flag

Invoked without a flag, file name, or arguments, the MaxL Shell starts in interactive mode and waits for you to log in. Note to UNIX users: In the following examples, replace `startMAXL.bat` with `startMAXL.sh`.

Example:

```
startMAXL.bat


 Essbase MaxL Shell - Release 11.1.2
 Copyright (c) 2000, 2010, Oracle and/or its affiliates.
 All rights reserved.
MAXL> login Fiona identified by sunflower;
```

```
      49 - User logged in: [Fiona].
```

-a Flag: Arguments

With the -a flag, the MaxL Shell starts in interactive mode and accepts space-separated arguments to be referenced at the keyboard with positional parameters.

> **Note:**
>
> If interactive arguments are used with spooling turned on, variables are recorded in the log file just as you typed them (for example, $1, $2, $ARBORPATH).

Example:

```
startMAXL.bat -a Fiona sunflower appname dbsname



Essbase MaxL Shell - Release 11.1.1
Copyright (c) 2000, 2008, Oracle and/or its affiliates.
All rights reserved.
MAXL> spool on to 'D:\output\createapp.out';

MAXL> login $1 identified by $2;

      49 - User logged in: [Fiona].

MAXL> create application $3;

      30 - Application created: ['appname'].

MAXL> create database $3.$4 as Sample.Basic;

      36 - Database created: ['appname'.'dbsname'].

MAXL> echo $ARBORPATH;

C:\Hyperion\products\Essbase\EssbaseClient

MAXL> spool off;
```

Contents of logfile createapp.out:

```
MAXL> login $1 identified by $2;

 OK/INFO - 1051034 - Logging in user Fiona.
 OK/INFO - 1051035 - Last login on Friday, January 18, 2008 4:09:16 PM.
 OK/INFO - 1241001 - Logged in to Essbase.
```

```
MAXL> create application $3;

 OK/INFO - 1051061 - Application appname loaded - connection
established.
 OK/INFO - 1054027 - Application [appname] started with process id
[404].
 OK/INFO - 1056010 - Application appname created.

MAXL> create database $3.$4 as Sample.Basic;

 OK/INFO - 1056020 - Database appname.dbname created.

MAXL> echo $ARBORPATH;

C:\Hyperion\products\Essbase\EssbaseClient

MAXL> spool off;
```

-l Flag: Login

When the -l flag is used followed by a user name and password, the MaxL Shell logs in the given user name and password and starts in interactive or non-interactive mode. The user name and password must immediately follow the -l, and be separated from it by a space.

Example:

```
startMAXL.bat -l Fiona sunflower
```

Entered at the command prompt, this starts the MaxL Shell in interactive mode and logs in user **Fiona**, who can henceforth issue MaxL statements at the keyboard.

-u, -p, and -s Flags: Login Prompts and Hostname Selection

The MaxL Shell can be invoked using -u and -p options in interactive mode, for passing the user name and password to the shell upon startup. To be prompted for both username and password, use the -s option with the host name of the Essbase Server.

- If -s <host-name> is passed to the shell, MaxL will prompt for the user name and password, and the password will be hidden.

```
startMAXL.bat -s localhost
Enter UserName> admin
Enter Password> ********

 OK/INFO - 1051034 - Logging in user admin.
 OK/INFO - 1051035 - Last login on Monday, January 28, 2003
10:06:16 AM.
 OK/INFO - 1241001 - Logged in to Essbase.
```

- If -u <username> is passed to the shell and -p <password> is omitted, MaxL Shell will prompt for the password, and the password will be hidden.

  ```
  startMAXL.bat -u smith
   Enter Password > ******
  ```

- If -p <password> is passed to the shell and -u <username> is omitted, MaxL Shell will prompt for the user name.

  ```
  startMAXL.bat -p password
   Enter Username > smith
  ```

- If -m <messageLevel> is passed to the shell, only the specified level of messages will be returned by the shell.

  ```
  startMAXL.bat -m error
  ```

  Values for <messageLevel> include: **default**, **all**, **warning**, **error**, and **fatal**. The default value is **all** (same as specifying **default**).

-m Flag: Message Level

If -m <messageLevel> is passed to the shell, only the specified level of messages will be returned by the shell.

Example:`startMAXL.bat -m error`

Values for the <messageLevel> include: **default**, **all**, **warning**, **error**, and **fatal**. The default value is **all** (same as specifying **default**).

File Input

You invoke the MaxL Shell to run scripts (instead of typing statements at the keyboard) in the following ways. See MaxL Invocation Summary for a complete description of login flags.

File Only

File Only

File Only

If you type `startMAXL.bat` followed by a file name or path, the shell takes input from the specified file.

Examples:

```
startMAXL.bat
C:\Hyperion\products\Essbase\EssbaseClient\scripts\filename.msh
```

Entered at the command prompt, this starts the shell, tells it to read MaxL statements from a file, and terminates the session when it is finished.

```
startMAXL.bat filename
```

Starts the shell to read MaxL statements from `filename`, located in the current directory (the directory from which the MaxL Shell was invoked).

File with Arguments

If you type `startMAXL.bat` followed by a file name followed by an argument or list of space-separated arguments, essmsh remembers the command-line arguments, which can be referenced as `$1`, `$2`, etc. in the specified file. If spooling is turned on, all variables are expanded in the log file.

Example:

```
D:\Scripts>startMAXL.bat filename.msh Fiona sunflower localhost
```

Starts the shell to read MaxL statements from `filename.msh`, located in the current directory.

Contents of script filename.msh:

```
spool on to $HOME\\output\\filename.out;
login $1 $2 on $3;
echo "Essbase is installed in $ESSBASEPATH";
spool off;
exit;
```

Contents of logfile filename.out:

```
MAXL> login Fiona sunflower on localhost;

      49 - User logged in: [Fiona].

Essbase is installed in C:\Hyperion\products\Essbase\EssbaseClient
```

Standard Input

With the -i flag, essmsh uses standard input, which could be input from another process. For example,

```
program.sh | startMAXL.bat -i
```

When **program.sh** generates MaxL statements as output, you can pipe `program.sh` to `startMAXL.bat -i` to use the standard output of `program.sh` as standard input for essmsh. Essmsh receives input as `program.sh` generates output, allowing for efficient co-execution of scripts.

Example:

```
echo login Fiona sunflower on localhost; display privilege user;|
startMAXL.bat -i
```

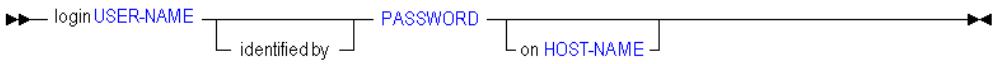
The MaxL Shell takes input from the echo command's output. User Fiona is logged in, and user privileges are displayed.

Login

Before you can send MaxL statements from the MaxL Shell to Essbase Server, you must log in to an Essbase Server session.

As a prerequisite to using MaxL, follow the client setup instructions in Manage Essbase Using the MaxL Client.

> **Note:**
>
> Before logging in to an Essbase Server session, you must start the MaxL Shell (see MaxL Invocation Summary).Or, you can start the MaxL Shell and log in (see -l Flag: Login) at the same time.



- USER-NAME
- PASSWORD
- HOST-NAME

Example

```
login admin pa5sw0rd on "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent";
```

Establishes a connection for user Admin.

LoginAs

To facilitate creating scheduled reports with user-appropriate permissions, administrators can log in as another user from MaxL.

Example of "log in as" statement:

```
loginas USER-NAME PASSWORD MIMICKED-USER-NAME [on HOST-NAME];
```

Example of "log in as" invocation method:

```
essmsh -la USER-NAME PASSWORD MIMICKED-USER-NAME [-s HOST-NAME]
```

Interactive example:

```
MAXL>loginas;
Enter UserName> username
Enter Password> password
Enter Host> machine_name
Enter UserName to Login As> mimicked_user_name
```

Encryption

You can encrypt user and password information stored in MaxL scripts.

The following MaxL Shell invocation generates a public-private key pair that you can use to encrypt a MaxL script.

```
essmsh -gk
```

The following MaxL Shell invocation encrypts the input MaxL script, obscuring user name and password, and changing the file extension to .mxls.

```
essmsh -E scriptname.mxl PUBLIC-KEY
```

Nested scripts are also encrypted. To avoid this and encrypt only the base script, use `-Em`.

The following MaxL Shell invocation decrypts and executes the MaxL script.

```
essmsh -D scriptname.mxls PRIVATE-KEY
```

The following invocation encrypts input data and returns it in encrypted form. This is useful if there is a need to manually prepare secure scripts.

```
essmsh -ep DATA PUBLIC-KEY
```

The following invocation enables you to encrypt the base script while saving any nested scripts for manual encryption.

```
essmsh –Em scriptname.mxl PUBLIC-KEY
```

Query Cancellation

You can use the Esc key to cancel a query running from MaxL Shell.

MaxL Shell Invocation

The MaxL Shell (`essmsh`) is a pre-parser mechanism for entering MaxL statements.

You can start the shell to be used interactively, to read input from a file, or to read stream-oriented input (standard input from another process). You can log in after you start the shell, interactively or using a login statement in the input file. You can also log in at invocation time, by using the -l flag (see -l Flag: Login).

To start the essmsh shell, do not invoke it directly. In order for the environment to be set correctly, you must start essmsh using startMAXL.bat (Windows) or startMAXL.sh (UNIX).

Prerequisites for Using MaxL

Before the Essbase Server can receive MaxL statements,

1. The Essbase Server must be running.
2. The MaxL Shell (essmsh) must be invoked (see MaxL Invocation Summary), if you are using the shell.
3. You must log in (see Login) to the Essbase Server from the MaxL Shell. If you are running a MaxL script, the first line of your script must be a login statement.

You must use a semicolon (;) to terminate each MaxL statement.

MaxL Invocation Summary

The following MaxL Shell help page summarizes invocation options. This help is also available at the operating-system command prompt if you type startMAXL.bat -h | more.

> **Note:**
>
> The following help text is for essmsh shell; however, in order for the environment to be set correctly, you must start essmsh using startMAXL.bat (Windows) or startMAXL.sh (UNIX). You can pass the same arguments to startMAXL as you would formerly pass to essmsh. For example, instead of essmsh -l username password, you should now use startMAXL.bat -l username password.

```
essmsh(1)

NAME
     essmsh -- MaxL Shell

SYNOPSIS
     essmsh [-hlsmup] [-a | -i | file] [arguments...]
```

```
DESCRIPTION
     This document describes ways to invoke the MaxL Shell.
     The shell, invoked and nicknamed essmsh, takes input in the
following
     ways: interactively (from the keyboard), standard input (piped
from another
     program), or file input (taken from file specified on the command
line).
     The MaxL Shell also accepts any number of command-line arguments,
     which can be used to represent any name.

OPTIONS
     essmsh accepts the following options on the command line:

     -h
      Prints this help.

     -l <user> <pwd>
      Logs in a user name and password to the local Essbase Server
instance.

     -u <user>
      Specifies a user to be logged in to an Essbase Server instance.
      If omitted but the '-p' or '-s' flags are used, essmsh will
      prompt for the username.

     -p <pwd>
      Specifies a password of the user set by the '-u' option to
      be logged in to an Essbase Server instance. If omitted, essmsh
      will prompt for the password, and the password will be hidden
      on the screen.

     -s <server>
      Used after -l, or with [-u -p], logs the specified user into a
named
      server. When omitted, localhost is implied.

     -m <msglevel>
      Sets the level of messages returned by the shell. Values for
<msglevel>
      are: all (the default), warning, error, and fatal.

     -i
      Starts a MaxL session which reads from <STDIN>, piped in from
another program.
      The end of the session is signalled by the EOF character in that
program.

     -a
      Allows a string of command-line arguments to be referenced from
within the
      subsequent INTERACTIVE session. These arguments can be referenced
with positional
      parameters, such as $1, $2, $3, etc. Note: omit the -a when using
```

```
arguments with
      a file-input session.

NOTES

     No option is required to pass a filename to essmsh.

     Arguments passed to essmsh can represent anything: for example, a
user name, an
     application name, or a filter name. Arguments must appear at the
end of the
     invocation line, following '-a', '-i', or filename.

EXAMPLES

     Interactive session, simplest case:
          essmsh

     Interactive session, logging in a user:
          essmsh -l user pwd

     Interactive session, logging user in to a server:
          essmsh -l user pwd -s server

     Interactive session, logging in with two command-line arguments
     (referenced thereafter at the keyboard as $1 and $2):
           essmsh -l user pwd -a argument1 argument2



     Interactive session, with setting the message level:
      essmsh -m error

     Interactive session, hiding the password:
      essmsh -u user1
      Enter Password > ******

     File-input session, simplest case:
          essmsh filename

     File-input session, with three command-line arguments
     (referenced anonymously in the file as $1, $2, and $3):
          essmsh filename argument1 argument2 argument3


     Session reading from <STDIN>, logging into a server with two
     command-line arguments:
          essmsh -l user pwd -s server -i argument1 argument2
```

Interactive Input

You can log into the MaxL Shell for interactive use (typing statements at the keyboard) in the following ways. See MaxL Invocation Summary for more descriptions of login flags.

No Flag

-a Flag: Arguments

-l Flag: Login

-u, -p, and -s Flags: Login Prompts and Hostname Selection

-m Flag: Message Level

No Flag

Invoked without a flag, file name, or arguments, the MaxL Shell starts in interactive mode and waits for you to log in. Note to UNIX users: In the following examples, replace `startMAXL.bat` with `startMAXL.sh`.

Example:

```
startMAXL.bat



 Essbase MaxL Shell - Release 11.1.2
 Copyright (c) 2000, 2010, Oracle and/or its affiliates.
 All rights reserved.
MAXL> login Fiona identified by sunflower;

      49 - User logged in: [Fiona].
```

-a Flag: Arguments

With the -a flag, the MaxL Shell starts in interactive mode and accepts space-separated arguments to be referenced at the keyboard with positional parameters.

> **Note:**
>
> If interactive arguments are used with spooling turned on, variables are recorded in the log file just as you typed them (for example, `$1`, `$2`, `$ARBORPATH`).

Example:

```
startMAXL.bat -a Fiona sunflower appname dbsname



Essbase MaxL Shell - Release 11.1.1
Copyright (c) 2000, 2008, Oracle and/or its affiliates.
All rights reserved.
MAXL> spool on to 'D:\output\createapp.out';

MAXL> login $1 identified by $2;

      49 - User logged in: [Fiona].

MAXL> create application $3;

      30 - Application created: ['appname'].

MAXL> create database $3.$4 as Sample.Basic;

      36 - Database created: ['appname'.'dbsname'].

MAXL> echo $ARBORPATH;

C:\Hyperion\products\Essbase\EssbaseClient

MAXL> spool off;
```

Contents of logfile createapp.out:

```
MAXL> login $1 identified by $2;

 OK/INFO - 1051034 - Logging in user Fiona.
 OK/INFO - 1051035 - Last login on Friday, January 18, 2008 4:09:16 PM.
 OK/INFO - 1241001 - Logged in to Essbase.

MAXL> create application $3;

 OK/INFO - 1051061 - Application appname loaded - connection
established.
 OK/INFO - 1054027 - Application [appname] started with process id
[404].
 OK/INFO - 1056010 - Application appname created.

MAXL> create database $3.$4 as Sample.Basic;

 OK/INFO - 1056020 - Database appname.dbname created.

MAXL> echo $ARBORPATH;

C:\Hyperion\products\Essbase\EssbaseClient
```

```
MAXL> spool off;
```

-l Flag: Login

When the -l flag is used followed by a user name and password, the MaxL Shell logs in the given user name and password and starts in interactive or non-interactive mode. The user name and password must immediately follow the -l, and be separated from it by a space.

Example:

```
startMAXL.bat -l Fiona sunflower
```

Entered at the command prompt, this starts the MaxL Shell in interactive mode and logs in user **Fiona**, who can henceforth issue MaxL statements at the keyboard.

-u, -p, and -s Flags: Login Prompts and Hostname Selection

The MaxL Shell can be invoked using -u and -p options in interactive mode, for passing the user name and password to the shell upon startup. To be prompted for both username and password, use the -s option with the host name of the Essbase Server.

- If -s <host-name> is passed to the shell, MaxL will prompt for the user name and password, and the password will be hidden.

  ```
  startMAXL.bat -s localhost
  Enter UserName> admin
  Enter Password> ********

   OK/INFO - 1051034 - Logging in user admin.
   OK/INFO - 1051035 - Last login on Monday, January 28, 2003
  10:06:16 AM.
   OK/INFO - 1241001 - Logged in to Essbase.
  ```

- If -u <username> is passed to the shell and -p <password> is omitted, MaxL Shell will prompt for the password, and the password will be hidden.

  ```
  startMAXL.bat -u smith
   Enter Password > ******
  ```

- If -p <password> is passed to the shell and -u <username> is omitted, MaxL Shell will prompt for the user name.

  ```
  startMAXL.bat -p password
   Enter Username > smith
  ```

- If -m <messageLevel> is passed to the shell, only the specified level of messages will be returned by the shell.

  ```
  startMAXL.bat -m error
  ```

Values for <messageLevel> include: **default**, **all**, **warning**, **error**, and **fatal**. The default value is **all** (same as specifying **default**).

-m Flag: Message Level

If -m <messageLevel> is passed to the shell, only the specified level of messages will be returned by the shell.

Example:`startMAXL.bat -m error`

Values for the <messageLevel> include: **default**, **all**, **warning**, **error**, and **fatal**. The default value is **all** (same as specifying **default**).

File Input

You invoke the MaxL Shell to run scripts (instead of typing statements at the keyboard) in the following ways. See MaxL Invocation Summary for a complete description of login flags.

File Only

File Only

File Only

If you type `startMAXL.bat` followed by a file name or path, the shell takes input from the specified file.

Examples:

```
startMAXL.bat
C:\Hyperion\products\Essbase\EssbaseClient\scripts\filename.msh
```

Entered at the command prompt, this starts the shell, tells it to read MaxL statements from a file, and terminates the session when it is finished.

```
startMAXL.bat filename
```

Starts the shell to read MaxL statements from `filename`, located in the current directory (the directory from which the MaxL Shell was invoked).

File with Arguments

If you type `startMAXL.bat` followed by a file name followed by an argument or list of space-separated arguments, essmsh remembers the command-line arguments, which can be referenced as `$1`, `$2`, etc. in the specified file. If spooling is turned on, all variables are expanded in the log file.

Example:

```
D:\Scripts>startMAXL.bat filename.msh Fiona sunflower localhost
```

Starts the shell to read MaxL statements from `filename.msh`, located in the current directory.

Contents of script filename.msh:

```
spool on to $HOME\\output\\filename.out;
login $1 $2 on $3;
echo "Essbase is installed in $ESSBASEPATH";
spool off;
exit;
```

Contents of logfile filename.out:

```
MAXL> login Fiona sunflower on localhost;

      49 - User logged in: [Fiona].

Essbase is installed in C:\Hyperion\products\Essbase\EssbaseClient
```

Standard Input

With the -i flag, essmsh uses standard input, which could be input from another process. For example,

```
program.sh | startMAXL.bat -i
```

When **program.sh** generates MaxL statements as output, you can pipe `program.sh` to `startMAXL.bat -i` to use the standard output of `program.sh` as standard input for essmsh. Essmsh receives input as `program.sh` generates output, allowing for efficient co-execution of scripts.

Example:

```
echo login Fiona sunflower on localhost; display privilege user;|
startMAXL.bat -i
```

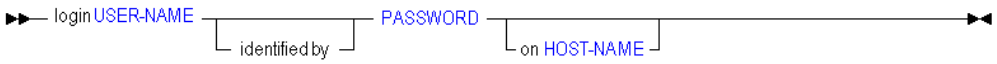
The MaxL Shell takes input from the echo command's output. User Fiona is logged in, and user privileges are displayed.

Login

Before you can send MaxL statements from the MaxL Shell to Essbase Server, you must log in to an Essbase Server session.

As a prerequisite to using MaxL, follow the client setup instructions in Manage Essbase Using the MaxL Client.

> **Note:**
>
> Before logging in to an Essbase Server session, you must start the MaxL Shell (see MaxL Invocation Summary).Or, you can start the MaxL Shell and log in (see -l Flag: Login) at the same time.



- USER-NAME
- PASSWORD
- HOST-NAME

Example

```
login admin pa5sw0rd on "https://myEssbase-
myDomain.analytics.us2.example.com/essbase/agent";
```

Establishes a connection for user Admin.

LoginAs

To facilitate creating scheduled reports with user-appropriate permissions, administrators can log in as another user from MaxL.

Example of "log in as" statement:

```
loginas USER-NAME PASSWORD MIMICKED-USER-NAME [on HOST-NAME];
```

Example of "log in as" invocation method:

```
essmsh -la USER-NAME PASSWORD MIMICKED-USER-NAME [-s HOST-NAME]
```

Interactive example:

```
MAXL>loginas;
Enter UserName> username
Enter Password> password
Enter Host> machine_name
Enter UserName to Login As> mimicked_user_name
```

Encryption

You can encrypt user and password information stored in MaxL scripts.

The following MaxL Shell invocation generates a public-private key pair that you can use to encrypt a MaxL script.

```
essmsh -gk
```

The following MaxL Shell invocation encrypts the input MaxL script, obscuring user name and password, and changing the file extension to .mxls.

```
essmsh -E scriptname.mxl PUBLIC-KEY
```

Nested scripts are also encrypted. To avoid this and encrypt only the base script, use `-Em`.

The following MaxL Shell invocation decrypts and executes the MaxL script.

```
essmsh -D scriptname.mxls PRIVATE-KEY
```

The following invocation encrypts input data and returns it in encrypted form. This is useful if there is a need to manually prepare secure scripts.

```
essmsh -ep DATA PUBLIC-KEY
```

The following invocation enables you to encrypt the base script while saving any nested scripts for manual encryption.

```
essmsh –Em scriptname.mxl PUBLIC-KEY
```

Query Cancellation

You can use the Esc key to cancel a query running from MaxL Shell.

MaxL Shell Syntax Rules and Variables

The MaxL Shell (essmsh) is a pre-parser mechanism for entering MaxL statements. The following syntax information can help you use the MaxL Shell successfully.

Semicolons

Variables

Quoting and Special Characters Rules for MaxL Language

Semicolons

When a MaxL statement is passed to Essbase Server interactively or in batch mode via the MaxL Shell (essmsh), it must be terminated by a semicolon. Semicolons are used only to tell essmsh when to terminate the statement; semicolons are not part of the MaxL language itself. Therefore, when issuing MaxL statements programmatically, do *not* use semicolons.

Examples

Table 3-13 Semicolon Usage Examples in MaxL

Program	Example
Interactive MaxL Shell	`create application Sample;`

Table 3-13 (Cont.) Semicolon Usage Examples in MaxL

Program	Example

MaxL Shell script:	`login $1 identified by $2;`
	`create application Sample;`
	`create currency database`
	`Sample.Interntl;`
	`display database Sample.Interntl;`
	`exit;`

Variables

Overview of MaxL Shell Variables

In the MaxL Shell, you can use **variables** as placeholders for any data that is subject to change or that you refer to often; for example, the name of a computer, user names, and passwords. You can use variables in MaxL scripts as well as during interactive use of the shell. Using variables in MaxL scripts eliminates the need to create many customized scripts for each user, database, or host.

Variables can be environment variables (for example, `$ESSBASEPATH`, which references the directory Essbase is installed to), positional parameters (for example, `$1`, `$2`, etc.), or locally defined shell variables.

All variables must begin with a $ (dollar sign). Locally defined shell variables should be *set* without the dollar sign, but should be *referenced* with the dollar sign. Example:

```
set A = val_1;
echo $A;
val_1
```

> **Note:**
>
> Variables can be in parentheses. Example: if `$1 = arg1`, then `$(1)23 = arg123`.

Use double quotation marks around a string when you want the string interpreted as a single token with the variables recognized and expanded. For example, "$ESSBASEPATH" is interpreted as `C:\Hyperion\products\Essbase\EssbaseServer`.

Use single quotation marks around a string to tell essmsh to recognize the string as a single token, *without* expanding variables. For example, `'$ESSBASEPATH'` is interpreted as `$ESSBASEPATH`, not `C:\Hyperion\products\Essbase\EssbaseServer`.

Environment Variables

You can reference any environment variable in the MaxL Shell.

Example (Windows): `spool on to "$ESSBASEPATH\\out.txt";`

Result: MaxL Shell session is recorded to `C:\Hyperion\products\Essbase\EssbaseServer\out.txt`.

Example (UNIX): `spool on to "$HOME/output.txt";`

Result: MaxL Shell session is recorded to `output.txt` in the directory referenced by the `$HOME` environment variable.

Positional Parameters

Positional parameter variables are passed in to the shell at invocation time as arguments, and can be referred to generically by the subsequent script or interactive MaxL Shell session using $n, where n is the number representing the order in which the argument was passed on the command line.

For example, given the following invocation of the MaxL Shell,

```
essmsh filename Fiona sunflower
```

and the following subsequent login statement in that session,

```
login $1 identified by $2 on $COMPUTERNAME;
```

- `$COMPUTERNAME` is a Windows environment variable.
- `$1` and `$2` refer to the user name and password passed in as arguments at invocation time.

The values of positional parameters can be changed within a session. For example, if the value of `$1` was originally `Fiona` (because essmsh was invoked with `Fiona` as the first argument), you can change it using the following syntax:`set 1 = arg_new;`

> **Note:**
>
> If you nest MaxL Shell scripts or interactive sessions, the nested shell does not recognize positional parameters of the parent shell. The nested shell should be passed separate arguments, if positional parameters are to be used.

The file or process that the MaxL Shell reads from can be referred to with the positional parameter $0. Examples:

```
1) Invocation: essmsh filename
    $0 = filename
2) Invocation: program.sh | essmsh -i
    $0 = stdin
3) Invocation: essmsh
    $0 = null
```

Locally Defined Shell Variables

You can create variables of any name in the MaxL Shell without the use of arguments or positional parameters. These variables persist for the duration of the shell session, including in any nested shell sessions.

Example:

```
MaxL>login user1 identified by password1;
MaxL>set var1 = sample;
MaxL>echo $var1; /* see what the value of $var1 is */
sample
MaxL>display application $var1; /* MaxL displays application "sample" */
```

> **Note:**
>
> Locally defined variables can be named using alphabetic characters, numbers, and the underscore (_). Variable *values* can be any characters, but take note of the usual quoting and syntax rules that apply for the MaxL Shell (see MaxL Shell Syntax Rules and Variables).

> **Note:**
>
> Variables defined or changed in a nested script persist into the parent script after the nested script executes.

Quotation Marks and Variable Expansion

In the following examples, assume you logged in to the MaxL Shell interactively with arguments, as follows. In addition to these examples, see Quoting and Special Characters Rules for MaxL Shell.

```
essmsh -a Fiona sunflower sample basic login $1 $2;
```

Table 3-14 Quotation Marks' Usage and Effect on Variables in MaxL

Example	Return Value	Explanation
echo $1;	`Fiona`	$1 is expanded as the first invocation argument.
echo "$1's hat";	`Fiona's hat`	$1 is expanded as the first invocation argument, and the special character ' is allowed because double quotation marks are used.
echo $3;	`sample`	$3 is expanded as the third invocation argument.
echo '$3';	`$3`	$3 is taken literally and not expanded, because it is protected by single quotation marks.
display database $3.$4;	Database `sample.basic` is displayed.	$3 and $4 are expanded as the third and fourth invocation arguments. `$3.$4` is interpreted as two tokens, which makes it suitable for DBS-NAME.
echo "$3.$4";	`sample.basic`, but interpreted as one token (NOT suitable for DBS-NAME, which requires two tokens).	$3 and $4 are expanded as the third and fourth invocation arguments, but the entire string is interpreted as a single token, because of the double quotation marks.

Exit Status Variable

A successful MaxL Shell operation should have an exit status of zero. Most unsuccessful MaxL Shell operations have an exit status number, usually 1. Exit status can be referred to from within the shell, using `$?`. For example,

```
MAXL> create application test1;
 OK/INFO - 1051061 - Application test1 loaded - connection established.
 OK/INFO - 1054027 - Application [test1] started with process id [234].
 OK/INFO - 1056010 - Application test1 created.
MAXL> echo $?;
0

MAXL> drop application no_such;
   ERROR - 1051030 - Application no_such does not exist.
MAXL> echo $?;
2
```

Quoting and Special Characters Rules for MaxL Shell

These rules are for MaxL Shell commands. Applicable MaxL Shell commands include Spool on/off, Echo, and Nesting.

See Also

Quoting and Special Characters Rules for MaxL Language

Tokens enclosed in Single Quotation Marks

Tokens enclosed in single quotation marks

Contents within single quotation marks are preserved as literal, without variable expansion.

Example: `echo '$3';`

Result: `$3`

Tokens enclosed in double quotation marks

Contents of double quotation marks are treated as a single token, and the contents are perceived as literal except that variables are expanded.

Example: `spool on to "$ESSBASEPATH\\out.txt";`

Result: MaxL Shell session is recorded to `C:\Hyperion\products\Essbase\EssbaseServer\out.txt`.

Example: `spool on to "Ten o'clock.txt"`

Result: MaxL Shell session is recorded to a file named `Ten o'clock.txt`

Use of apostrophes (single quotation marks)

Preserved if enclosed in double quotation marks. Otherwise, causes a syntax error.

Example: `spool on to "Ten o'clock.txt"`

Result: MaxL Shell session is recorded to a file named `Ten o'clock.txt`

Use of Backslashes

Backslashes must be enclosed in single or double quotation marks because they are special characters.

One backslash is treated as one backslash by the shell, but is ignored or treated as an escape character by MaxL. Two backslashes are treated as one backslash by the shell and MaxL.

- `'\ ' = \` (MaxL Shell)
- `'\ ' = (nothing)` (MaxL)
- `'\\' = \\` (MaxL Shell)
- `'\\' = \` (MaxL)

Example: `spool on to 'D:\output.txt'`

Result: MaxL Shell records output to `D:\output.txt`.

Example: `spool on to 'D:\\output.txt'`

Result: MaxL Shell records output to `D:\output.txt`.

Example: `import database sample.basic lro from directory "$ARBORPATH\app\sample-basic-lros";`

Result: Error. Import is a MaxL statement, and for MaxL, `'\'` is ignored.

Example: `import database sample.basic lro from directory "$ARBORPATH\\app\\sample-basic-lros";`

Result: MaxL imports LRO information to Sample.Basic from `$ARBORPATH\app\sample-basic-lros`.

MaxL Shell and Unicode

MaxL Shell is in native mode when started in interactive mode.

MaxL Shell is in native mode when processing a script without a UTF8 byte header.

MaxL Shell is in UTF8 mode when processing a script with the UTF8 byte header.

MaxL Shell Command Reference

The following topics describe the MaxL Shell commands.

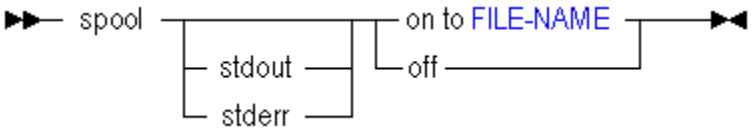
- Spool on/off
- Set display column width
- Set message level
- Set Timestamp
- Echo
- Nesting
- Error Checking and Branching
- Version
- Logout
- Exit

Spool on/off

Log the output of a MaxL Shell session to a file. Send standard output, informational messages, error messages, and/or warning messages generated by the execution of MaxL statements to a file.

If FILE-NAME does not exist, it is created. If FILE-NAME already exists, it is overwritten. If a directory path is not specified for FILE-NAME, FILE-NAME is created in the current directory of the MaxL Shell. Directories cannot be created using the spool command.

Message logging begins with **spool on** and ends with **spool off**.



FILE-NAME

Example

```
spool on to 'output.txt';
```

{MaxL statements}

```
spool off;
```

Sends output of MaxL statements to a file called output.txt, located in the current directory where the MaxL Shell was invoked.

```
spool on to 'c:\hyperion\output.txt';
```

Sends output of MaxL statements to a file called output.txt, located in the pre-existing directory specified by an absolute path.

```
spool on to '../../../output.txt';
```

Sends output of MaxL statements to a file called output.txt, located in the pre-existing directory specified by a relative path. The file would be located three directories above the current directory.

Description

Most operating systems support three channels for input/output:

- STDIN (standard input channel)
- STDOUT (standard output channel)
- STDERR (standard error channel)

Most operating systems also provide command-line options for re-directing data generated by applications, depending on which of the above channels the data is piped through.

Errors in MaxL are flagged as STDERR, allowing command-line redirection of errors using operating-system redirection handles. Non errors are flagged as STDOUT; thus normal output may be logged separately from error output. Here is an example of redirecting error-output at invocation time:

```
essmsh script.mxl 2>errorfile.err
```

> **Note:**
>
> Operating-system redirection handles vary; check the platform documentation.

You can also redirect STDERR and STDOUT independently to different MaxL output logs, using the corresponding options in the `spool` command. For example, you can direct errors to one file and output to another by placing the following lines in your script:

```
spool stdout on to 'output.txt';
spool stderr on to 'errors.txt';
```

or you can direct errors only:

```
spool stderr on to 'errors.txt';
```

or you can direct output only:

```
spool stdout on to 'output.txt';
```

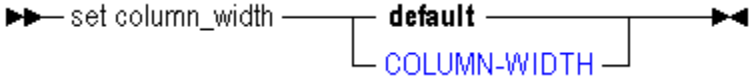
> **Note:**
>
> You cannot use the generic spool and the special output-channel spools in the same script. For example, the following is not valid:

```
spool on to 'session.txt';
spool stderr on to 'errors.txt';
```

Set Display Column Width

Set the width of the columns that appear in MaxL display output tables, for the current MaxL Shell session.

- Default: 20 characters
- Minimum: 8 characters
- Maximum: None.



COLUMN-WIDTH

Example

```
set column_width 10;
```

Sets the column width to 10 characters.

```
set column_width default;
```

Sets the column width back to 20 characters.

Set Message Level

Set the level of messaging you want returned from MaxL Shell sessions. By default, all messages are returned.

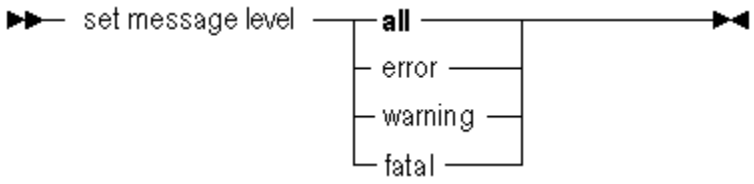


Table 3-15 MaxL Shell Message Levels

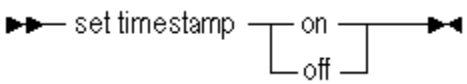
Message level	Description
all	Errors, warnings, status reporting, and informational messages. This is the default message level.
error	Essbase and MaxL Shell error messages.
warning	Essbase warning messages.
fatal	Only errors which cause the shell to disconnect from Essbase.

Example

```
set message level all;
```

Set Timestamp

Enable or disable the display of a timestamp after execution of each MaxL statement. By default, no timestamps are returned.



Notes

The timestamp information does not display after the error-control shell statements goto, iferror, and define.

Example

```
set timestamp on;
```

Echo

Display text or expand variables to the screen or to a log file. When used in scripts with spooling (log-file generation) turned on, echo expands variables in the log file. For interactive sessions, variables are not expanded in the log file; instead, the variable name you typed is recorded (for example, $1).

Syntax

```
echo <text> | <variablename>
```

Example

See examples of echo under the discussion of variables (Quotation Marks and Variable Expansion).

Nesting

Reference (include) a MaxL script from within another MaxL script. You might use this if variables are defined in the referenced MaxL script which are useful to the current MaxL script.

Syntax

msh <scriptfile>;

Example

```
login fiona sunflower;
alter database sample.basic end archive;
msh calculate.msh;
alter database sample.basic
 begin archive to file bak;
 logout;
```

> **Note:**
>
> Variables defined or changed in a nested script persist into the parent script after the nested script executes.

> **Note:**
>
> Because msh is a shell command, it is limited to the originating session. Therefore, you should not reference MaxL scripts that contain new login statements.

Error Checking and Branching

IfError instructs the MaxL Shell to respond to an error in the previous statement by skipping subsequent statements, up to a certain location in the script that is defined by a label name.

IfError checks the presence of errors only in the precedent statement. IfError checks for:

- Errors in MaxL statement execution
- Errors in MaxL Shell command execution, including:
  - Errors in `spool on/off`, such as permission errors
  - Errors in `set column_width`, such as invalid widths
  - Errors in script nesting, such as permission errors or nonexistent include files

Goto forces the MaxL Shell to branch to a certain location in the script defined by a label name; goto is not dependent on the occurence of an error.

Syntax

```
iferror LABELNAME
goto LABELNAME
define label LABELNAME
```

Example: Iferror (MaxL)

The following example script contains a dimension build statement and a data load statement. If the dimension build fails, the data load is skipped.

```
login $1 $2;

import database sample.basic dimensions
 from data_file 'C:\\data\\dimensions.txt'
 using rules_file 'C:\\\\data\\rulesfile.rul'
 on error append to 'C:\\\\logs\\dimbuild.log';

iferror 'dimbuildFailed';

import database sample.basic data from data_file
"$ARBORPATH\\app\\sample\\basic\\calcdat.txt"
 on error abort;

define label 'dimbuildFailed';
exit;
```

Example: Iferror (MaxL Shell)

The following example script tests various errors including MaxL Shell errors, and demonstrates how you can set the exit status variable to a nonzero argument to return an exit status to the MaxL Shell.

```
###  Begin Script ###

login $1 $2;
echo "Testing syntactic errors...";

spool on to spool.out;

set timestampTypo on;
iferror 'End';

msh "doesnotexistlerr.mxl";
iferror 'FileDoesNotExistError';

echo "Script completed successfully...";
spool off;
logout;
exit 0;


define label 'FileDoesNotExistError';
echo "Error detected: Script file does not exist";
spool off;
logout;
exit 1;

define label 'ShellError';
echo ' Shell error detected...';
spool off;
logout;
exit 2;

define label 'End';
echo ' Syntax error detected...';
spool off;
logout;
exit 3;

###  End Script ###
```

Example: Goto

The following example script contains a dimension build statement and a data load statement. Goto is used to skip the data load.

```
login $1 $2;

import database sample.basic dimensions
 from data_file 'C:\\data\\dimensions.txt'
```

```
 using rules_file 'C:\\\\data\\rulesfile.rul'
 on error append to 'C:\\\\logs\\dimbuild.log';

goto 'Finished';

import database sample.basic data from data_file
"$ARBORPATH\\app\\sample\\basic\\calcdat.txt"
 on error abort;

define label 'Finished';
exit;
```

Notes

The MaxL Shell will skip forward in the script to LABELNAME but not backwards.

Version

To see which version of MaxL you are using, type **version**.

Example

```
version;
```

Returns

```
Essbase MaxL Shell - Release 11.1.2
Copyright (c) 2000, 2010, Oracle and/or its affiliates.
All rights reserved.
MAXL>
```

Logout

Log out from Essbase without exiting the interactive MaxL Shell.

Syntax

```
logout;
```

Example

```
logout;
```

Exit

Exit from the `MAXL>` prompt after using interactive mode. You can optionally set the exit status variable to a non zero argument to return an exit status to the parent shell.

> **Note:**
>
> It is not necessary to exit at the end of MaxL script files or stream-oriented input (using the -i switch).

Syntax

```
exit;
```

Example

```
exit;
```

Closes the MaxL Shell window or terminal.

```
exit 10;
```

Closes the MaxL Shell window or terminal with a return status of 10. You can use this in combination with IfError to return a non zero error status to the parent shell.

ESSCMD Script Conversion

cmd2mxl is a fully supported utility for converting existing ESSCMD shell scripts to their corresponding MaxL scripts. To convert an ESSCMD shell script to a MaxL script, go to the operating-system command prompt and enter the executable name, the ESSCMD shell script name, the desired MaxL script name, and the name of a logfile to write to in case of errors.

- ESSCMD Script Utility Usage
- Things to Note About the ESSCMD shell Script Utility
- ESSCMD to MaxL Mapping

ESSCMD Script Utility Usage

```
cmd2mxl esscmd_script maxl_output logfile
```

For example, if the ESSCMD shell script name is `%ARBORPATH%\dailyupd.scr`, the command issued on the operating-system command line would be:

```
cmd2mxl %ARBORPATH%\dailyupd.scr %ARBORPATH%\dailyupd.mxl %ARBORPATH%
\log\dailyupd.log
```

Subsequently, the MaxL script can be executed using the MaxL Shell by the follwing command:

```
essmsh %ARBORPATH%\dailyupd.mxl
```

Things to Note About the ESSCMD shell Script Utility

1. The utility will only translate syntactically and semantically valid ESSCMD shell scripts.
2. For invalid ESSCMD shell scripts, the resulting MaxL script is undefined.
3. All ESSCMD shell statements in the scripts should end with a semicolon (;) statement terminator.
4. This utility will only work on Windows platforms.
5. Although most ESSCMD shell commands have corresponding MaxL statements, there are exceptions. For such exceptions, a comment will be generated in the logfile, and the resulting MaxL script will have to be modified to work correctly. Note that if an ESSCMD shell command is still needed, it can be invoked from a MaxL script using `shell esscmd <scriptname>`.
6. All strings in the ESSCMD shell scripts should be surrounded by double quotation marks ("").

ESSCMD to MaxL Mapping

The following table compares ESSCMD shell usage to MaxL usage, and the following conversions are supported by **cmd2mxl**.

Table 3-16 ESSCMD shell to MaxL Mapping

ESSCMD shell Command	ESSCMD shell Usage Example	MaxL Equivalent Example
ADDUSER	ADDUSER finance essexer1;	N/A. User management statements no longer supported in MaxL.
BEGINARCHIVE	beginarchive sample basic "test.txt";	alter database Sample.Basic begin archive to file 'test.txt';
BEGININCBUILDDIM	beginincbuilddim;	import database Sample.Basic dimensions from local text data_file 'c:\\data.txt' using local rules_file 'c:\ \data_rule.rul' on error write to 'c:\ \error.log';
BUILDDIM	builddim 1 "c:\data_rul.rul" 3 "c:\data.txt" 4 "c:\error.log";	Same as BEGININCDIMBUILD
CALC	calc "CALC ALL;";	execute calculation 'CALC ALL' on sample.basic;
CALCDEFAULT	calcdefault;	execute calculation default on Sample.Basic;
CALCLINE	calcline "CALC ALL;";	execute calculation 'CALC ALL;' on sample.basic;
COPYAPP	copyapp sample sampnew;	create application sampnew as sample;
COPYDB	copydb sample basic sample basic2;	create or replace database sample.basic2 as sample.basic;
COPYFILTER	copyfilter sample basic westwrite sample basic westmgr;	create filter sample.basic.westmgr as sample.basic.westwrite;

Table 3-16 (Cont.) ESSCMD shell to MaxL Mapping

ESSCMD shell Command	ESSCMD shell Usage Example	MaxL Equivalent Example
COPYOBJECT	copyobject "9" "sample" "basic" "calcdat" "sample" "basic" "calcdat2";	alter object sample.basic.calcdat of type text copy to 'sample.basic.calcdat2';
CREATEAPP	createapp finance;	create or replace application finance;
CREATEDB	createdb finance investor;	create or replace database finance.investor;
CREATEGROUP	creategroup managers;	N/A. User management statements no longer supported in MaxL.
CREATELOCATION	select sample basic; createlocation hq hqserver finance investor admin password;	alter system load application sample; alter application sample load database basic; create location alias hq from sample.basic to finance.investor at hqserver as admin identified by 'password';
CREATEUSER	createuser karen password;	N/A. User management statements no longer supported in MaxL.
CREATEVARIABLE	createvariable CurMnth localhost sample basic Jan;	alter database sample.basic add variable CurMnth 'Jan'; alter application sample add variable CurMnth 'Jan'; alter system add variable CurMnth 'Jan';
DELETEAPP	deleteapp sampnew;	drop application sampnew cascade;
DELETEDB	deletedb demo basic;	drop database demo.basic;
DELETEGROUP	deletegroup engg;	N/A. User management statements no longer supported in MaxL.
DELETELOCATION	select finance investor; deletelocation hq1;	alter system load application finance; alter application finance load database investor; drop location alias finance.investor.hq1;
DELETELOG	deletelog sample;	alter application sample clear logfile;
DELETEUSER	deleteuser rob;	N/A. User management statements no longer supported in MaxL.
DELETEVARIABLE	select sample basic; deletevariable CurMnth "localhost";	alter system load application sample; alter application sample load database basic; alter database sample.basic drop variable CurMnth; alter application sample drop variable CurMnth; alter system drop variable CurMnth;
DISABLELOGIN	disablelogin demo;	alter application demo disable connects;
DISPLAYALIAS	select sample basic; displayalias "default";	query database sample.basic list alias_names in alias_table 'Default';

Table 3-16 (Cont.) ESSCMD shell to MaxL Mapping

ESSCMD shell Command	ESSCMD shell Usage Example	MaxL Equivalent Example
ENABLELOGIN	enablelogin demo;	alter application demo enable connects;
ENDARCHIVE	endarchive sample basic;	alter database sample.basic end archive;
ENDINCBUILDDIM	ENDINCBUILDDIM;	See BEGININCBUILDDIM
ESTIMATEFULLDBSIZE	select sample basic; estimatefulldbsize;	query database sample.basic get estimated size;
EXIT	exit;	exit;
EXPORT	select sample basic; export "c:\data.txt" 1;	alter system load application sample; alter application sample load database basic; export database Sample.Basic all data to data_file 'c:\\data.txt';
GETALLREPLCELLS	select samppart company; getallreplcells "svr2" "sampeast" "east";	alter system load application samppart; alter application samppart load database company; refresh replicated partition samppart.company from sampeast.east at svr2;
GETAPPINFO	getappinfo "demo";	display application demo;
GETAPPSTATE	getappstate demo;	display application demo;
GETATTRIBUTESPECS	select sample basic; getattributespecs;	query database sample.basic get attribute_spec;
GETATTRINFO	select sample basic; getattrinfo "Caffeinated_True";	query database sample.basic get attribute_info 'Caffeinated_True';
GETDBINFO	select sample basic; getdbinfo;	display database sample.basic request_history;
GETDBSTATE	getdbstate sample basic;	display database sample.basic;
GETDBSTATS	select sample basic; getdbstats;	query database sample.basic get dbstats data_block;
GETCRRATE	getcrrate;	query database sample.basic get currency_rate;
GETDEFAULTCALC	select sample basic; getdefaultcalc;	query database sample.basic get default calculation;
GETMBRCALC	select sample basic; getmbrcalc "Profit %";	query database sample.basic get member_calculation 'Profit %';
GETMBRINFO	select sample basic; getmbrinfo "Ounces_20";	query database sample.basic get member_info 'Ounces_20';
GETPERFSTATS	select sample basic; getperfstats;	query database sample.basic get performance statistics kernel_cache table;
GETUPDATEDREPLCELLS	See GETALLREPLCELLS	See GETALLREPLCELLS
GETUSERINFO	getuserinfo admin;	display user admin;

Table 3-16 (Cont.) ESSCMD shell to MaxL Mapping

ESSCMD shell Command	ESSCMD shell Usage Example	MaxL Equivalent Example
GETVERSION	getversion;	version;
IMPORT	select sample basic; import 1 "c:\data.txt" 4 y 3 "c:\import.rul" n "c:\data_load.err";	alter system load application sample; alter application sample load database basic; import database sample.basic data from local text data_file 'c:\ \data.txt' using local rules_file 'c:\ \data_rule.rul' on error write to 'c:\ \data_load.err';
INCBUILDDIM	See BEGININCBUILDDIM	See BEGININCBUILDDIM
LISTALIASES	select sample basic; listaliases;	query database sample.basic list alias_table;
LISTAPP	listapp;	display application all;
LISTDB	listdb;	display database all;
LISTFILES	listfiles "" "sample" "basic";	query database sample.basic list all file information;
LISTFILTERS	listfilters sample basic;	display filter on database Sample.Basic;
LISTGROUPS	listgroups;	display group all;
LISTGROUPUSERS	listgroupusers finance;	display user in group finance;
LISTLINKEDOBJECTS	select sample basic; listlinkedobjects "Fiona" "07/07/2003";	query database sample.basic list lro by Fiona before '07/07/2003';
LISTLOCATIONS	select sample basic; listlocations;	alter system load application sample; alter application sample load database basic; display location alias on database sample.basic;
LISTLOCKS	listlocks;	display lock;
LISTLOGINS	listlogins;	display session all;
LISTOBJECTS	listobjects "2" "Sample" "Basic";	display object of type calc_script on database sample.basic;
LISTUSERS	listusers;	display user all;
LISTVARIABLES	listvariables localhost sample basic;	display variable on database sample.basic;
LOADALIAS	select sample basic; loadalias "special_flavors" "C:\Hyperion\products\Essbase\EssbaseServer\app\sample\basic\seasonal.txt";	alter database sample.basic load alias_table 'special_flavors' from data_file "$ARBORPATH\\app\ \sample\\basic\\seasonal.txt";
LOADAPP	loadapp sample;	alter system load application sample;
LOADDB	loaddb sample basic;	alter application sample load database basic;

Table 3-16 (Cont.) ESSCMD shell to MaxL Mapping

ESSCMD shell Command	ESSCMD shell Usage Example	MaxL Equivalent Example
LOADDATA	select sample basic; loaddata 3 "c:\data.txt";	alter system load application sample; alter application sample load database basic; import database sample.basic data from local text data_file 'c:\\data.txt' on error abort;
LOGIN	login local admin password;	login admin 'password' on local;
LOGOUT	logout;	logout;
LOGOUTALLUSERS	logoutallusers y;	alter system logout session all;
LOGOUTUSER	Available only in interactive ESSCMD shell sessions.	alter system logout session 4294967295;
OUTPUT	output 1 c:\test.log; output 4;	spool on to 'c:\test.log'; spool off;
PURGELINKEDOBJECTS	purgelinkedobjects "Fiona" "07/07/2002";	alter database sample.basic delete lro by 'fiona' before '07/07/2002';
PUTALLREPLCELLS	select sampeast east; putallreplcells svr1 samppart company;	alter system load application sampeast; alter application sampeast load database east; refresh replicated partition sampeast.east from samppart.company at svr1 updated data;
PUTUPDATEDREPLCELLS	See PUTALLREPLCELLS	See PUTALLREPLCELLS
REMOVELOCKS	removelocks "2";	drop lock held by Fiona;
REMOVEUSER	removeuser finance steve;	N/A. User management statements no longer supported in MaxL.
RENAMEAPP	renameapp sample newsamp1;	alter application sample rename to newsamp1;
RENAMEDB	renamedb sample basic newbasic;	alter database sample.basic rename to newbasic;
RENAMEFILTER	renamefilter sample basic westmgr allwest;	create or replace filter sample.basic.westmgr as sample.basic.allwest; drop filter sample.basic.westmgr;
RENAMEOBJECT	RENAMEOBJECT "9" "sample" "basic" "calcdat" "calcdat2";	alter object sample.basic.calcdat of type text rename to 'calcdat2';
RENAMEUSER	renameuser steve_m m_steve;	N/A. User management statements no longer supported in MaxL.
RESETDB	select sample basic; resetdb;	alter database sample.basic reset;
RESETPERFSTATS	resetperfstats enable;	alter database sample.basic set performance statistics enabled;
RUNCALC	The only command supported is the server based calc script execution. Select Sample.Basic; Runcalc 2 one;	execute calculation Sample.Basic.one;

Table 3-16 (Cont.) ESSCMD shell to MaxL Mapping

ESSCMD shell Command	ESSCMD shell Usage Example	MaxL Equivalent Example
RUNREPT	select sample basic; runrept 2 complex "c:\complex.out";	alter system load application sample; alter application load database basic; export database sample.basic using server report_file 'complex' to data_file 'c:\\complex.out';
SELECT	select sample basic;	alter system load application sample; alter application load database basic;
SETALIAS	select sample basic; setalias "long names";	alter database sample.basic set active alias_table 'Long Names';
SETAPPSTATE	setappstate sample "" y y 4 y y y y 1000 1000;	alter application sample enable startup; alter application sample enable autostartup; alter application sample set minimum permission manager; alter application sample enable connects; alter application sample enable commands; alter application sample enable updates; alter application sample enable security; alter application sample set lock_timeout after 1000 seconds; alter application sample set max_lro_file_size 1000 kb;
SETDBSTATE	setdbstate "" "Y" "Y" 4 3145728 "Y" "Y" "Y" "" "" 0 1048576 1025 "Y";	alter database sample.basic enable startup; alter database sample.basic enable autostartup; alter database sample.basic set minimum permission manager; alter database sample.basic set data_cache_size 3145728; alter database sample.basic enable aggregate_missing; alter database sample.basic enable two_pass_calc; alter database sample.basic enable create_blocks; alter database sample.basic set currency_conversion division; alter database sample.basic set index_cache_size 1048576; alter database sample.basic enable compression;

Table 3-16 (Cont.) ESSCMD shell to MaxL Mapping

ESSCMD shell Command	ESSCMD shell Usage Example	MaxL Equivalent Example
SETDBSTATEITEM	.	See the alter database statement.
SETDEFAULTCALC	select sample basic; setdefaultcalc "CALC ALL;";	alter database sample.basic set default calculation as 'CALC ALL';
SETDEFAULTCALCFILE	select sample basic; setdefaultcalcfile defcalc;	Create a calculation file in the server containing the calculation string. Then, **alter database sample.sasic set default calculation sample.basic.defcalc;** will set the default calculation.
SETMSGLEVEL	setmsglevel 2;	set message level all;
SETPASSWORD	setpassword steve newpass;	N/A. User management statements no longer supported in MaxL.
SHUTDOWNSERVER	shutdownserver local admin password;	login admin 'password' on local; alter system shutdown;
SLEEP	sleep 10;	shell sleep 10;
UNLOADALIAS	select sample basic; unloadalias "flavors";	alter database sample.basic unload alias_table 'flavors';
UNLOADAPP	unloadapp sample;	alter system unload application sample;
UNLOADDB	unloaddb sample basic;	alter application sample unload database basic;
UNLOCKOBJECT	unlockobject "1" "sample" "basic" "basic";	alter object 'sample.basic.basic' of type outline unlock;
UPDATE	select sample.basic update "Jan Sales '100-10' Florida Actual 220";	import database sample.basic from data_string 'Jan Sales 100-10 Florida Actual 220';
UPDATEFILE	updatefile 3 "c:\data.txt" 1;	same as LOADDATA;

Note:

This is part of the separate MaxL Shell grammar, not the MaxL language itself.

Table 3-16 (Cont.) ESSCMD shell to MaxL Mapping

ESSCMD shell Command	ESSCMD shell Usage Example	MaxL Equivalent Example
UPDATEVARIABLE	updatevariable hot_product local sample basic "100-10";	alter system set variable 'hot_product' '100-10'; alter application sample set variable 'hot_product' "100-10"; alter database Sample.Basic set variable 'hot_product' '100-10';
VALIDATE	validate;	alter database sample.basic validate data to local logfile 'validation.txt';

MaxL Reserved Words List

The following keywords are part of the MaxL grammar, and are reserved. If you intend to use any of these words as names or passwords, you must enclose the word in single quotation marks.

```
abort
absolute_value
account_type
active
add
administrator
advanced
after
aggregate
aggregates
aggregate_assume_equal
aggregate_missing
aggregate_storage
aggregate_sum
aggregate_view
aggregate_use_last
algorithm
alias
alias_names
alias_table
all
all_users_groups
allocation
alloc_rule
allow
allow_merge
alter
alternate_rollups
amount
amountcontext
amounttimespan
any
append
```

```
application
application_access_type
apply
archive
archive_file
area
as
aso_level_info
at
attribute
attribute_calc
attribute_info
attribute_spec
attribute_to_base_member_association
auto_password
autostartup
b
backup_file
based
basis
basistimespan
basistimespanoptions
before
begin
bitmap
blocks
buffer_id
buffered
build
by
cache_pinning
cache_size
calc_formula
calc_script
calc_string
calculation
cascade
cell_status
change_file
clear
client
cnt_sempaphore
column_width
columns
combinebasis
commands
comment
commitblock
committed_mode
compact
compression
compression_info
config_values
connect
connects
```

```
consolidation
copy
copy_subvar
copy_useraccess
create
create_application
create_blocks
create_user
creation
creation_user
creditmember
cube_size_info
currency
currency_category
currency_conversion
currency_database
currency_member
currency_rate
custom
data
data_block
data_cache_size
data_file
data_file_cache_size
data_storage
data_string
database
database_synch
database_asynch
days
dbstats
debitmember
debug
default
definition_only
definitions
delete
designer
destroy
dimension
dimensions
direct
direction
directory
disable
disabled
disallow
discard_errors
disk
display
divideamount
division
drillthrough
dml_output
drop
```

```
dump
dynamic_calc
eas_loc
enable
enabled
encrypted
end
end_transaction
enforce
eqd
error
error_file
errors_to_highest
errors_to_location
errors_to_lowest
estimated
event
exact
excel
exceeds
excludedrange
execute
existing_views
export
export_directory
external
failed_sss_migration
fragmentation_percent
freespace
from
file
file_location
file_size
file_type
filter
filter_access
fixed_decimal
for
force
force_dump
formatted_value
function
gb
get
get_missing_cells
get_meaningless_cells
global
grant
group
group_id
ha_trace
held
high
hostname
identified
```

```
identify
ignore_missing_values
ignore_zero_values
immediate
implicit_commit
import
in
inactive
inactive_user_days
including
incremental
index
index_cache_size
index_data
index_page_size
information
initialize
input
instead
invalid_block_headers
invalid_login_limit
io_access_mode
kb
kernel_io
kernel_cache
kill
level
level0
license_info
linked
list
load
load_buffer
load_buffers
load_buffer_block
local
location
lock
lock_timeout
locked
log_level
logfile
login
logout
long
lotus_2
lotus_3
lotus_4
low
lro
macro
manager
mapped
max_disk_size
max_file_size
```

```
max_lro_file_size
mb
medium
member
member_alias_namespace
member_calculation
member_comment
member_data
member_fixed_length_data
member_formula
member_info
member_name_namespace
member_property
member_uda
member_uda_namespace
member_variable_length_data
merge
meta_read
metadata_only
migr_modified_access
miner
minimum
mining
minutes
missing_value
mode
model
move
multiple
multiplication
mutex
name
negativebasisoptions
never
no_access
none
non_unique_members
nonunicode_mode
note
nothing
numerical_display
object
objects
of
off
offset
on
only
opg_cache
opg_state
optional
optional_group
options
or
outline
```

```
outline_id
outline_paging_file
output
override
overview
partition
partition_file
partition_size
passive
password
password_reset_days
performance
permission
persistence
perspective
physical
pmml_file
ports
pov
pre_image_access
precision
preserve
preserve_groups
private
privilege
process
project
property
protocol
purge
query
query_data
query_tracking
range
read
recover
reference_cube
reference_cube_reg
refresh
region
registration
reregister
remote
remove
remove_zero_cells
rename
repair
repeatamount
replace
replay
replicated
replication_assume_identical
report_file
request
request_history
```

```
request_id
reset
resource_usage
restore
restructure
result
resync
retrieve_buffer_size
retrieve_sort_buffer_size
reverse
revoke
rle
round
row
rows
rules_file
runtime
runtime_info
save
scientific_notation
scope
score
script_file
seconds
security
security_backup
select
selecting
selection
self_session_info
semaphore
sequence_id_range
server
server_port
session
session_idle_limit
session_idle_poll
set
shared_services_native
short
shutdown
single
singlecell
size
size_limit
skip_to_next_amount
skip_missing
skip_negative
skip_zero
slice
sourceregion
spec
spinlock
splitbasis
spread
```

```
SSL
sss
sss_mode
sss_name
starting
startup
statistics
status
stop
stopping
storage
storage_info
structure_file
subtract
supervisor
suppress
sync
system
table
tablespace
target
targettimespan
targettimespanoptions
task
tb
template
text
thread
to
total_size
transactions
transformation
transparent
trigger
trigger_func
trigger_spool
two_pass_calc
type
uda
unicode
unicode_mode
unlimited
unload
unlock
update
updated
updates
use
user
username_as_password
using
validate
values
variable
vector
```

```
verification
version
view_file
views
volume
wait_for_resources
warn
when
with
wizard
worksheet
write
xml_file
zero_value
zeroamountoptions
zerobasisoptions
```

MaxL BNF

MaxL BNF diagrams are an optional alternative to railroad diagrams, for reading MaxL syntax.

Key

```
{}      Alternatives (at least one required)
[]      Options (none required)
!!      Default option if none indicated
|       Separates options (OR)
[,...]  Comma-separated list (of previous item) allowed
[ ...]  Whitespace-separated list (of previous item) allowed
''      Literal
::=     "is defined as." Symbol to the left is to be replaced with
expression on the right
TERMINAL
%NON-TERMINAL%
```

alter application

```
alter application
{APP-Name
 {set
  {lock_timeout after INTEGER[!seconds!|minutes]
   |max_lro_file_size {unlimited|SIZE-STRING}
   |minimum permission %DBS-SYSTEM-ROLE%
   |variable VARIABLE-NAME STRING
   |cache_size SIZE-STRING
   |type unicode_mode
  }
 |{load|unload} database DBS-STRING
 |{enable|disable} {startup|autostartup|commands|updates|connects|
security}
 |comment COMMENT-STRING
```

```
 |clear logfile
 |add variable VARIABLE-NAME [STRING]
 |drop variable VARIABLE-NAME
 |rename to APP-NAME
}

DBS-SYSTEM-ROLE::=
{no_access|read|write|execute|manager}
```

alter application (aggregate storage)

```
alter application
{APP-Name
 {set
  {
   |minimum permission %DBS-SYSTEM-ROLE%
   |variable VARIABLE-NAME STRING
   |cache_size SIZE-STRING
   |type unicode_mode
  }
 |{load|unload} database DBS-STRING
 |{enable|disable} {startup|autostartup|commands|updates|connects|
security}
 |comment COMMENT-STRING
 |clear logfile
 |add variable VARIABLE-NAME [STRING]
 |drop variable VARIABLE-NAME
 |rename to APP-NAME
}

DBS-SYSTEM-ROLE::=
{no_access|read|write|execute|manager}
```

alter database enable|disable

```
alter database DBS-NAME
  {enable|disable}
   {
    two_pass_calc
    |aggregate_missing
    |startup
    |autostartup
    |compression
    |create_blocks
    |committed_mode
    |pre_image_access
   }
```

alter database set

```
alter database DBS-NAME
 set
   {
```

```
    retrieve_buffer_size SIZE-STRING
    |retrieve_sort_buffer_size SIZE-STRING
    |data_cache_size SIZE-STRING
    |index_cache_size SIZE-STRING
    |currency_database DBS-STRING
    |currency_member MEMBER-NAME
    |currency_conversion {division|multiplication}
    |minimum permission %DBS-SYSTEM-ROLE%
    |compression {rle|bitmap}
    |lock_timeout
     {
      immediate
      |never
      |after INTEGER {[!seconds!|minutes]}
     }
    |implicit_commit after INTEGER {blocks|rows}
    |variable VARIABLE-NAME STRING
    |default calculation {CALC-NAME-SINGLE|as calc_string CALC-STRING}
    |active alias_table ALT-NAME-SINGLE
    |performance statistics {enabled|disabled|mode to %PST-SPEC%}
    |note COMMENT-STRING
   }

DBS-SYSTEM-ROLE::=
 {no_access|read|write|execute|manager}

PST-SPEC::=
 {
  default
  |{medium|long} persistence {all|database|server} scope
 }
```

alter database misc

```
alter database DBS-NAME
 {
  reset [{all|data}]
  |validate
   {
    data to local logfile FILE-NAME
    |using {error_file FILE-NAME|default error_file}
   }
  |force restructure
  |load alias_table ALT-NAME-SINGLE from data_file FILE-NAME
  |unload alias_table ALT-NAME-SINGLE
  |add variable VARIABLE-NAME [STRING]
  |drop variable VARIABLE-NAME
  |delete lro
   {
    all
    |by USER-NAME
    |before DATE
    |by USER-NAME before DATE
   }
```

```
  |unlock all objects
  |begin archive to file FILE-NAME
  |end archive
  |[force] archive to file FILE-NAME
  |[force] restore from file FILE-NAME
  |replay transactions
   {
    after LOG-TIME
    |using sequence_id_range ID-RANGE
   }
  |rename to DBS-STRING
  |comment COMMENT-STRING
 }
```

alter database disk volumes

```
alter database DBS-NAME
 {
  {add|drop} disk volume VOLUME-NAME
  |set disk volume VOLUME-NAME
   {
    file_type {data|index|index_data}
    |file_size SIZE-STRING
    |partition_size {SIZE-STRING|unlimited}
   }
 }
```

alter database (aggregate storage)

```
alter database DBS-NAME
 {
  {enable|disable}
   {
    startup
    |autostartup
    |query_tracking
    |replication_assume_identical_outline
   }
  |set
   {
    retrieve_buffer_size SIZE-STRING
    |retrieve_sort_buffer_size SIZE-STRING
    |minimum permission %DBS-SYSTEM-ROLE%
    |variable VARIABLE-NAME STRING
    |active alias_table ALT-NAME-SINGLE
   }
  |reset [{all|data}]
  |compact outline
  |add variable VARIABLE-NAME [STRING]
  |drop variable VARIABLE-NAME
  |%LOAD-BUFFER-INIT%
  |destroy load_buffer with buffer_id BUFFER-ID[,...]
  |unlock all objects
  |rename to DBS-STRING
```

```
  |comment COMMENT-STRING
  |merge {all|incremental} data
  |begin archive to file FILE-NAME
  |end archive
 }

DBS-SYSTEM-ROLE::=
 {no_access|read|write|execute|manager}

LOAD-BUFFER-INIT::=
 initialize load_buffer with buffer_id BUFFER-ID[,...]
  [resource_usage RNUM][property PROPS][wait_for_resources]
```

alter drillthrough

```
alter drillthrough
 URL-NAME from xml_file FILE-NAME
 on '{'MEMBER-EXPRESSION'}'[,...]
 [allow_merge]
```

alter filter

```
alter filter FILTER-NAME
 add {no_access|read|write|meta_read} on MEMBER-EXPRESSION [,...]
```

alter object

```
alter object OBJ-NAME of type %OBJ-TYPE%
 {rename to OBJ-NAME-SINGLE|unlock|[force]copy to OBJ-NAME}

 OBJ-TYPE::=
  outline
 |calc_script
 |report_file
 |rules_file
 |text
 |partition_file
 |lro
 |selection
 |wizard
 |eqd
 |outline_paging_file
 |worksheet
 |alias_table
```

alter partition

```
alter {transparent|replicated} partition DBS-NAME
 {to|from} DBS-NAME [at HOST-NAME]
 set{
  connect as USER-NAME identified by PASSWORD
  |hostname as HOST-NAME instead of HOST-NAME direction {single|all}
```

```
  |application as APP-NAME instead of APP-NAME direction {single|all}
  |database as DSB-STRING instead of DBS-STRING
 }
```

alter session

```
alter session set dml_output
 {
  [
   !default!
   |alias {on|off}
   |metadata_only {on|off}
   |cell_status {on|off}
   |numerical_display {!default!|fixed_decimal|scientific_notation}
   |precision PRECISION-DIGITS]
   |formatted_value {on|off}
   |get_missing_cells {on|off}
   |get_meaningless_cells {on|off}
  [,...]
 }
```

alter system

```
alter system
 {
  load application {all|APP-NAME}
  |unload application {all|APP-NAME} [no_force]
  |set
   {
    session_idle_limit {INTEGER[!seconds!|minutes]|none}
    |session_idle_poll {INTEGER[!seconds!|minutes]|none}
    |invalid_login_limit {INTEGER|none|}
    |inactive_user_days {INTEGER[days]|none}
    |password_reset_days {INTEGER[days]|none}
    |variable VARIABLE-NAME STRING
    |server_port begin at INTEGER end at INTEGER
   }
  |delete export_directory EXPORT-DIR
  |add variable VARIABLE-NAME[STRING]
  |drop variable VARIABLE-NAME
  |logout session %SESSION-SPEC% [force]
  |shutdown
  |kill request %SESSION-SPEC%
  |{enable|disable} unicode
  |reconcile[force]
 }



SESSION SPEC::=
 all
 |SESSION-ID
 |by user USER-NAME
  [
```

```
   on application APP-NAME
   |on database DBS-NAME
  ]
 |on application APP-NAME
 |on database DBS-NAME
```

alter system (aggregate storage)

```
alter system
 {
  load application {all|APP-NAME}
  |unload application {all|APP-NAME} [no_force]
  |set
   {
    session_idle_limit {INTEGER[!seconds!|minutes]|none}
    |session_idle_poll {INTEGER[!seconds!|minutes]|none}
    |invalid_login_limit {INTEGER|none|}
    |inactive_user_days {INTEGER[days]|none}
    |password_reset_days {INTEGER[days]|none}
    |variable VARIABLE-NAME STRING
    |server_port begin at INTEGER end at INTEGER
   }
  |add variable VARIABLE-NAME[STRING]
  |drop variable VARIABLE-NAME
  |logout session %SESSION-SPEC% [force]
  |shutdown
  |kill request %SESSION-SPEC%
  |reconcile [force]
 }
```

```
SESSION SPEC::=
 all
 |SESSION-ID
 |by user USER-NAME
  [
   on application APP-NAME
   |on database DBS-NAME
  ]
 |on application APP-NAME
 |on database DBS-NAME
```

alter tablespace (aggregate storage)

```
alter tablespace TABLSP-NAME
 {
  add file_location FILE-NAME
   [
    set max_file_size SIZE-STRING
    |set max_disk_size SIZE-STRING
```

```
    [,...]
   ]
  |alter file_location FILE-NAME
   [
    set max_file_size SIZE-STRING
    |set max_disk_size SIZE-STRING
    [,...]
   ]
  |drop file_location FILE-NAME
 }
```

alter trigger

```
alter trigger
 {
  TRIGGER-NAME {enable|disable}
  |on database DBS-NAME disable
 }
```

create application

```
create [or replace] application APP-NAME
 [type {!nonunicode_mode!|unicode_mode}]
 [as APP-NAME]
 [comment COMMENT-STRING]
```

create application (aggregate storage)

```
create [or replace] application APP-NAME
 [type {!nonunicode_mode!|unicode_mode}]
 [using aggregate_storage]
 [as APP-NAME]
 [comment COMMENT-STRING]
```

create calculation

```
create [or replace] calculation CALC-NAME {CALC-STRING|as CALC-NAME}
```

create database

```
create [or replace] [currency] database DBS-NAME
 [using non_unique_members]
 [as DBS-NAME]
 [comment COMMENT-STRING]
```

create database (aggregate storage)

```
create [or replace] database DBS-NAME
 [using non_unique_members]
 [comment COMMENT-STRING]
```

create drillthrough

```
create drillthrough URL-NAME from xml_file FILE-NAME
 on '{'MEMBER-EXPRESSION [,...]'}'
 [level0 only]
```

create filter

```
create [or replace] filter FILTER-NAME
 {
  as FILTER-NAME
  |
   {
    no_access
    |read
    |write
    |meta_read
   }
    on MEMBER-EXPRESSION
    [,...]
 }
 [definition_only]
```

create location alias

```
create [or replace] location alias
 {
  LOC-ALIAS-SINGLE from DBS-NAME
  |LOCATION-ALIAS-NAME
 }
 to DBS-NAME at HOST-NAME as USER-NAME identified by PASSWORD
```

create replicated partition

```
create [or replace] replicated partition DBS-NAME
 %AREA-SPEC%
 {to|from}
 DBS-NAME [at HOST-NAME][as USER-NAME identified by PASSWORD]
 [using USER-NAME identified by PASSWORD for creation]
 [%AREA-SPEC%]
 [
  mapped
  {globally|AREA-ALIAS}
  '('MEMBER-NAME [,...]')'
  to '('MEMBER-NAME [,...]')'
  [,...]
 ]
 [outline {!direct!|reverse}]
 [comment COMMENT-STRING]
 [remote comment COMMENT-STRING]
 [update {allow|disallow}]
 [validate only]
```

```
AREA-SPEC::=
 area MEMBER-EXPRESSION [AREA-ALIAS] [ ...]
```

create transparent partition

```
create [or replace] transparent partition DBS-NAME
 %AREA-SPEC%
 {to|from}
 DBS-NAME [at HOST-NAME][as USER-NAME identified by PASSWORD]
 [using USER-NAME identified by PASSWORD for creation]
 [%AREA-SPEC%]
 [
  mapped
  {globally|AREA-ALIAS}
  '('MEMBER-NAME [,...]')'
  to '('MEMBER-NAME [,...]')'
  [,...]
 ]
 [outline {!direct!|reverse}]
 [comment COMMENT-STRING]
 [remote comment COMMENT-STRING]
 [validate only]

AREA-SPEC::=
 area MEMBER-EXPRESSION [AREA-ALIAS] [ ...]
```

create after-update trigger

```
create [or replace] after update trigger TRIGGER-NAME
 where CUBE-AREA [when CONDITION then ACTION][ ...] end
```

create on-update trigger

```
create [or replace] [!on update!] trigger TRIGGER-NAME
 [log_value {!OFF!|ON}]
 where CUBE-AREA
 [when CONDITION then ACTION][ ...]
 [else ACTION]
 end
```

display application

```
display application [!all!|APP-NAME [message_level]]
```

display calculation

```
display calculation
 [
  !all!
  |CALC-NAME
  |on application APP-NAME
```

```
  |on database DBS-NAME
 ]
```

display database

```
display database
 [
  !all!
  |DBS-NAME
  |on application APP-NAME
 ]
 [request_history]
```

display disk volume

```
display disk volume
 [!all!|UNIQUE-VOL-NAME|on database DBS-NAME]
```

display drillthrough

```
display drillthrough
 {
  DBS-NAME [to FILE-NAME-PREFIX]
  |URL-NAME [to FILE-NAME]
 }
```

display filter

```
display filter [!all!|FILTER-NAME|on database DBS-NAME]
```

display filter row

```
display filter row [!all!|FILTER-NAME|on database DBS-NAME]
```

display group

```
display group [!all!|GROUP-NAME]
```

display location alias

```
display location alias [!all!|LOCATION-ALIAS-NAME|on application APP-
NAME|on database DBS-NAME]
```

display lock

```
display lock [!all!|on system|on application APP-NAME|on database DBS-
NAME]
```

display object

```
display [locked] object
 [
  [!all!|of type %OBJ-TYPE%]
  [!on system!|on application APP-NAME|on database DBS-NAME]
  |OBJ-NAME of type %OBJ-TYPE%
 ]

OBJ-TYPE::=
 outline
 |calc_script
 |report_file
 |rules_file
 |text
 |patition_file
 |lro
 |selection
 |wizard
 |eqd
 |outline_paging_file
 |worksheet
 |alias_table
```

display partition

```
display partition [!all!|on database DBS-NAME][advanced]
```

display privilege

```
display privilege
 {
  user [!all!|USER-NAME]
  |group [!all!|GROUP-NAME]
 }
```

display session

```
display session
 [
  !all!
  |SESSION-ID
  |by user USER-NAME [on application APP-NAME|on database DBS-NAME]
  |on application APP-NAME
  |on database DBS-NAME
 ]
```

display system

```
display system
 [
```

```
  version
  |ports {in use|overview}
  |export_directory
  |license_info
  |security mode
  |configuration
   {
   |agent
   |network
   |errors
   |on database DBS-NAME
   }
  |message_level
 ]
```

display trigger

```
display trigger
 [
  !all!
  |on system
  |on application APP-NAME
  |on database DBS-NAME
  |TRIGGER-NAME
 ]
```

display trigger spool

```
display trigger_spool
 [
  !all!
  |on application APP-NAME
  |on database DBS-NAME
  |SPOOL-NAME
 ]
```

display user

```
display user
 [
  in group [!all!|GROUP-NAME]
  |USER-NAME
 ]
```

display variable

```
display variable
 [
  !all!
  |VARIABLE-NAME
  |on application APP-NAME
```

```
  |on database DBS-NAME
  |on system
 ]
```

drop application

```
drop application APP-NAME [cascade] [force]
```

drop calculation

```
drop calculation CALC-NAME
```

drop database

```
drop database DBS-NAME [force]
```

drop drillthrough

```
drop drillthrough URL-NAME
```

drop filter

```
drop filter FILTER-NAME
```

drop location alias

```
drop location alias LOCATION-ALIAS-NAME
```

drop lock

```
drop lock
 [
  !all!
  |[
    !on system!
    |on application APP-NAME
    |on database DBS-NAME
   ]
  [!all!|held by USER-NAME]
 ]
```

drop object

```
drop object OBJ-NAME of type %OBJ-TYPE% [force]

OBJ-TYPE::=
 outline
 |calc_script
 |report_file
 |rules_file
```

```
 |text
 |patition_file
 |lro
 |selection
 |wizard
 |eqd
 |outline_paging_file
 |worksheet
 |alias_table
```

drop partition

```
drop
 {transparent|replicated}
 partition DBS-NAME {from|to} DBS-NAME
 [at HOST-NAME][force]
```

drop trigger

```
drop trigger TRIGGER-NAME
```

drop trigger spool

```
drop trigger_spool {SPOOL-NAME|all on database DBS-NAME}
```

execute aggregate build

```
execute aggregate build on database DBS-NAME
 using
  {
   views VIEW-ID VIEW-SIZE [,...] with outline_id OUTLINE-ID
   |view_file VIEW-FILE-NAME
  }
```

execute aggregate process

```
execute aggregate process on database DBS-NAME
 [stopping when total_size exceeds STOPPING-VAL]
 [based on query_data]
 [{enable|!disable!} alternate_rollups]
```

execute aggregate selection

```
execute aggregate selection on database DBS-NAME
 [
  using views VIEW-ID[,...]
  with outline_id OUTLINE-ID
  [[!suppress!|force] display]
 ]
 [selecting INTEGER views]
 [stopping when total_size exceeds STOPPING-VAL]
```

```
 [based on query_data]
 [{dump|force_dump} to view_file VIEW-FILE-NAME]
 [{enable|!disable!} alternate_rollups]
```

execute allocation (aggregate storage)

```
execute allocation process on database DBS-NAME with
 {
  pov MDX-SET
  amount ALLOC-NUMERIC
  {
   [amountcontext MDX-TUPLE]
   [amounttimespan MDX-SET ]
  }
  target MDX-TUPLE
  {
   [targettimespan MDX-SET]
   [targettimespanoptions {!divideamout!|repeatamount}]
   [offset MDX-TUPLE]
   [debitmember MDX-MBR]
   [creditmember MDX-MBR]
  }
  range MDX-SET
  {
   [excludedrange MDX-SET]
   [basis MDX-TUPLE]
   [basistimespan MDX-SET]
   [basistimespanoptions {splitbasis|combinebasis}]
   [
    share
    |spread [{skip_missing|skip_zero|skip_negative},...]
   ]
  [zeroamountoptions {skip_to_next_amount|abort}]
  [zerobasisoptions
    {
     skip_to_next_amount
     |abort
    }
  ]
  [negativebasisoptions
    {
     skip_to_next_amount
     |abort
     |absolute_value
     |missing_value
     |zero_value
    }
  ]
  [round
    {INTEGER|MDX-NUMERIC}
    {
     discard errors
     |errors_to_lowest
     |errors_to_highest
```

```
     |errors_to_location MDX-TUPLE
    }
  ]
  [{!override!|add|subtract} values]
 }
```

execute calculation

```
execute calculation
 {
  CALC-NAME
  |CALC-NAME on database DBS-STRING
  |{CALC-STRING|default} on DBS-NAME
 }
```

execute calculation (aggregate storage)

```
execute calculation on database DBS-NAME with
local script_file FILE-NAME pov MDX-SET sourceregion MDX-SET
  {
   [target MDX-TUPLE]
   |[debitmember MDX-MBR]
   |[creditmember MDX-MBR]
   |[offset MDX-TUPLE]
  }
 [{!override!|add|subtract} values]
```

export data

```
export database DBS-NAME
 {
  [!all!|level0|input]
   data [anonymous] [in columns] to [!server!] data_file FILE-NAME[,...]
  |using [!local!|server] report_file FILE-NAME to data_file FILE-NAME
 }
```

export data (aggregate storage)

```
export database DBS-NAME
 {
  [!level0!|input]
   data [anonymous] to [!server!] data_file FILE-NAME[,...]
  |using [!local!|server] report_file FILE-NAME to data_file FILE-NAME
```

export lro

```
export databse DBS-NAME lro to
 [!server!|local] directory
{DBS-EXPORT-DIR|FULL-EXPORT-DIR}
```

export outline

```
export outline {DBS-NAME|FILE-NAME}
 {
  all dimensions
  |list dimensions '{'DIM-NAME'}'[,...]
 }
 [tree|with alias_table ALT-NAME-SINGLE]
 to xml_file FILE-NAME
```

export query_tracking

```
export query_tracking DBS-NAME to
  [!server!] file FILE-NAME
```

grant

```
grant
 {
  {create_application|create_user|no_access|administrator}
   [on system]
   |{no_access|manager} on application APP-NAME
   |{no_access||read|write|manager} on database DBS-NAME
   |filter FILTER-NAME
   |execute
    {
     CALC-NAME
     |{any|default}
       [
        !on system!
        |on application APP-NAME
        |on database DBS-NAME
       ]
  }
  to {USER-NAME|GROUP-NAME}
```

import data

```
import database DBS-NAME
 [using max_threads INTEGER]
 data
 {
  from
   [!local!|server]
   [!text!]
   data_file IMP-FILE
   [using [!local!|server] rules_file IMP-FILE]
  |from data_string STRING
  |connect as SQL-USR identified by SQL-PASS
   using [!local!|server] rules_file IMP-FILE
```

```
 }
 on error {{write|append}to FILE-NAME|abort}
```

import data (aggregate storage)

```
import database DBS-NAME data
 {
  from
   [!local!|server]
   [!text!]
   data_file IMP-FILE
   [using [!local!|server] rules_file IMP-FILE]
  |from data_string STRING
  |connect as SQL-USR identified by SQL-PASS
   using
   {
    [!local!|server] rules_file IMP-FILE
    |multiple rules_file RULE-FILE-NAME[,...]
     to load_buffer_block starting with buffer id BUFFER-ID
     on error {write to FILE-NAME|abort}
   }
  |from load_buffer with buffer_id BUFFER-ID[,...]
   [{!override!|add|subtract} values]
   [create slice]
   |override {all|incremental} data

   ]
 }
 on error {{write|append}to FILE-NAME|abort}
```

import dimensions

```
import database DBS-NAME dimensions
 {
  from
   [!local!|server]
   [!text!] data_file IMP-FILE
   using[!local!|server] rules_file IMP-FILE
   [[!enforce!|suppress] verification]
   |connect as SQL-USR identified by SQL-PASS
   using[!local!|server] rules_file IMP-FILE
 }[,...]
 [
  !preserve all data!
  |preserve {level0|input} data
 ]
 on error {write|append} to FILE-NAME
```

import lro

```
import database DBS-NAME
 lro from [!local!|server] directory IMPORT-DIR
```

import query_tracking

```
import query_tracking DBS-NAME from
  [!server!] file FILE-NAME
```

query application

```
query application APP-NAME get cache_size
```

query application (aggregate storage)

```
query application APP-NAME
 {
  get cache_size
  |list aggregate_storage storage_info
 }
```

query archive file

```
query archive_file FILE-NAME {get overview|list disk volume}
```

query database

```
query database DBS-NAME
 {
  get
   {
    active_alias_table
    |attribute_info MEMBER-NAME
    |attribute_spec
    |currency_rate
    |dbstats {dimension|data_block}
    |default calculation
    |member_info MEMBER-NAME
    |member_calculation MEMBER-NAME
    |estimated size
    |performance statistics
     {
      kernel_io
      |kernel_cache
      |end_transaction
      |database_synch
      |database_asynch
      |dynamic_calc
     }
     table
   }
    |list
     {
      alias_table
      |alias_names in alias_table ALT-NAME-SINGLE
```

```
      |lro
       [
        !all!
        |by USER-NAME
        |before DATE
        |by USER-NAME before DATE
       ]
      |{all|data|index} file information
      |transactions
       [
        after LOG-TIME
        [[force] write to file PATHNAME_FILENAME]
       ]
     }
 }
```

query database (aggregate storage)

```
query database DBS-NAME
 {
  get
   {
    active_alias_table
    |attribute_info MEMBER-NAME
    |attribute_spec
    |cube_size_info
    |dbstats {dimension|data_block}
    |member_info MEMBER-NAME
    |opg_state of %OPG-SECTION% for dimension DIM-NAME
   }
   |list
    {
     |aggregate_storage runtime_info
     |aggregate_storage compression_info
     |aggregate_storage group_id_info
     |aggregate_storage slice_info
     |aggregate_storage uncommitted_transaction_info
     |alias_table
     |alias_names in alias_table ALT-NAME-SINGLE
     |existing_views [based on query_data
     |{!all!|data|index} file information
     |load_buffers
     |aso_level_info
    }
   |{dump|force_dump}
     existing_views to view_file VIEW-FILE-NAME
     [based on query_data]
 }
```

refresh outline

```
refresh outline on {transparent|replicated}
 partition DBS-NAME {to|from} DBS-NAME
 [at HOST-NAME]
```

```
 {
  purge outline change_file
  |apply all
  |apply nothing
  |%OTL-CHANGE-SPEC%
 }

OTL-CHANGE-SPEC::=
 apply on dimension
  {add|delete|rename|update|move}[...,]
 apply on member
  {add|delete|rename|move}[...,]
 apply on member_property  {
   account_type
   |alias
   |calc_formula
   |consolidation
   |currency_conversion
   |currency_category
   |data_storage
   |uda
  }[...,]
```

refresh replicated partition

```
refresh replicated partition DBS-NAME
 {to|from} DBS-NAME
 [at HOST-NAME]
 [[!all!|updated]data]
```

MaxL Statements (Aggregate Storage)

Click here for non-aggregate storage list

Some MaxL grammar is applicable only to aggregate storage mode, and some standard grammar is not applicable to aggregate storage mode. The following statements support aggregate storage application and database operations.

- alter application
- alter database
- alter filter
- alter object
- alter partition
- alter system
- alter tablespace
- alter trigger
- create application
- create database
- create filter

- create outline
- create partition
- create after-update trigger
- display application
- display calculation
- display database
- display filter
- display filter row
- display group
- display lock
- display object
- display partition
- display privilege
- display session
- display system
- display tablespace
- display trigger
- display user
- display variable
- drop application
- drop calculation
- drop database
- drop filter
- drop lock
- drop object
- drop partition
- drop trigger
- execute aggregate process
- execute aggregate build
- execute aggregate selection
- export data
- grant
- import data
- import dimensions
- login
- query application
- query database

- refresh outline
- refresh replicated partition

The MaxL grammar is case-insensitive. Semicolon statement-terminators are required when using the MaxL Shell. Key words of the MaxL grammar are represented in this document in lower-case. Terminals, represented in upper-case, are to be replaced by the appropriate names, numbers, privileges, or strings. For more information about components of MaxL statements, see MaxL Definitions.

> **Note:**
>
> MaxL Shell Invocation is part of the separate command shell grammar, not the MaxL language itself.

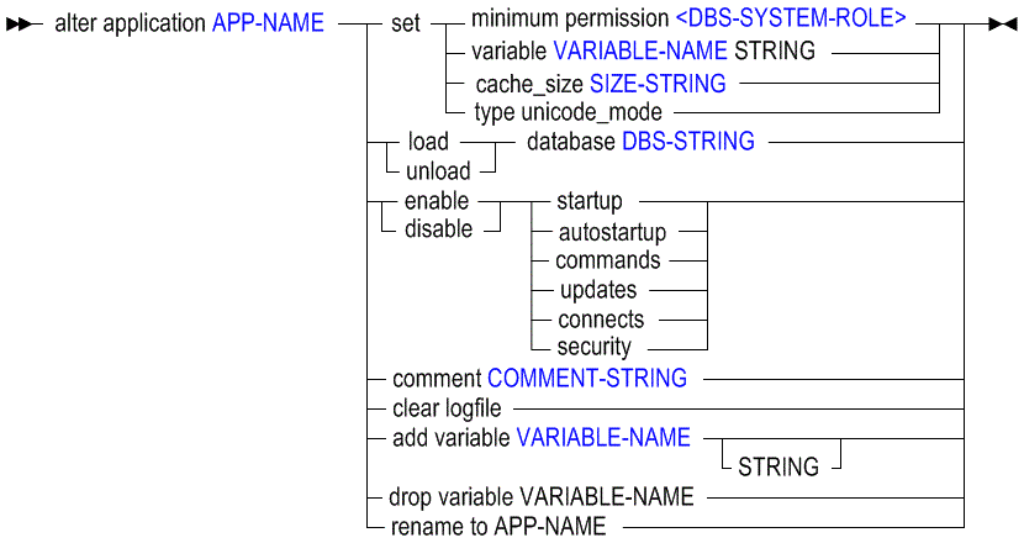
Alter Application (Aggregate Storage)

Click here for non-aggregate storage version

Change application-wide settings.

Permission required: Application Manager.

Syntax



- APP-NAME
- Database-Level System Roles
- VARIABLE-NAME
- SIZE-STRING
- DBS-STRING
- COMMENT-STRING

- VARIABLE-NAME

You can change the following application-wide settings using **alter application**.

Keywords

set minimum permission
Grant all users a minimum level of permission to all databases in the application. Users with higher permissions than this minimum are not affected.

set variable
Assign a string value to an existing substitution-variable name. If the variable does not exist, first create it using **add variable**. Substitution variables may be referenced by calculations in the application.

set cache_size
Set the maximum size to which the aggregate storage cache may grow. The aggregate storage cache grows dynamically until it reaches this limit. This setting takes effect after you restart the application. To check the currently set limit, use the following MaxL statement:

```
query application APP-NAME get cache_size;
```

set type unicode_mode
Migrate an application to Unicode mode. Migration to Unicode mode cannot be reversed.

load database
Start (by loading into memory) an idle database. The statement will fail if you do not have at least read privilege for the database.

unload database
Stop (by unloading from memory) an active database. The statement will fail if you do not have at least read privilege for the database.

enable startup
Permit all users to load (start) the application. This only applies to users who have at least read privilege for the application. Startup is enabled by default.

disable startup
Prevent all users from loading (starting) the application. Startup is enabled by default.

enable autostartup
Start the application automatically when Essbase Server starts. By default, autostartup is disabled.

disable autostartup
Do not start the application automatically when Essbase Server starts. By default, autostartup is disabled.

enable commands
Allow all users with sufficient permissions to make requests to databases in the application. Use to reverse the effect of **disable commands**. The disable commands setting remains in effect only for the duration of your session. By default, commands are enabled.

disable commands
Prevent all requests to databases in the application, including non-data-specific requests, such as viewing database information or changing database settings. All users are affected, including other administrators. Administrators are affected by this setting as a safety mechanism to prevent accidental updates to databases during maintenance operations. This setting remains in effect only for the duration of your session. The setting takes effect immediately, and affects users who are currently logged in, as well as users who log in later during your session.

> **Caution:**
>
> If performing maintenance operations that require disabling commands, you must make those maintenance operations within the same session and the same script as the one in which commands were disabled.

By default, commands are enabled.

enable updates
Allow all users with sufficient permissions to make requests to databases in the application. Use to reverse the effect of **disable updates**. Disabling updates remains in effect only for the duration of your session. By default, updates are enabled.

disable updates
Prevent all users from making requests to databases in the application. Use before performing update and maintenance operations. The disable updates setting remains in effect only for the duration of your session.

> **Caution:**
>
> If performing maintenance operations that require updates to be disabled, you must make those maintenance operations within the same session and the same script as the one in which updates were disabled. By default, updates are enabled.

enable connects
Allow all users with sufficient permissions to make connections to databases in the application. Use to reverse the effect of **disable connects**. By default, connections are enabled.

disable connects
Prevent any user with a permission lower than Application Manager from making connections to the databases that require the databases to be started. Database connections remain disabled for all databases in the application, until the application setting is re-enabled by the administrator.
By default, connections are enabled.

enable security
When security is disabled, Essbase ignores all security settings in the application and treats all users as Application Managers. By default, security is enabled.

disable security
When security is disabled, Essbase ignores all security settings in the application and treats all users as Application Managers. By default, security is enabled.

comment
Enter an application description (optional). The description can contain up to 80 characters.

clear logfile
Delete the application log located in the application directory. A new log is created for entries recording subsequent application activity.

add variable
Create an application-level substitution variable by name, and optionally assign a string value for the variable to represent. You can assign or change the value later using **set variable**. A substitution variable acts as a global placeholder for information that changes regularly. Substitution variables may be referenced by calculations and report scripts.
If substitution variables with the same name exist at server, application, and database levels, the order of precedence for the variables is as follows: a database level substitution variable supersedes an application level variable, which supersedes a server level variable.

drop variable
Remove a substitution variable and its corresponding value from the application.

rename to
Rename the application. When you rename an application, the application and the application directory (*ARBORPATH*\app*appname*) are renamed.

Example

```
alter application ASOsamp set cache_size 64MB;
```

Sets the maximum size of the aggregate storage cache to 64 MB.

```
alter application ASOsamp disable commands;
```

Prevents all users from making requests to the application scope. Use this statement before performing application-wide update and maintenance operations.

```
alter application ASOsamp comment 'Aggregate storage application';
```

Attaches a descriptive comment to the ASOsamp application.

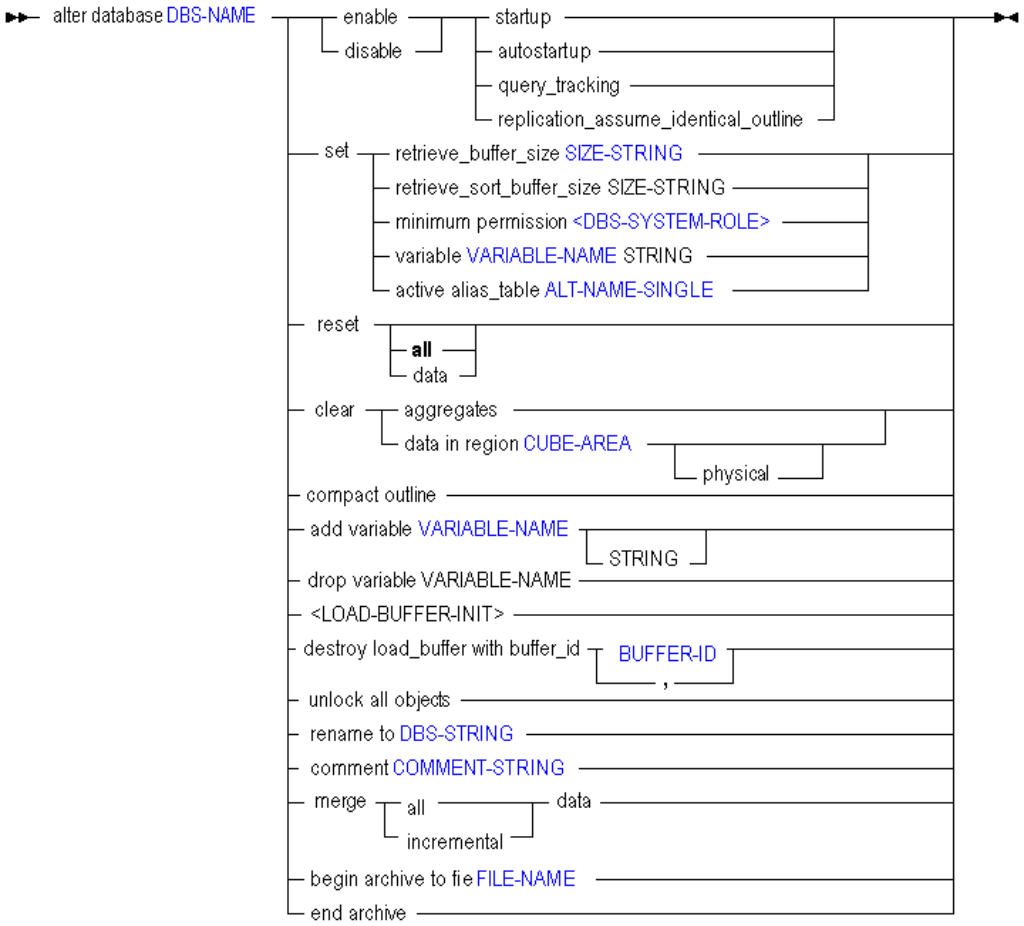
Alter Database (Aggregate Storage)

Click here for non-aggregate storage version

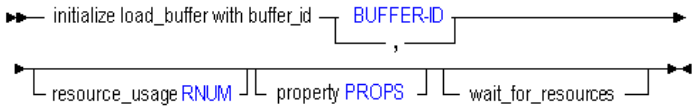
Change database-wide settings.

Permission required: create_application.

Syntax



<LOAD-BUFFER-INIT> ::=



- DBS-NAME
- DBS-SYSTEM-ROLE
- SIZE-STRING
- VARIABLE-NAME
- ALT-NAME-SINGLE
- CUBE-AREA
- BUFFER-ID
- RNUM
- PROPS
- DBS-STRING

- COMMENT-STRING
- FILE-NAME

You can change the following database-wide settings using **alter database**.

Keywords

enable startup
Enable users to start the database directly or as a result of requests requiring the database to be started. Startup is enabled by default.

disable startup
Prevent all users from starting the database directly or as a result of requests that would start the database. Startup is enabled by default.

enable autostartup
Automatically start the database when the application to which it belongs starts. Autostartup is enabled by default. This setting is applicable only when startup is enabled.

disable autostartup
Prevent automatic starting of the database when the application to which it belongs starts. Autostartup is enabled by default.

enable query_tracking
Begin collecting query data for this database, to be used for query-based view optimization.
To utilize the results of query tracking, use the optional **based on query_data** grammar in any of the following statements:

- query database <dbs-name> list existing_views
- execute aggregate process
- execute aggregate selection

Query tracking is disabled by default.

disable query_tracking
Stop collecting query data for query-based view optimization. Query tracking is disabled by default.

set retrieve_buffer_size
Change the database retrieval buffer size. This buffer holds extracted row data cells before they are evaluated by the RESTRICT or TOP/BOTTOM Report Writer commands. The default size is 10 KB. The minimum size is 2 KB. Increasing the size may improve retrieval performance.

set retrieve_sort_buffer_size
Change the database retrieval sort buffer size. This buffer holds data until it is sorted. The default size is 10 KB. The minimum size is 2 KB. Increasing the size may improve retrieval performance.

set minimum permission
Set a level of permission that all users or groups can have to the database. Users or groups with higher granted permissions than the minimum permission are not affected.

set variable
Change the value of an existing subsitution variable on the database. The value must not exceed 256 bytes. It may contain any character except a leading ampersand (&).

set active alias_table
Set an alias table as the primary table for reporting and any additional alias requests. Only one alias table can be used at a time. This setting is user-specific; it only sets the active alias table for the user issuing the statement.

reset
Clear all data and linked-reporting objects from the database, but preserve the outline.

> **Note:**
>
> If kernel queries are running when a clear data operation starts, the clear data operation waits for the kernel queries to complete and then the clear data operation proceeds. This information also applies to the **reset all** and **reset data** grammar.

reset all
Clear all data, Linked Reporting Objects, and the outline.

reset data
Same as using `reset`.

clear aggregates
Delete all aggregate views.

compact outline
Compact the outline file to decrease the outline file size. Compaction helps keeps the outline file at an optimal size. After the outline file is compacted, the file continues to grow as before, when members are added or deleted.

> **Note:**
>
> Compacting the outline does not cause Essbase to clear the data. When a member is deleted from the outline, the corresponding record of that member in the outline file is marked as deleted but the record remains in the outline file. Compacting the outline file does not remove the records of deleted members.

add variable
Create a database-level substitution variable by name, and optionally assign a string value for the variable to represent. You can assign or change the value later using `set variable`. A substitution variable acts as a global placeholder for information that changes regularly. Substitution variables may be referenced by calculations and report scripts.
If substitution variables with the same name exist at server, application, and database levels, the order of precedence for the variables is as follows: a database level substitution variable supersedes an application level variable, which supersedes a server level variable.

drop variable
Remove a substitution variable and its corresponding value from the database.

initialize load_buffer
Create a temporary buffer in memory for loading data.
Data load buffers are used in aggregate storage databases for allocations, custom calculations, and lock and send operations. Multiple data load buffers can exist on a single aggregate storage database.
You can control the share of aggregate storage cache resources the load buffer is allowed to use and how long to wait for resources to become available before aborting load buffer operations. You can also set properties that determine how missing and zero values, duplicate values, and multiple values for the same cell in the data source are processed.

- resource_usage
- property

destroy load_buffer
Destroy the temporary data-load memory buffer.

unlock all objects
Unlock all objects on the database that are in use by a user or process.

rename to
Rename the database. When you rename a database, the database directory is also renamed.

comment
Create a description of the database. The maximum number of characters is 80. This description is available to database administrators. To annotate the database for Smart View or other grid client users, use `set note`.

merge all|incremental data [remove_zero_cells]
Merge incremental data slices. Use these keywords:

- all—Merge all incremental data slices into the main database slice.
- incremental—Merge all incremental data slices into a single data slice. The main database slice is not changed.
- (Optional) remove_zero_cells—When merging incremental data slices, remove cells that have a value of zero (logically clearing data from a region results in cell with a value of zero).

clear data in region …
Clear the data in the specified region.
There are two methods for clearing data from a region:

- Physical, in which the input cells in the specified region are physically removed from the aggregate storage database. The process for physically clearing data completes in a length of time that is proportional to the size of the input data, not the size of the data being cleared. Therefore, you might typically use this method only when you need to remove large slices of data.

Use the MaxL statement with the physical keyword:

```
alter database appname.dbname clear data in region 'MDX set
expression' physical;
```

- Logical, in which the input cells in the specified region are written to a new data slice with negative, compensating values that result in a value of zero for the cells you want to clear. The process for logically clearing data completes in a length of time that is proportional to the size of the data being cleared. Because compensating cells are created, this option increases the size of the database.

  Use the MaxL statement without a keyword:

  ```
  alter database appname.dbname clear data in region 'MDX set
  expression';
  ```

The region must be symmetrical. Members in any dimension in the region must be stored members. When physically clearing data, members in the region can be upper-level members in alternate hierarchies. (If the region contains upper-level members from alternate hierarchies, you may experience a decrease in performance.) Members cannot be dynamic members (members with implicit or explicit MDX formulas), nor can they be from an attribute dimension.
To remove cells with a value of zero, use the **alter database** MaxL statement with the **merge** grammar and the **remove_zero_cells** keyword.

enable replication_assume_identical_outline
Optimize the replication of an aggregate storage database when the aggregate storage database is the target and a block storage database is the source and the two outlines are identical.
Replication optimization affects only the target aggregate storage application; the source block storage application is not affected. This functionality does not apply to block storage replication.

disable replication_assume_identical_outline
Do not optimize the replication of an aggregate storage database when the aggregate storage database is the target and a block storage database is the source and the two outlines are identical.

begin archive to file
Prepare the database for backup by an archiving program, and prevent writing to the files during backup.
Begin archive achieves the following outcomes:

- Switches the database to read-only mode. The read-only state persists, even after the application is restarted, until it is changed back to read-write using `end archive`.
- Creates a file containing a list of files that need to be backed up. Unless a different path is specified, the file is stored in the database directory.

Begin archive and end archive do not perform the backup; they simply protect the database during the backup process.

> **Note:**
>
> Using the **begin archive to file** and **end archive** grammar is the only supported way to backup and recover a database using MaxL.

end archive
Return the database to read-write mode after backing up the database files.

> **Note:**
>
> Using the **begin archive to file** and **end archive** grammar is the only supported way to backup and recover a database using MaxL.

Example

```
alter database ASOsamp.Sample clear aggregates;
```

Deletes all aggregate views in the ASOsamp.Sample database.

```
alter database ASOsamp.Sample initialize load_buffer with buffer_id 1;
```

See Loading Data Using Buffers.

```
alter database ASOsamp.Sample initialize load_buffer with buffer_id 1
resource_usage .5 property ignore_missing_values, ignore_zero_values;
```

Creates a data-load buffer in memory for the ASOsamp.Sample database. The buffer can use only 50% of available resources. Missing values and zeros in the data source are ignored.

```
alter database ASOsamp.Sample disable query_tracking;
```

Turns off the harvesting of query data for the ASOsamp.Sample database.

```
alter database ASOsamp.Sample merge all data;
```

Merges all incremental data slices into the main slice in the ASOsamp.Sample database.

```
alter database ASOsamp.Sample merge incremental data;
```

Merges all incremental data slices into a single data slice within the ASOsamp.Sample database.

```
alter database ASOsamp.Sample merge all data remove_zero_cells;
```

Merges all incremental data slices into the main slice in the ASOsamp.Sample database, and removes cells with a value of zero.

```
alter database ASOsamp.Sample clear data in region '{Jan, Budget}';
```

Clears all Budget data for the month of Jan, using the logical method, from the ASOsamp.Sample database.

```
alter database ASOsamp.Sample clear data in region '{Jan, Budget}'
physical;
```

Clears all Budget data for the month of Jan, using the physical method, from the ASOsamp.Sample database.

```
alter database ASOsamp.Sample clear data in region 'CrossJoin({Jan},
{Forecast1, Forecast2})';
```

Clears all January data for the Forecast1 and Forecast2 scenarios from the ASOsamp.Sample database.

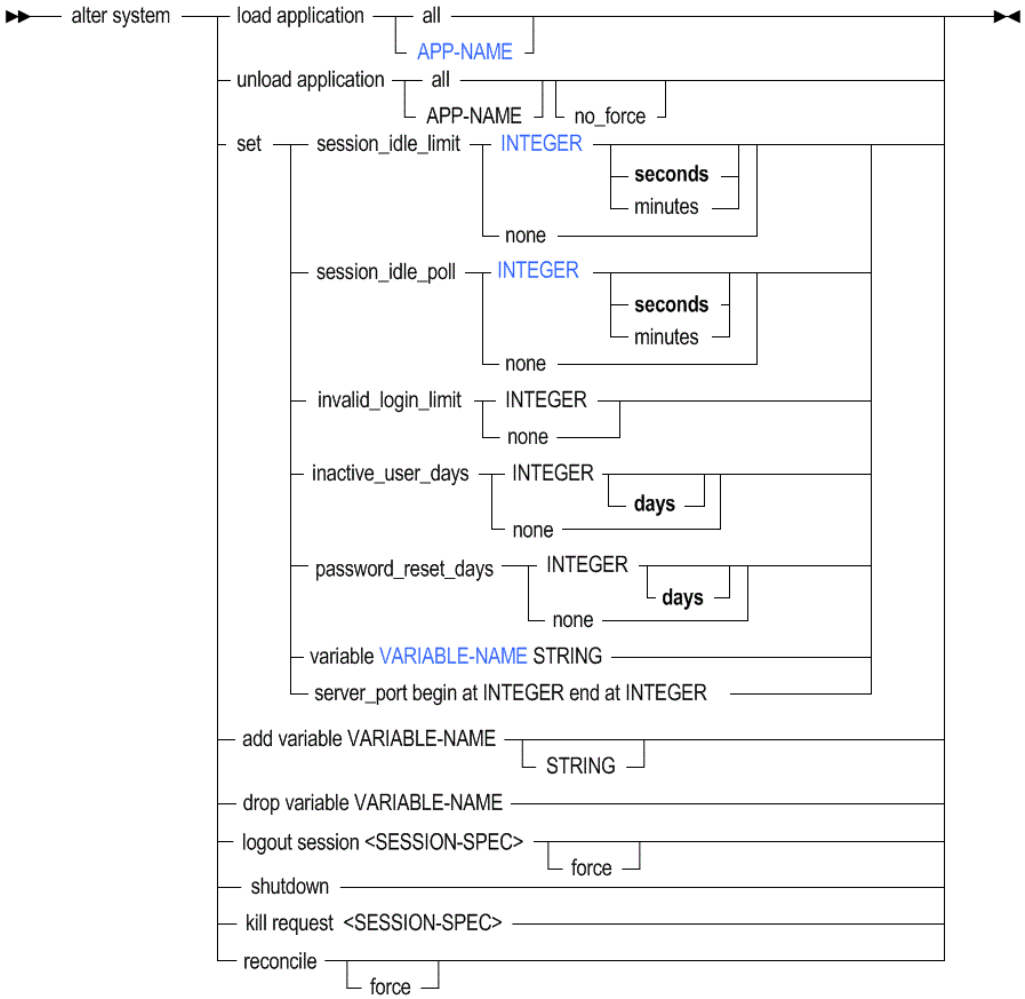
Alter System (Aggregate Storage)

Click here for non-aggregate storage version

Change the state of the Essbase Server. Start and stop applications, manipulate system-wide variables, manage password and login activity, disconnect users, kill processes, and shut down the server.

Permission required: Administrator.

Syntax



- APP-NAME
- INTEGER
- VARIABLE-NAME

You can change the following system-wide settings using **alter system**.

Keywords

load application
Start an application, or start all applications on the Essbase Server.

unload application
Stop an application, or stop all applications on the Essbase Server. Unloading an application cancels all active requests and database connections, and stops the application. If Essbase encounters a problem when trying to cancel active requests and database connections, and stopping the application, an error is logged in the application log.
If you do not want to stop an application if it has active requests and database connections, use the **no_force** grammar. When using **no_force**:

- If the application has active requests and database connections, the application is not stopped; it continues running
- If the application does not have active requests and database connections, the application is stopped, as if you used **unload application** without specifying **no_force**

> **Note:**
>
> Unloading an application cancels all active requests and database connections, and stops the application, unless you explicitly specify otherwise using the *no_force* option. The *no_force* option causes Essbase to return an error if active requests are running on the application.
> An internal logic error [200] is logged when a database is unable to shut down gracefully when unloading an application or shutting down the system while a process is running on the database.

set session_idle_limit
Set the interval of time permitted for a session to be inactive before Essbase Server logs off the user. The minimum limit that you can set is five minutes (or 300 seconds). When the session idle limit is set to **none**, all users can stay logged on until the Essbase Server is shut down.
The default user idle logout time is 60 minutes. When a user initiates a calculation in the background, after 60 minutes the user is considered idle and is logged out, but the calculation continues in the background.
Because the user may mistakenly assume that the calculation stopped because he or she was logged out, you can do one of the following to correct the user experience:

- Run the calculation in the foreground
- Increase the session idle limit in to a time that exceeds the duration of the calculation, or to none

set session_idle_poll
Set the time interval for inactivity checking and security-backup refreshing. The time interval specified in the session idle poll gives Essbase instructions:

- Tells it how often to check whether user sessions have passed the allowed inactivity interval indicated by `session_idle_limit` in the **alter system** statement.
- Tells it how often to refresh the security backup file. If `session_idle_poll` is set to zero, the security backup file is still refreshed every five minutes.

set invalid_login_limit
Set the number of unsuccessful login attempts allowed by any user before the user account becomes disabled. When you change this setting, the counter resets to 0. When the invalid login limit is set to **none**, there is no limit. By default, there is no limit.

set inactive_user_days
Set the number of days a user account may remain inactive before the system disables it. The counter resets when the user logs in, is edited, or is activated by an administrator. When the inactive days limit is set to **none**, user accounts remain enabled even if they are not used. By default, there is no limit.

set password_reset_days
Set the number of days users may retain passwords. After the allotted number of days, users are prompted at login to change their passwords. The counter resets for a user when the user changes the password, is edited, or is activated by an administrator. When the password reset days limit is set to **none**, there is no built-in limit for password retention. By default, there is no limit.

set variable
Change the value of an existing subsitution variable on the system. The value must not exceed 256 bytes. It may contain any character except a leading ampersand (&).

set server_port
Expand a port range specified in Essbase configuration properties. Each Essbase application uses two ports from this range. If no more ports are available, an error message is displayed.

> **Note:**
>
> You can expand port ranges only so that the beginning port range is less than SERVERPORTBEGIN and the ending port range is greater than SERVERPORTEND.

add variable
Create a system-level substitution variable by name, and optionally assign a string value for the variable to represent. You can assign or change the value later using **set variable**. A substitution variable acts as a global placeholder for information that changes regularly. Substitution variables may be referenced by calculations and report scripts.
If substitution variables with the same name exist at server, application, and database levels, the order of precedence for the variables is as follows: a database-level substitution variable supersedes an application-level variable, which supersedes a server-level variable.

drop variable
Remove a substitution variable and its corresponding value from the system.

logout session all
Terminate all user sessions currently running on the Essbase Server.

logout session...force
Terminate a session (or sessions) even if it is currently processing a request. The request is allowed to proceed to a safe point, and then the transaction is rolled back.

logout session <session-id>
Terminate a session by its unique session ID number. To see the session ID number, use display session.

logout session by user
Terminate all current sessions by a particular user, either across the entire Essbase Server, or limited to a specific application or database.

logout session by user on application
Terminate all current sessions by a particular user across a specific application.

logout session by user on database
Terminate all current sessions by a particular user across a specific database.

logout session on application
Terminate all current user sessions across a specific application.

logout session on database
Terminate all current user sessions across a specific database.

shutdown
Shut down the Essbase Server.

kill request all
Terminate all current requests on the Essbase Server.

> **Note:**
>
> To terminate your own active request in MaxL Shell, press the ESC key.

kill request <session-id>
Terminate the current request indicated by the session ID. You can obtain session IDs using display session.

kill request by user
Terminate all current requests by the specified user on the Essbase Server.

kill request on application
Terminate all current requests on the specified application.

kill request on database
Terminate all current requests on the specified database.

reconcile
When Essbase is started using a security backup file (`essbase_`*`timestamp`*`.bak`) instead of `essbase.sec`, reconcile the security file to match the state of Essbase on an external disk. This grammar displays discrepancies in application and database information between the security file and the external disk:

- If an application folder is on the disk but not in the security file, display a message indicating the discrepancy. (Essbase checks for the presence of a *`appname`*`.app` file in the *`ARBORPATH`*`/app/`*`appname`* directory.)

  The *`force`* option does not apply in this scenario.
- If an application file is in the security file but not on the disk, display a message indicating the discrepancy.

  The *`force`* option removes the application from the security file.
- If an application database folder is on the disk but not in the security file, display a message indicating the discrepancy. (Essbase checks for the presence of a *`dbname`*`.otl` file in the *`ARBORPATH`*`/app/`*`appname`*`/`*`dbname`* directory.)

  The *`force`* option does not apply in this scenario.

- If an application database file is in the security file but not on the disk, display a message indicating the discrepancy.

  The *force* option removes the database from the security file.

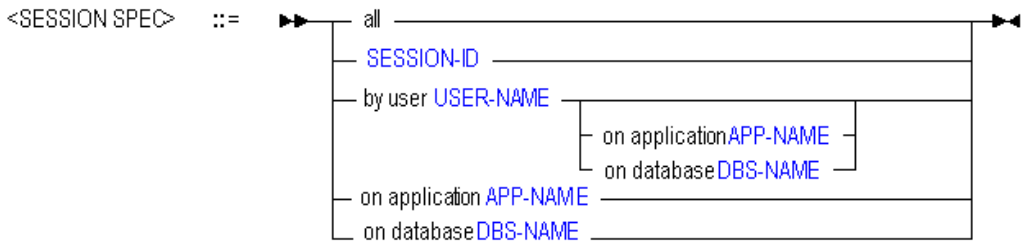
Notes

SESSION SPECIFICATION

A *session* is a single user connection to Essbase Server. The session can be identified by keywords and names indicating context, or by a unique session ID number.

A *request* is a query sent to Essbase Server by a user or by another process; for example, starting an application or restructuring a database outline. Only one request at a time can be processed in each session.

If a session is processing a request at the time that an administrator attempts to terminate the session, the administrator must either terminate the request first, or use the **force** keyword available with **alter system** to terminate the session *and* the current request.



Example

```
alter system unload application Sample;
```

Stops the Sample application, if it is currently running.

```
alter system unload application all;
```

Terminates all active requests and stops all applications.

```
alter system unload application Sample no_force;
```

Essbase prepares to unload the Sample application; however, if active requests are running, the application is not stopped.

```
alter system shutdown;
```

Stops all running applications and shuts down Essbase Server.

```
alter system logout session by user Fiona;
```

Disconnects Fiona from any applications or databases to which she is connected.

To log out a user, log out the sessions owned by that user.

```
alter system set password_reset_days 10;
```

Specifies that all users will be prompted after 10 days to change their passwords. The day count for any user is reset when the user changes the password or is edited or reactivated by an administrator.

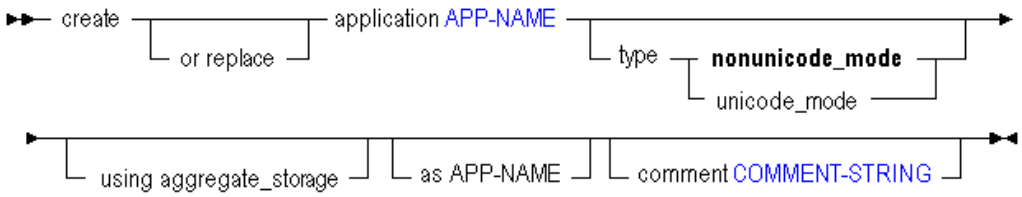
```
alter system unload application Sample;
```

Create Application (Aggregate Storage)

Click here for non-aggregate storage version

Create or re-create an application, either from scratch or as a copy of another application on the same system. See APP-NAME for information on the maximum length of and special characters that are allowed in an application name. Application names are not case-sensitive.

Syntax



- APP-NAME
- COMMENT-STRING

You can create an application in the following ways using the aggregate storage version of **create application**.

Keywords

create application
Create a new application. Application names are not case-sensitive.

create or replace application
Create an application, or replace an existing application of the same name. Application names are not case-sensitive.

...type nonunicode_mode
Create a Non Unicode-mode application. This is also the default if these keywords are omitted.

...type unicode_mode
Create a Unicode-mode application.

...using aggregate_storage
Create an application using an aggregate storage model. Only one database per application is allowed. Selecting to use aggregate storage model for an application is non-reversible.
Use the aggregate storage model if the following is true for your database:

- The database is sparse and has many dimensions, or a large hierarchical depth of members in the dimensions.
- The database is used primarily for read-only purposes; there are few or no data updates.
- There are no formulas on the outline except in the dimension tagged as Accounts.
- Calculation of the database is frequent and highly aggregational, with no dependency on calculation scripts.

create application as
Create an application as a copy of another application. Application names are not case-sensitive.
You cannot copy block storage applications to aggregate storage applications or vice versa. The copy will always use the same storage as the original. However, you can convert an outline from a block storage database to an aggregate storage database, using create outline.
Before you copy an aggregate storage application, you must merge all incremental data slices into the main database slice. Data in unmerged incremental data slices is not copied.

comment
Create an application description (optional). The description can contain up to 80 characters.

Example

```
create application Sample2 using aggregate_storage comment 'aggregate
storage application.';
```

Creates a new aggregate storage application called Sample2, with an associated comment.

Create Database (Aggregate Storage)

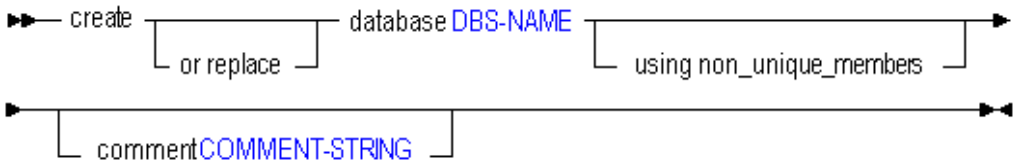
Click here for non-aggregate storage version

Create or re-create a database for an aggregate storage application. See DBS-NAME for information on the maximum length of and special characters that are allowed in a database name. Database names are not case-sensitive.

The syntax for creating an aggregate storage database is the same as for creating a block storage database. You must create an aggregate storage database as part of an aggregate storage application.

Permission required: Application Manager.

Syntax



- DBS-NAME
- COMMENT-STRING

Use **create database** to create a database in the following ways:

Keywords

create database
Create a new database. Database names are not case-sensitive.

create or replace database
Create a database, or replace an existing database of the same name. Database names are not case-sensitive.

create database using non_unique_members
Create a database that supports the use of duplicate member names. Once you have created a database with a duplicate member outline, you cannot convert it back to a unique member outline.
For more information about duplicate member names, see Creating and Working With Duplicate Member Outlines.

comment
Create a database description (optional). The description can contain up to 80 characters.

Notes

- You cannot create an aggregate storage database as a copy of another aggregate storage database. Only one aggregate storage database is allowed per application.
- You cannot copy a block storage database to an aggregate storage database. For an example of how to create an aggregate storage application and database based on a block storage application and database, see Creating an Aggregate Storage Sample Using MaxL.

Example

```
create or replace database Sample.Basic comment 'This is a test.';
```

Creates a database called Basic within the Sample application. If a database named Basic within the Sample application already exists, it is overwritten.

Create Outline (Aggregate Storage)

Create an aggregate storage outline based on a block storage outline. The outline you are creating must be for an aggregate storage cube that is local to your current login session. The block-storage cube you are using as a source can be remote. If a remote host is specified, you can also specify a user name and password if the connection is remote.

Permission required: Database Manager.

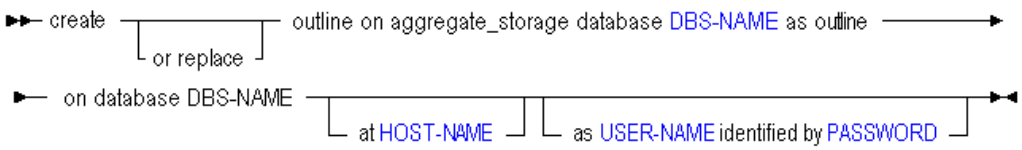
Essbase supports the following scenarios for converting block storage outlines to aggregate storage outlines:

- Non-Unicode block storage outline to non-Unicode aggregate storage outline
- Non-Unicode block storage outline to Unicode aggregate storage outline
- Unicode block storage outline to Unicode aggregate storage outline

The following conversion scenarios are not supported:

- Unicode block storage outline to non-Unicode aggregate storage outline
- Aggregate storage outline to a block storage outline

Syntax



- DBS-NAME
- HOST-NAME
- USER-NAME
- PASSWORD

You can create an outline in the following ways using **create outline**.

Keywords

create outline...
Create an aggregate-storage cube outline based on a block storage outline. If an outline of the same name already exists, it is replaced.

create or replace outline...
This statement has the same result as `create outline` above.

at HOST-NAME
If the block-storage cube you are using as a source is remote, specify its discovery URL. For example, `"https://myEssbase-myDomain.analytics.us2.example.com/essbase/agent"`

as USER-NAME identified by PASSWORD
If the block-storage cube you are using as a source is remote, specify the location. If the connection is also remote (requires a different authentication), provide the user name and password, as you would do when creating a remote partition.

Example

```
create or replace outline on aggregate_storage database Sample2.Basic2
as outline on database sample.basic;
```

Creates an aggregate storage outline based on the Sample.Basic outline. For a complete example of how to create an aggregate storage version of a block storage cube, see Creating an Aggregate Storage Sample Using MaxL.

Display Tablespace (Aggregate Storage)

View details about a tablespace.

Tablespaces are applicable only to aggregate storage databases.

Permission required: Application Manager.

This statement requires the application to be started.

Syntax

TABLSP-NAME

Example

```
set column_width 50; /* so file_location will not be truncated */
display tablespace ASOsamp.'default';
```

This example displays the following output:

Table 3-17 Display Tablespace MaxL Output Columns

Column Header	Contents
file_location	C:\Hyperion\products\Essbase\EssbaseServer\APP\
max_file_size	56
max_disk_size	4294967295

Execute Allocation

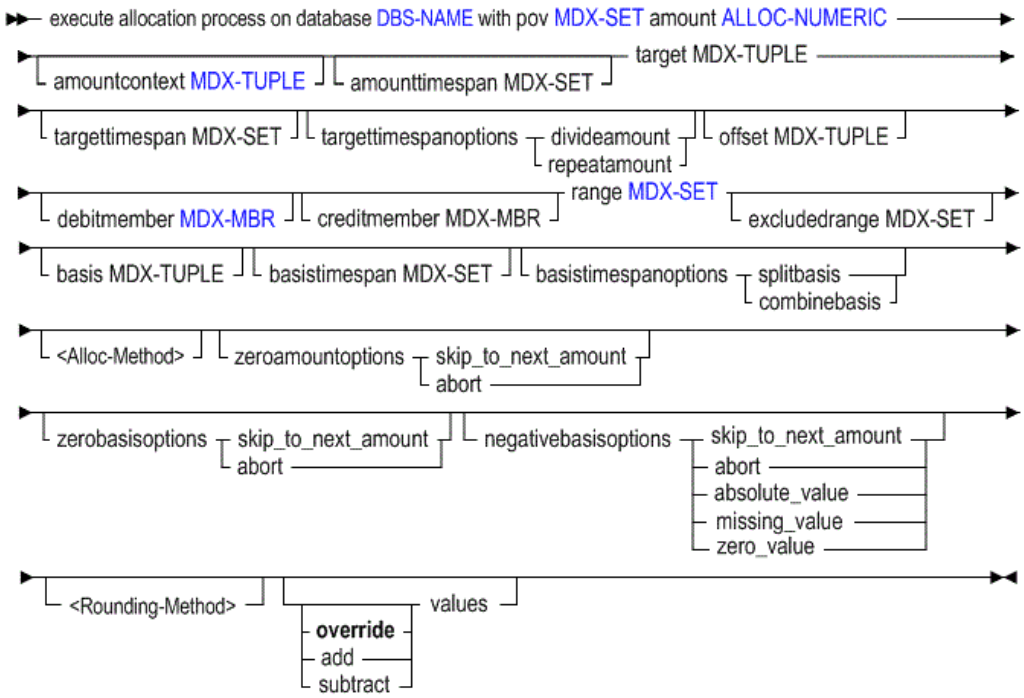
Allocate one or more given source amounts to a target range of cells in an aggregate storage database. The source amount can be allocated to the target proportionately to a given basis, or the source amount can be spread evenly to the target region.

Allocations are typically used in the budgeting process to distribute revenues or costs.

Minimum permission required: Execute.

For more information about allocations and to understand the input parameters, see Performing Custom Calculations and Allocations on Aggregate Storage Databases.

Syntax



- DBS-NAME
- MDX-SET
- ALLOC-NUMERIC
- MDX-TUPLE
- MDX-MBR

Keywords

pov <mdx-set>
Required. Provide an MDX set defining the context region in which the allocation is performed.

amount <alloc-numeric>
Required. Provide an MDX numeric value expression indicating the amount to be allocated.

amountcontext <mdx-tuple>
Optional. Provide an MDX tuple with one member from each dimension missing from **pov** and **amount**. This clause is required when **amount** is an arithmetic expression and **pov** does not specify two or more dimensions. It should not be used otherwise.

amounttimespan <mdx-set>
Optional. Provide an MDX set indicating one or more time periods to be considered for the amount. The amount value is aggregated over the specified time periods, and the aggregated amount value is allocated. Time periods must be level 0 members in a Time dimension.

target <mdx-tuple>
Required. Provide an MDX tuple defining the database region where results are written.

targettimespan <mdx-set>
Optional. Provide an MDX set indicating one or more time periods to be considered for the target. Time periods must be level 0 members in a Time dimension.

targettimespanoptions
Optional, but required if `targettimespan` is used.
Select a method for allocating values across the target time span:

- `divideamount`–Divide the amount evenly across the time periods
- `repeatamount`–Repeat the amount across the time periods

offset <mdx-tuple>
Optional. If offsetting entries are used, provide an MDX tuple defining the location in the database where an offsetting value is written for each source amount.

debitmember <mdx-mbr>
Optional. If double-entry accounting is used, provide an MDX member expression indicating the member to which positive result values are written.

creditmember <mdx-mbr>
Optional. If double-entry accounting is used, provide an MDX member expression indicating the member to which negative result values are written.

range <mdx-set>
Required. Provide an MDX set indicating the database region in which allocated values are calculated and written.

excludedrange <mdx-set>
Optional. Provide an MDX set specifying locations in the range where you do not want allocation values written.

basis <mdx-tuple>
Required in most cases. Provide an MDX tuple that, when combined with the range, defines the location of basis values that determine how the amount is allocated. The basis can consist of upper-level or level 0 members.

Optional if the allocation method used is **spread**, and no values are skipped; required otherwise. Basis must be omitted when the allocation method **spread** is used without **skip** options.

basistimespan <mdx-set>
Optional. Provide an MDX set that indicates one or more time periods to be considered for the basis. Time periods must be level 0 members in a Time dimension.

basistimespanoptions
Optional, but required if **basistimespan** is used. Select a method for using the basis time span:

- `splitbasis`–Use the basis value for each time period individually
- `combinebasis`–Use the sum of the basis values across the time periods specified by **basistimespan**

share
Optional. Specify to allocate the amount(s) proportionately to the basis values. For syntax, see Allocation Method Specification in Notes.

spread
Optional. Specify to allocate the amount(s) evenly. For syntax, see Allocation Method Specification in Notes. You can include one or more of the following skip options when using spread allocation:

- `skip_missing`–Skip missing basis values
- `skip_zero`–Skip zero basis values
- `skip_negative`–Skip negative basis values

zeroamountoptions
Optional. If omitted, zero or #MISSING amount values are allocated. Otherwise, specify treatment of amount values that are zero or #MISSING:

- `skip_to_next_amount`–Skip to the next nonzero, non-#MISSING amount value
- `abort`–Cancel the entire allocation operation

zerobasisoptions
Optional. For **share**, this option specifies the action when the sum of all basis values is zero. For **spread**, this option specifies the action when all the basis values are skipped. Select one of the following options:

- `skip_to_next_amount`–Skip to the next nonzero, non-#MISSING amount value
- `abort`–Cancel the entire allocation operation

round
Optional. Specify rounding options. The following options are available:

- Round to a specified number of decimal places, using an integer or MDX numeric value expression. The value must be between 100 and -100, and is truncated if it is not a whole number.
- Perform rounding, but discard rounding errors
- Add rounding errors to the highest allocated value

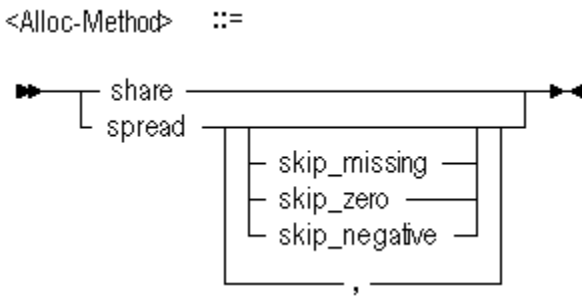
- Add rounding errors to the lowest allocated value
- Provide an MDX tuple indicating a cell to which the rounding error should be added

override|add|subtract values
Optional. Generated allocation values can be added to (or subtracted from) existing values, instead of overwriting them. Overwriting is the default.

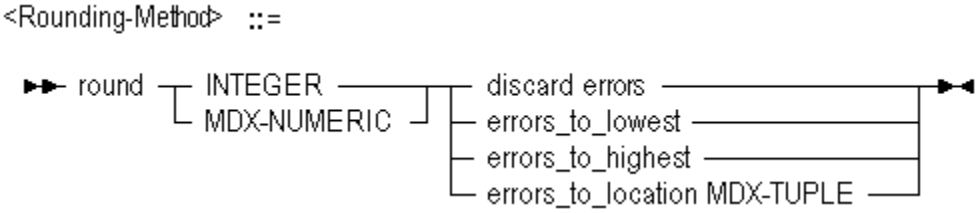
Notes

- The clauses following the **with** keyword can be entered in any order, each separated by white space.
- Each clause can only be entered once.
- The **pov**, **amount** , **target**, **range**, and **basis** clauses are mandatory; the others are optional.
- You can specify only stored, level-0 members in all of the clauses except for **amount**, **amountcontext**, **basis**, and the number of rounding digits; for all other arguments, do not use upper-level members, attribute members, or dynamic calc members.

Allocation Method Specification



Rounding Method Specification



Example

The following statement executes an allocation.

```
execute allocation process on database glrpt.db with
pov             "Crossjoin({[VisionUS]},
                Crossjoin({[5740]},
```

```
                  Crossjoin({[USD]},
                    Descendants([Geography],[Geography].Levels(0)))))"
amount          "Jan + Feb"
amountcontext     "([100], [Beginning Balance], [Actual],
[CostCenter1])"
target          "([Allocation], [CostCenter1])"
offset          "([Allocation], [CostCenter1], [100], [YearNA])"
debitmember      "[Debit]"
creditmember      "[Credit]"
range          "Crossjoin(Descendants([999], [Department].Levels(0)),
           Descendants([Year], [Year].Levels(0)))"
excludedrange      "{[9994], [9995], [9996]}"
basis          "([SQFT], [Balance], [Actual], [CostCenter2])"
share
zeroamountoptions    abort
zerobasisoptions  abort
negativebasisoptions  zero_value
targettimespanoptions  divideamount
round            "Currency.CurrentMember.CurrencyPrecision"
errors_to_location      "([101], [Jan])"
add values;
```

Execute Calculation (Aggregate Storage)

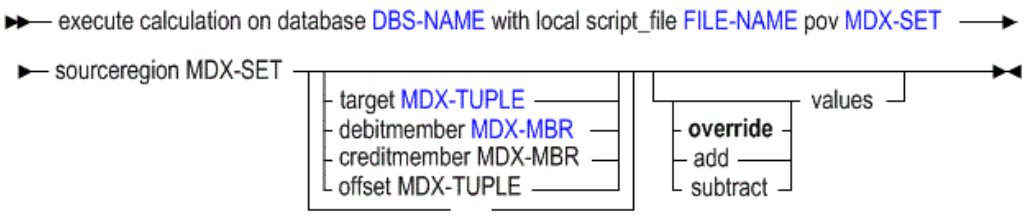
Click here for non-aggregate storage version

Execute a custom calculation script expressed in MDX, specifying the script file, source region, and point of view (POV). Optionally specify the target, offset, and debit or credit members.

Minimum permission required: Database Update.

For more information about custom calculation script parameters, see Performing Custom Calculations and Allocations on Aggregate Storage Databases.

Syntax



- DBS-NAME
- FILE-NAME
- MDX-SET
- MDX-TUPLE
- MDX-MBR

You can execute custom calculations with the following options:

Keywords

local script_file
Required. Run the specified local calculation script file. Custom calculation scripts are expressed in MDX. The following is an example of a custom calculation script, `script.txt`.

```
(AccountA,Proj1) := 100;
([AccountB], [Proj1]) := ([AccountB], [Proj1]) * 1.1;
(AccountC,Proj1) :=
    ((AccountB,Proj1,2007) + (AccountB, Proj1)) / 2;
(AccountA,Proj2) :=
    ((AccountD,Proj1) +
      (AccountB,Proj2)) / 2;
```

For information about writing custom calculation scripts, see Performing Custom Calculations and Allocations on Aggregate Storage Databases.

pov <mdx-set>
Required. Provide an MDX set defining the context region in which the calculation is performed. The calculation script will be executed once for every cross-product in the POV region.

sourceregion <mdx-set>
Required. Provide an MDX set specifying the region of the cube referred to by the formulas in the script. At a minimum, the source region should include all members from the right-hand sides of the assignment statements in the custom calculation script.

target <mdx-tuple>
Optional. Provide an MDX tuple defining the database region where results are written. You can use only stored, level-0 members in the tuple; do not use upper-level members, attribute members, or dynamic calc members.

debitmember <mdx-mbr>
Optional. If double-entry accounting is used, provide an MDX member expression indicating the member to which positive result values are written. You can specify only stored, level-0 members; do not use upper-level members, attribute members, or dynamic calc members.

creditmember <mdx-mbr>
Optional. If double-entry accounting is used, provide an MDX member expression indicating the member to which negative result values are written. You can specify only stored, level-0 members; do not use upper-level members, attribute members, or dynamic calc members.

offset <mdx-tuple>
Optional. If offsetting entries are used, provide an MDX tuple defining the location in the database where an offsetting value for each source amount is written. You can use only stored, level-0 members in the tuple; do not use upper-level members, attribute members, or dynamic calc members.

override|add|subtract values
Optional. Generated calculation values can be added to (or subtracted from) existing values, instead of overwriting them. Overwriting is the default.

Notes

- Each clause can only be entered once.
- The **script_file**, **pov**, and **sourceregion** clauses are mandatory; the others are optional.
- The optional clauses following the **sourceregion** specification can be entered in any order, each separated by white space.
- You can specify only stored, level-0 members on the left side of the assignment statement in the custom calculation script; do not use upper-level members, attribute members, or dynamic calc members.
- You can specify only stored, level-0 members in the following clauses: DebitMember, CreditMember, Target, and Offset.

Example

The following statement executes `script.txt` referenced above. For a sample use case, see Performing Custom Calculations and Allocations on Aggregate Storage Databases.

```
execute calculation on database app.db with
  local script_file "script.txt"
  POV "Crossjoin({[VisionUS]},
    Crossjoin({[101]},
      Crossjoin ({[Jan]},
        Crossjoin({[Scenario]},
          Descendants(Geography, Geography.Levels(0))))))"
  SourceRegion "Crossjoin({[AccountB], [AccountD]},
    Crossjoin({[Proj1], [Proj2]}, {[2007]}))"
  Target "(Allocation)"
  DebitMember "[BeginningBalance_Debit]"
  CreditMember "[BeginningBalance_Credit]"
  Offset  "([Account_000], [Project_000])"
  add values;
```

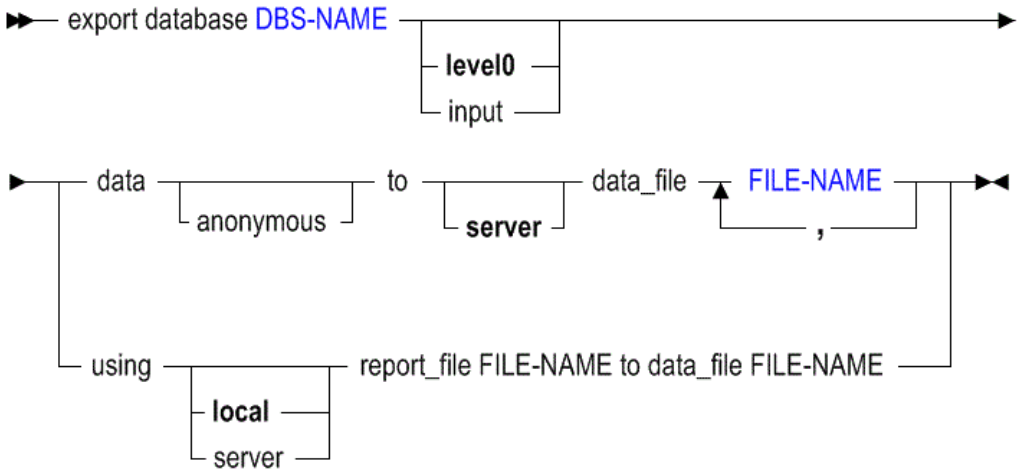
Export Data (Aggregate Storage)

Click here for non-aggregate storage version

Export level-0 data, which does not include calculated values, from an aggregate storage database. Export data files are written to the application directory, unless an absolute path is specified. To use Report Writer, export the data using a report file.

Minimum permission required: Read.

Syntax



- DBS-NAME
- FILE-NAME

On aggregate storage databases, use **export data** to export in the following ways:

Keywords

export database <dbs-name> level0 data...
Export level-0 input data to a text file. You cannot export aggregates, upper level data, or data from dynamically calculated members.

> **Note:**
>
> Exporting data does not clear the data from the database.

export database <dbs-name> input data...
This statement performs the same action as **export database <dbs-name> level0 data**....

export database <dbs-name> ... data anonymous
Export data in anonymized format. Anonymization removes the risk of sensitive data disclosure, and can be used in case sample data needs to be provided for technical support. Essbase replaces real data values with 1, for each value in the block.

export database <dbs-name> ...using...report_file...
Run a stored report script, exporting a subset of the database.

Notes

- This statement requires the database to be started.
- Exports on aggregate storage databases are limited as follows:
  - You can export level-0 data only (level-0 data is the same as input data in aggregate storage databases).
  - You cannot perform upper-level data export on an aggregate storage database.

- You cannot perform columnar export on an aggregate storage database.
- To export data in parallel, specify a comma-separated list of export files, from 1 to 8 file names. This number should generally be equal to the number of processors on the machine that you wish to commit to doing parallel export. The number of threads Essbase uses typically depends on the number of file names you specify. However, on a very small aggregate storage database with a small number of data blocks, it is possible that only a single file will be created (in effect, performing serial export), even though parallel export to multiple files is requested. In this case, the export file name will be the first file name given as input.
- During a data export, the export process allows users to connect and perform read-only operations.
- If the data for a thread exceeds 2 GB, Essbase may divide the export data into multiple files with numbers appended to the file names.

  The naming convention for additional export files is as follows: `_1`, `_2`, etc. are appended to the additional file names. If the specified output file name contains a period, the numbers are appended before the period. Otherwise, they are appended at the end of the file name.

  For example, if the given file name is `exportfile.txt`, the next additional file is `exportfile_1.txt`.

Example

Example 1

The following example exports all level 0 data from ASOsamp.Sample to an export file.

```
export database ASOsamp.Sample data to server data_file
'myfilesamp.txt';
```

Example 2

The following example uses a report script, `Bottom.rep`, to export a subset of sorted data from ASOsamp.Sample to an output file, `Bottom.rpt`.

```
export database ASOsamp.Sample using report_file 'Bottom.rep' to
data_file 'Bottom.rpt';
```

Sample Report Script and Output

For example 2, assume that `Bottom.rep` is the following report script file based on ASOsamp.Sample:

```
//Bottom.rep
<Sym
<Column (Measures, Years)
<Row (Geography, Products)
<ICHILDREN Geography
<ICHILDREN Products
<Bottom (3, @DataColumn(1))
    !
```

The report script produces the following report (`Bottom.rpt`):

```
Measures Years Time Transaction Type Payment Type Promotions Age Income
Level Stores

North East       All Merchandise  43,250,241
                   Products       43,250,241
                 High End Mercha~ 11,379,402
South            All Merchandise  32,790,838
                   Products       32,790,838
                 High End Mercha~ 8,436,598
  Geography      All Merchandise  76,041,079
                   Products       76,041,079
                 High End Mercha~ 19,816,000
```

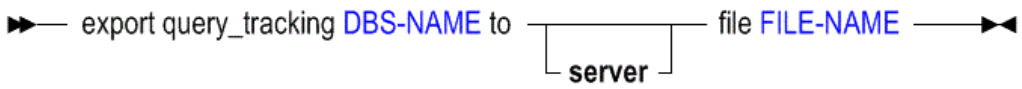
Export Query Tracking (Aggregate Storage)

Export query data from an aggregate storage database to a text file.

Query data can be used to select the most appropriate set of aggregate views to materialize for an aggregate storage database. When an aggregate storage database is refreshed or restarted, query data is not persisted in the database. To rebuild aggregate views after a database refresh or restart, query data can be exported to and then imported from a text file.

Permission required: Database Manager.

Syntax



- DBS-NAME
- FILE-NAME

You can use **export query_tracking** in the following ways.

Keywords

export query_tracking <dbs-name> to server file...
Export query data from the specified aggregate storage database to the specified file. For FILE-NAME, specify the name, including the path, of the text file that contains the query data to export. If a path is not specified, the file is created in the *$ARBORPATH*/app/*appname*/*dbname* folder.

export query_tracking <dbs-name> to file...
You can omit the **server** keyword, but the result is the same.

Notes

- To export and import query data, query tracking must be enabled for the aggregate storage database. Use the **alter database** (aggregate storage) statement with the **enable query_tracking** grammar. Query tracking is disabled by default.
- Do not edit the text file with the exported query data.

Example

```
export query_tracking ASOsamp.Sample to server file
'query_data_aso_sample.txt';
```

Exports query data from the ASOsamp.Sample database to the named file.

See Also

Import Query Tracking (Aggregate Storage)

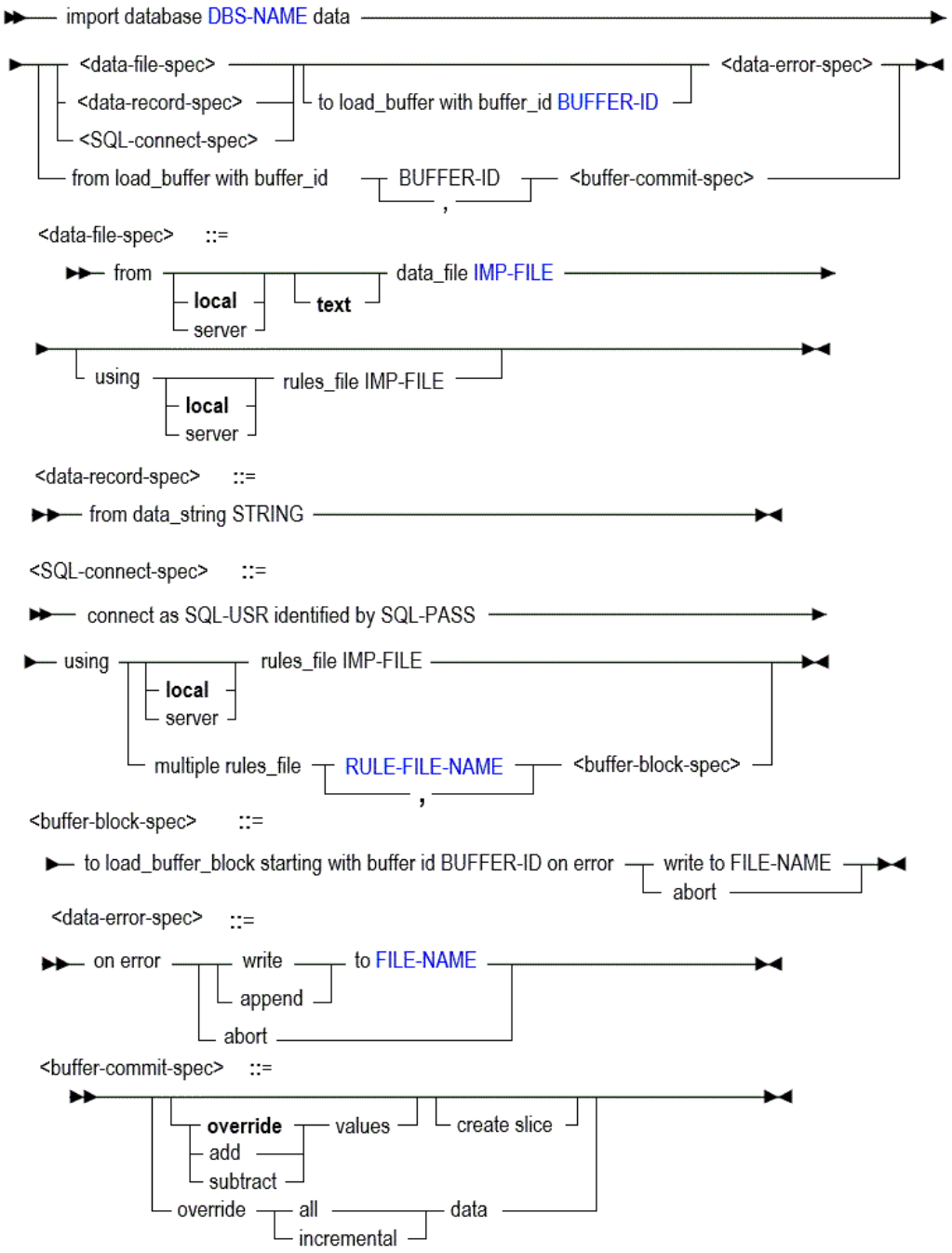
Import Data (Aggregate Storage)

Click here for non-aggregate storage version

Import data from text or spreadsheet data files, with or without a rules file.

Minimum permission required: Write.

Syntax



- DBS-NAME
- BUFFER-ID
- IMP-FILE
- RULE-FILE-NAME
- FILE-NAME

Use **import data** in the following ways to load data into an aggregate storage database:

Keywords

import database <dbs-name> data from...
Specify whether the data import is from a local or server file, and what type of file to import data from.

...using ... rules_file
Import data into the database using a specified rules file.

...<data error spec> (on error...)
Required. Tell Essbase what to do in case of errors during the data load: abort the operation, or write or append to a specified error log.

...<data record spec> from data_string
Load a single data record into the selected database. The string following **data_string** must be a contiguous line, without newline characters.

...<SQL connect spec> (connect as...)
If you are importing data from an SQL source, provide your SQL user name and password. You must always use a rules file when you load SQL data sources.
When loading SQL data into aggregate storage databases, you can use up to eight rules files to load data in parallel by using the **multiple rules_file** grammar with the grammar specified in <buffer-block-spec>. Essbase initializes multiple temporary aggregate storage data load buffers (one for each rules file) and, when the data is fully loaded into the buffers, commits the contents of all buffers into the database in one operation.
Each rules file must use the same authentication information (SQL user name and password).
In the following example, SQL data is loaded from two rules files (`rule1.rul` and `rule2.rul`):

```
import database ASOsamp.Sample data
   connect as TBC identified by 'password'
   using multiple rules_file 'rule1','rule2'
   to load_buffer_block starting with buffer_id 100
   on error write to "error.txt";
```

In specifying the list of rules files, use a comma-separated string of rules file names (excluding the `.rul` extension). The file name for rules files must not exceed eight bytes and the rules files must reside on Essbase Server.
In initializing a data load buffer for each rules file, Essbase uses the starting data load buffer ID you specify for the first rules file in the list (for example, ID 100 for rule1) and increments the ID number by one for each subsequent data load buffer (for example, ID 101 for rule2).
The ODBC driver you are using must be configured for parallel SQL connections.

> **Note:**
>
> Performing multiple SQL data loads in parallel to aggregate storage databases is different than using the **to load_buffer with buffer_id** grammar to load data into a buffer, and then using the **from load_buffer with buffer_id** grammar to explicitly commit the buffer contents to the database.

...to load_buffer with buffer_id
If you are importing data from multiple data files to an aggregate storage database, you can import to a buffer first, in order to make the data import operation more efficient.

...from load_buffer with buffer_id
If you are importing data from multiple data files to an aggregate storage database, you can import from a data load buffer in order to make the data import operation more efficient.

...from load_buffer with buffer_id...values
Specify whether you want to add to existing values, substract from existing values, or override existing values when committing the contents of the specified data load buffer to the database.

...from load_buffer with buffer_id...create slice
Commit the contents of the specified data load buffer to the database by creating a new data slice.

...from load_buffer with buffer_id override all data
Remove the current contents of the database and replace the database with the contents of the specified data load buffer.

...from load_buffer with buffer_id override incremental data
Remove the current contents of all incremental data slices in the database and create a new data slice with the contents of the specified data load buffer. The new data is created with the data load property "add values" (aggregate_sum). If there are duplicate cells between the new data and the primary slice, their values are added together when you query for them.

Notes

- This statement requires that the database is started.
- When using the import statement, you must specify what should happen in case of an error.
- To import from a SQL data source, you must connect as the relational user name and use a rules file.

Example

```
import database ASOsamp.Sample data from server data_file '/ASOsamp/
Sample/expsamp.txt' on error abort;
```

Loads data into the ASOsamp.Sample database.

```
import database ASOsamp.Sample data from load_buffer with buffer_id 1;
```

Commits the contents of a specified data load buffer to the ASOsamp.Sample database.

```
import database ASOsamp.Sample data from load_buffer with buffer_id 1,
2;
```

Commits the contents of multiple data load buffers (buffer_id 1 and buffer_id 2) to the ASOsamp.Sample database.

```
import database ASOsamp.Sample data from load_buffer with buffer_id 1
add values;
```

Commits the contents of a specified data load buffer to the ASOsamp.Sample database by adding values.

```
import database ASOsamp.Sample data from load_buffer with buffer_id 1
override values create slice;
```

Commits the contents of the specified data load buffer into a new data slice in the ASOsamp.Sample database.

```
import database ASOsamp.Sample data from load_buffer with buffer_id 1
override all data;
```

Replaces the contents of the ASOsamp.Sample database with the contents of the specified data load buffer.

```
import database ASOsamp.Sample data from load_buffer with buffer_id 1
override incremental data;
```

Replaces the contents of all incremental data slices in the ASOsamp.Sample database by creating a new data slice with the contents of the specified data load buffer. The new data is created with the data load property "add values" (aggregate_sum). If there are duplicate cells between the new data and the primary slice, their values are added together when you query for them.

See Loading Data Using Buffers.

Import Query Tracking (Aggregate Storage)

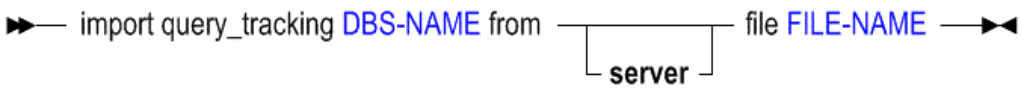
Import query data, which was previously exported from an aggregate storage database to a text file, to an aggregate storage database.

Query data can be used to select the most appropriate set of aggregate views to materialize for an aggregate storage database. When an aggregate storage database is refreshed or restarted, query data is not persisted in the database. To rebuild

aggregate views after a database refresh or restart, query data can be exported to and then imported from a text file.

Permission required: Database Manager.

Syntax



- DBS-NAME
- FILE-NAME

You can use **import query_tracking** in the following ways.

Keywords

import query_tracking <dbs-name> from server file...
Import query data to the specified aggregate storage database from the specified file. For FILE-NAME, specify the name, including the path, of the text file that contains the query data to import. By default, the file is created in the database directory.

import query_tracking <dbs-name> from file...
You can omit the **server** keyword, but the result is the same.

Notes

- To export and import query data, query tracking must be enabled for the aggregate storage database. Use the **alter database** (aggregate storage) statement with the **enable query_tracking** grammar. Query tracking is disabled by default.
- Do not edit the text file with the exported query data.

Example

```
import query_tracking ASOsamp.Sample from server file "'$ARBORPATH/app/
ASOsamp/Sample/query_data_aso_sample.txt'";
```

Imports the query data from the named file to the ASOsamp.Sample database.

See Also

Export Query Tracking (Aggregate Storage)

Query Application (Aggregate Storage)

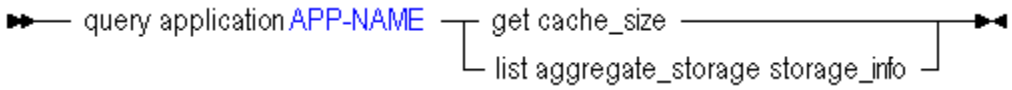
Click here for block storage version

Get information about the current state of the application.

This statement is only applicable for aggregate storage applications.

This statement requires the application to be started.

Syntax



APP-NAME

Example

The following MaxL statement:

```
query application sample get cache_size;
```

returns the maximum size (in kilobytes) to which the aggregate storage cache may grow.

The following MaxL statement:

```
query application asoapp list aggregate_storage storage_info;
```

returns the following information:

Table 3-18 Query Application List Aggregate Storage Storage_Info MaxL Output Columns

Output Columns	Description
`Cache hit ratio`	Ratio of the number of requests answered from aggregate storage cache as opposed to from the hard disk. **Note:** This statistic may not be accurate when parallel data load or parallel calculation operations are in use.
`Current cache size (KB)`	The current size of the aggregate storage cache. See description for `current cache size limit (KB)`.
`Current cache size limit (KB)`	The maximum size (in kilobytes) to which the aggregate storage cache may grow.
`Page reads since last startup`	Number of data blocks (pages) read from disk since the last time the application was started.

Table 3-18 (Cont.) Query Application List Aggregate Storage Storage_Info MaxL Output Columns

Output Columns	Description
`Page writes since last startup`	Number of data blocks (pages) written to disk since the last time the application was started.
`Page size (KB)`	Size of the data block (page) in kilobytes.
`Disk space allocated for data (KB)`	Total space used by all disk files in the `default` tablespace.
`Disk space used by data (KB)`	Total space actually in use within the disk files in the `default` tablespace (some space within files may be free).
`Temporary disk space allocated (KB)`	Total space used by all disk files in the `temp` tablespace.
`Temporary disk space used (KB)`	Total space actually in use within the disk files in the `temp` tablespace (some space within files may be free).

Query Database (Aggregate Storage)

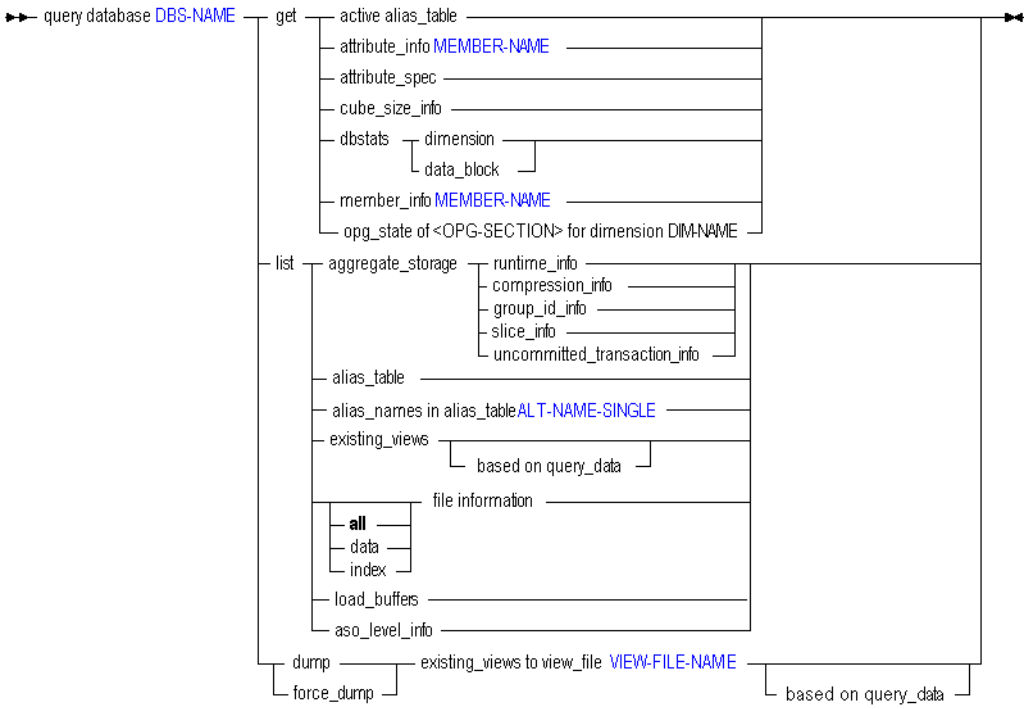
Click here for non-aggregate storage version

Get advanced information about the current state of the database.

Minimum permission required: Read.

This statement requires the database to be started.

Syntax



- DBS-NAME
- MEMBER-NAME
- ALT-NAME-SINGLE
- VIEW-FILE-NAME

You can query for database information in the following ways using **query database**:

Keywords

get active alias_table
Display the active alias table for the user issuing the statement.

get attribute_info
Get attribute member, dimension, and name information for the specified attribute member.

get attribute_spec
Display the current attribute specifications for the database. These specifications include attribute member name format, Attribute Calculation dimension member names, Boolean and date member names, and numeric range specifications.

get cube_size_info
Display information about input data size, aggregated data size, and number of queries tracked (when query tracking is enabled).
This statement returns the output listed in the following table:

Column Name	Contents
input_data_size_cells	Number of input-level cells in the cube.
input_data_size_bytes	Number of bytes used by the input-level data (approximate).
aggregate_data_size_cells	Total number of cells in all aggregate views in the cube.
aggregate_data_size_bytes	Number of bytes used by the aggregate cells (approximate).
kernel_queries_tracked	Number of kernel queries executed since the last time query tracking was enabled or query tracking information was reset.
total_query_cost	Total cost of all queries executed since the last time query tracking information was reset.

Column Name	Contents
query_tracking_enabled	Values: True or False. Tells whether user retrieval statistics are being collected for the aggregate storage database. The statistics can be used by the following MaxL statements for query-based view optimization: • query database <dbs-name> list existing_views • execute aggregate process • execute aggregate selection Query tracking is disabled by default.

get dbstats dimension
Get information about dimensions.
The **index_type** field values are numeric, and translate as follows:

```
0               Dense
1               Sparse
3               None (database is aggregate storage)
```

get dbstats data_block
Get information about data blocks. The information returned has little relevance to aggregate storage databases.

get member_info <MEMBER-NAME>
Get information on a specific member.
Output
The **unary_type** field values are numeric, and translate as follows:

```
0               Add
1               Subtract
2               Multiply
3               Divide
4               Percent
5               NoRollUp
```

The **member_tag_type** field values translate as follows:

```
0               SkipNone
16384           SkipMissing
32768           SkipZero
49152           SkipBoth
1               BalFirst
2               BalLast
4               TwoPass
8               Average
64              Expense
```

Variations are possible. The field value consists of one of the first four "skip" values plus any/all/none of the last five values. Some examples:

```
0               SkipNone
77              SkipNone, BalFirst, TwoPass, Average, Expense
16385           SkipMissing and BalFirst
```

The first four "skip" values are base values, and added to them are combinations of 1, 2, 4, 8, and 64.
The **status** field values are hexadecimal, and translate as follows:

```
0               Normal
1               Never Share
2               Label
4               Refer Share
8               Refer Share (with different name)
16              Implicit share
32              Virtual Member (stored)
64              Virtual Member (not stored)
2048            Attribute
32768           Referred
```

get opg_state of member_data
Display outline navigational information (for example, parent, child, or sibling), fixed-length information (for example, the line aggregation symbol or the number of children), and text strings (for example, member names or aliases).
See Outline Paging Dimension Statistics for a description of the output.

get opg_state of member_name_namespace
Display information that matches member names to internal member identifiers (one section per database, thus the information for all dimensions is the same).
See Outline Paging Dimension Statistics for a description of the output.

get opg_state of member_formula
Display all formulas for the dimension.
See Outline Paging Dimension Statistics for a description of the output.

get opg_state of member_UDA
Display all user defined attributes (UDAs) for the dimension.
See Outline Paging Dimension Statistics for a description of the output.

get opg_state of member_UDA_namespace
Display information that matches UDAs to internal member identifiers.
See Outline Paging Dimension Statistics for a description of the output.

get opg_state of attribute_to_base_member_association
Display information that identifies the attribute member associated with each base member of the dimension.
See Outline Paging Dimension Statistics for a description of the output.

get opg_state of member_comment
Display all member comments for the dimension.
See Outline Paging Dimension Statistics for a description of the output.

get opg_state of member_alias_namespace
Display information that matches member alias names to internal member identifiers (one section per alias table, thus the information for all dimensions is the same). See Outline Paging Dimension Statistics for a description of the output.

list aggregate_storage runtime_info
Display runtime statistics about the aggregate storage database. For a description of the output returned by this statement, see Aggregate Storage Runtime Statistics.

list aggregate_storage group_id_info
Display information about group IDs and their timestamps related to General Ledger cubes.

> **Note:**
>
> This grammar applies to General Ledger cubes, not to non-general-ledger aggregate storage databases. For normal aggregate storage databases, this table will be empty.
> This MaxL grammar is disabled for previous release Essbase MaxL clients.

This statement returns the following output:

Column Name	Contents
group_id	The allocation group id, according to the begin allocation command that created the allocation group. The number is an unsigned 64-bit integer.
transaction_id	The aggregate storage transaction ID that is used internally. The number is an unsigned 64-bit integer.
state	A string describing the state of the group ID. For example: BeginAllocation Done, Allocation In Progress, Allocation Done, EndAllocation In Progress.
time_last_used	The date and time the group ID was last used. The value is either the time the group ID was created or the time that an allocation or custom calculation was last performed with this group ID. The value is a string.
time_expired	The date and time when the group ID will time out (expire). The value is a string.

Column Name	Contents
expired	Indicates whether the group ID has timed out. If the group ID has expired, the group ID will be rolled back the next time a begin allocation command is executed. The value is a boolean.

For a description of the output returned by this statement, see Aggregate Storage Group ID Information Output.

list aggregate_storage slice_info
Display information about data slices and views, some information of which applies only to General Ledger cubes (not to non-general-ledger aggregate storage databases).

> **Note:**
>
> Small incremental slices may have fewer aggregate views than the primary slice (slice number 0). Incremental slices with less than 100,000 cells will never have any aggregate views built. However, if an incremental slice is larger than 100,000 cells and it is larger than the primary slice, then it will always have the same aggregate views as the primary slice.
> This MaxL grammar is disabled for previous release Essbase MaxL clients.

This statement returns the following output:

Column Name	Contents
transaction_id	**(Applies to General Ledger cubes only)** The ID of the transaction to which this slice and view belong. There is one transaction ID for each GL group ID. The number is an unsigned 64-bit integer. To find the corresponding group ID, use the following MaxL command: `query database app.db list aggregate_storage group_id_info;` For non-general-ledger aggregate storage databases, this number is always 0.
slice_id	ID number of the data slice. The number is an unsigned 32-bit integer.

Column Name	Contents
slice_tag	**(Applies to General Ledger cubes only)** When an allocation or custom calculation is done within an allocation begin/end, this number is the rule_id of the allocation that made this data slice. The number is an unsigned 64-bit integer. For non-general-ledger aggregate storage databases, this number is always 0.
view_id	0 indicates an input view; otherwise, the view is an aggregate view. The number is an unsigned 64-bit integer. To list the levels in a given aggregate view, use the following MaxL command: `query database app.db list existing_views;`
size_cells	The number of cells in the given view of the slice. The number is an unsigned 64-bit integer.
size_kb	The size in KB of the given view of the slice. The number is an unsigned 64-bit integer.

For a description of the output returned by this statement, see Aggregate Storage Slice Information Output.

list aggregate_storage uncommitted_transaction_info
Display information about uncommitted transactions that are related to General Ledger cubes.

> **Note:**
>
> This grammar applies to General Ledger cubes, not to non-general-ledger aggregate storage databases. For normal aggregate storage databases, this table will be empty.
> This MaxL grammar is disabled for previous release Essbase MaxL clients.

This statement returns the following output:

Column Name	Description
unc_transactions	The number of existing user transactions that are not yet committed.
unc_data_slices	The number of data slices used by uncommitted transactions.
unc_input_data_size_cells	The number of input cells used by uncommitted transactions.
unc_aggregate_views	The number of aggregate views used by uncommitted transactions.
unc_aggregate_data_size_cells	The number of aggregate cells used by uncommitted transactions.
unc_input_data_size_kb	The total disk space used by uncommitted input-level data.
unc_aggregate_data_size_kb	The total disk space occupied by uncommitted aggregate cells.

For a description of the output returned by this statement, see Aggregate Storage Uncommitted Transaction Information Output.

list aggregate_storage compression_info
Display estimated compression for aggregate storage databases when different dimensions are hypothetically used as the compression dimension. These estimates can help you choose the best dimension to use as the compression dimension.
In aggregate storage databases, the compression dimension enables database compression. A good candidate for a compression dimension is one that optimizes data compression and maintains retrieval performance. The following table lists data for all non-attribute dimensions, even though it may not be possible to select them as the compression dimension without significant changes to the outline. For information on the requirements of a compression dimension, see Understanding the Compression Dimension for Aggregate Storage Databases.
This statement returns the following output:

Column Name	Contents
dimension_name	Each dimension name in the database, hypothetically considered to be the compression dimension.
is_compression	Indicates whether the dimension is the aggregate storage compression dimension. (There can be only one compression dimension in an aggregate storage database.)
stored_level0_members	The number of leaf-level members in the dimension. A large number of stored level-0 members in a dimension indicates that it may not perform well as a compression dimension.

Column Name	Contents
average_bundle_fill	Estimated average number of values per compression dimension bundle. Choosing a compression dimension that has a higher average bundle fill means that the database compresses better.
average_value_length	Estimated average number of bytes required to store a value. Dimensions with a smaller average value length compress the database better.
level0_mb	Estimated size of the compressed database, in megabytes. A smaller expected level-0 size indicates that choosing this dimension enables better compression. Except for the scenario in which there is no compression dimension (*None*), all estimates assume that all pages are compressed. Since compressed pages require additional overhead that uncompressed pages do not, the estimated level-0 database size for some dimensions may be larger than the value for *None*.

list alias_table
Get a list of alias tables that are defined for the database.

list alias_names in alias_table
List the alias names defined in an alias table. Alias tables contain sets of aliases for member names and are stored in the database outline. Use this grammar to see a list of alias names defined in the specified table.

list existing_views
Display information about all aggregate views. An aggregate view is a collection of aggregate cells based on the levels of the members within each dimension.
The optional **based on query_data** clause causes the returned query cost information to be based on the collected cost of actual user queries. If this clause is not used, the default assumption is that all possible queries happen with the same probability.
To use the **based on query_data** clause, query tracking must first be enabled. To enable query tracking, use alter database <dbs-name> enable query tracking.

list ... file information
Get accurate index and data file information. Provides index and data file names, counts, sizes, and totals, and indicates whether or not each file is presently opened by Essbase. The file size information is accurate. Note that the file size information provided by the Windows operating system for index and data files that reside on NTFS volumes may not be accurate.

list load_buffers
Display a list and description of the data load buffers that exist on an aggregate storage database. See Using Aggregate Storage Data Load Buffers.

list aso_level_info
Display the aggregation level count for each real dimension in the outline. Aggregation level count is the total number of aggregation levels in a real dimension (including associated attribute dimensions) that exist on an aggregate storage database.

dump|force_dump existing views...
Saves existing views of this database to an aggregation script. This action requires a minimum permission of execute (Execute).
If the specified script name already exists, you can use the **force_dump** keyword to overwrite it; otherwise, an error is returned if the file name already exists.
If the **based on query_data** phrase is used, the view selection that is saved will be based on previously collected query-tracking data. You must have enabled query tracking to use this option. For more information about query tracking, see the `based on query_data` description in execute aggregate selection.

Example

```
query database ASOsamp.Sample list load_buffers;
```

Display a list and description of the data load buffers that exist on ASOsamp.Sample.

Outline Paging Dimension Statistics

The following columns are the output of the MaxL statement beginning with query database DBS-NAME get opg_state.

This statement is only applicable to databases using aggregate storage.

Table 3-24 Outline Paging Dimension Statistics MaxL Output Columns

Column Name	Contents
`version`	The version of the outline paging section (a Berkeley DB database).
`unique_keys`	The number of unique keys in the outline paging section.
`key/data_pairs`	The number of key/data pairs in the outline paging section.
`page_size`	The page size (in bytes) of the underlying database.
`minimum_keys_per_page`	The minimum number of keys per page.
`length of fixed_length_records`	The length of the fixed-length records (only available when the outline paging section is a Recno database).
`padding_byte_value_for_fixed_length_columns`	The padding byte value for fixed-length records.
`levels`	Number of levels in the underlying database corresponding to the outline paging section.

Table 3-24 (Cont.) Outline Paging Dimension Statistics MaxL Output Columns

Column Name	Contents
internal_pages	Number of internal pages in the underlying database.
leaf_pages	Number of leaf pages in the underlying database.
duplicate_pages	Number of duplicate pages in the underlying database.
overflow_pages	Number of overflow pages in the underlying database.
pages_on_free_list	Number of pages on the free list in the underlying database.
bytes_free_in_internal_pages	Number of bytes free in internal pages of the underlying database.
bytes_free_in_leaf_pages	Number of bytes free in leaf pages of the underlying database.
bytes_free_in_duplicate_pages	Number of bytes free in duplicate pages of the underlying database.
bytes_free_in_overflow_pages	Number of bytes free in overflow pages of the underlying database.

Aggregate Storage Runtime Statistics

Statistics per Dimension

The following MaxL statement:

```
query database asoapp.asodb list aggregate_storage runtime_info;
```

Returns output which includes the following lines:

```
 parameter                                      value
+-----------------------------------------+-------------------
Dimension [Year] has [3] levels, bits used                    4
Dimension [Measures] has [1] levels, bits                     4
Dimension [Product] has [3] levels, bits u                    5
Dimension [Market] has [3] levels, bits us                    5
Dimension [Scenario] has [1] levels, bits                     2
...
```

For each dimension, the following statistics are shown:

- The name of the dimension.
- How many stored levels the dimension has, in the aggregate storage perspective. Not all levels are stored in aggregate storage databases; some are virtual levels.
- The number of bits being used in the key for the dimension.

Each cell in an aggregate storage database is stored as a key/value pair. The key length is 8 bytes or a multiple of 8 bytes; for example, 8, 16, 24.

Each key corresponds to a numeric value in the database. The number of bits each dimension uses in the dimensional key is shown in the value column for each dimension.

The number of bits used in each key may amount to less than the bytes needed for physical storage of the key. As an example where this knowledge might be useful, consider a case in which a key is using 65 bits. If you can reduce the key length by one bit to 64, then you can have the key length be 8 bytes instead of 16, an improvement which reduces the overall size of the database. Another use for these statistics might be to examine them to see how much you gain from removing any particular dimension.

Statistics for the Whole Database

The same MaxL statement used above also returns the following lines in its output:

```
 parameter                              value
+-------------------------------------+----------------------
...
Max. key length (bits)                                    20
Max. key length (bytes)                                    8
Number of input-level cells                                0
Number of incremental data slices                          0
Number of incremental input cells                          0
Number of aggregate views                                  0
Number of aggregate cells                                  0
Number of incremental aggregate cells                      0
Cost of querying incr. data (ratio to total cost)          0
Input-level data size (KB)                                 0
Aggregate data size (KB)                                   0
```

The whole-database statistics are described in the following table.

Table 3-25 Aggregate Storage Runtime Statistics MaxL Output

Column Name	Description
`Max. key length (bits)`	The sum of all the bits used by each dimension. For example, there are 20 bits in the key used for dimensions, and the first 4 are used by Year.
`Max. key length (bytes)`	How many bytes the key uses per cell.
`Number of input-level cells`	The number of existing level-0 cells in the database, including incremental slices.
`Number of incremental data slices`	The number of data slices resulting from incremental data loads.
`Number of incremental input cells`	The number of level-0 cells in the incremental data slices. To see the number of unique aggregate views, use the MaxL statement: `query database` *`appname.dbname`* `list existing_views;`

Table 3-25 (Cont.) Aggregate Storage Runtime Statistics MaxL Output

Column Name	Description
`Number of aggregate views`	The number of aggregate views in the database, including those automatically built on incremental slices.
`Number of aggregate cells`	The number of cells stored in the database's aggregate views.
`Number of incremental aggregate cells`	The number of cells stored in the incremental slices' aggregate views.
`Cost of querying incr. data (ratio to total cost)`	The average percentage of query time spent processing incremental data slices. This functionality is useful in deciding when slices should be merged together to improve query performance.
`Input-level data size (KB)`	The total disk space used by input-level data.
`Aggregate data size (KB)`	The total disk space occupied by aggregate cells.

For input-level and aggregate cells, the above statistics show:

1. Number of cells
2. Disk space occupied by those cells

Because Essbase uses compression, these statistics are useful because it is not always possible to derive disk size based on the number of cells.

Aggregate Storage Slice Information Output

The following MaxL statement:

```
query database "dmglex4"."basic" list aggregate_storage slice_info;
```

Returns the following output:

```
 transaction_id  slice_id  slice_tag  view_id  size_cells  size_kb
+--------------+---------+----------+--------+-----------+----------
              0         0          0        0          38         64
              3         1         66        0          21         32
              3         2         77        0          21         32
```

See Query Database.

Aggregate Storage Group ID Information Output

The following MaxL statement:

```
query database "dmglex4"."basic" list aggregate_storage group_id_info;
```

Returns the following output:

```
 group_id  transaction_id  state            time_last_used
time_expired              expired
+---------+--------------+----------------+------------------
+---------------------+------------
      1234               1 Allocation Done  Wed
Jul 20 17:39:57  Wed Jul 20 17:44:57           FALSE
```

See Query Database.

Aggregate Storage Uncommitted Transaction Information Output

The following MaxL statement:

```
query database "dmglex4"."basic" list aggregate_storage
uncommitted_transaction_info;
```

Returns the following output (columns are truncated):

```
 unc_trans unc_data_ unc_input unc_aggre unc_aggre unc_input unc_aggre
+---------+---------+---------+---------+---------+---------+---------
         0         0         0         0         0         0         0
```

See Query Database.

MaxL Use Cases

The following topics demonstrate some tasks that can be accomplished using MaxL.

- Creating an Aggregate Storage Sample Using MaxL
- Loading Data Using Buffers
- Using Aggregate Storage Data Load Buffers
- Forcing Deletion of Partitions
- Metadata Filtering
- Examples of Triggers

Creating an Aggregate Storage Sample Using MaxL

Related MaxL statements: create application, create database, create outline, alter database, import data, execute aggregate process,

The following sample MaxL script creates an aggregate storage application and database based on Sample.Basic.

```
login $1 $2;

spool on to 'maxl_log.txt';
```

```
create or replace application Sample2 using aggregate_storage
 comment 'aggregate storage version of Sample';

create database Sample2.Basic2
 comment 'aggregate storage version of Sample Basic';

create or replace outline on aggregate_storage database Sample2.Basic2
 as outline on database sample.basic;

alter database Sample2.Basic2 initialize load buffer with buffer_id 1;

import database Sample2.Basic2 data
 from server data_file '/catalog/users/catalogUser/Data.txt'
 to load_buffer with buffer_id 1
 on error abort;

import database Sample2.Basic2 data from load_buffer with buffer_id 1;

execute aggregate process on database Sample2.Basic2
 stopping when total_size exceeds 1.9;

spool off;

logout;
```

Loading Data Using Buffers

Related MaxL Statements

- Alter Database (Aggregate Storage)
- Query Database (Aggregate Storage)
- Import Data (Aggregate Storage)

If you use multiple Import Data (Aggregate Storage) statements to load data values to aggregate storage databases, you can significantly improve performance by loading values to a temporary data load buffer first, with a final write to storage after all data sources have been read.

While the data load buffer exists in memory, you cannot build aggregations or merge slices, as these operations are resource-intensive. You can, however, load data to other data load buffers, and perform queries and other operations on the database. There might be a brief wait for queries, until the full data set is committed to the database and aggregations are created.

The data load buffer exists in memory until the buffer contents are committed to the database or the application is restarted, at which time the buffer is destroyed. Even if the commit operation fails, the buffer is destroyed and the data is not loaded into the database.

Multiple data load buffers can exist on a single aggregate storage database. To save time, you can load data into multiple data load buffers at the same time by using separate MaxL Shell sessions. Although only one data load commit operation on a database can be active at any time, you can commit multiple data load buffers in the same commit operation, which is faster than committing buffers individually.

You can query the database for a list and description of the data load buffers that exist on an aggregate storage database. See Using Aggregate Storage Data Load Buffers.

Examples:

- Example: Load Multiple Data Sources into a Single Data Load Buffer
- Example: Perform Multiple Data Loads in Parallel

Example: Load Multiple Data Sources into a Single Data Load Buffer

Assume there are three data files that need to be imported. With aggregate storage databases, data loads are most efficient when all data files are loaded using one import operation. Therefore, load buffers are useful when loading more than one data file.

1. Use Alter Database (Aggregate Storage) to create a load buffer.

   ```
   alter database ASOsamp.Sample
   initialize load_buffer with buffer_id 1;
   ```

2. Load data into the buffer, using the Import Data (Aggregate Storage) statement.

   ```
   import database ASOsamp.Sample data
   from server data_file 'file_1'
   to load_buffer with buffer_id 1
   on error abort;

   import database ASOsamp.Sample data
   from server data_file 'file_2'
   to load_buffer with buffer_id 1
   on error abort;

   import database ASOsamp.Sample data
   from server data_file 'file_3'
   to load_buffer with buffer_id 1
   on error abort;
   ```

3. Move the data from the buffer into the database.

   ```
   import database ASOsamp.Sample data
   from load_buffer with buffer_id 1;
   ```

   The data-load buffer is implicitly destroyed.

4. Assume that in Step 2, after loading 'file_2' into the load buffer, you decided not to load the data. Because the data is in a buffer and not yet in the database, you would simply use Alter Database (Aggregate Storage) to destroy the buffer without moving the data to the database.

   ```
   alter database ASOsamp.Sample
   destroy load_buffer with buffer_id 1;
   ```

Example: Perform Multiple Data Loads in Parallel

1. In one MaxL Shell session, load data into a buffer with an ID of 1:

```
alter database ASOsamp.Sample
initialize load_buffer with buffer_id 1 resource_usage 0.5;


import database ASOsamp.Sample data
from data_file "dataload1.txt"
to load_buffer with buffer_id 1
on error abort;
```

2. Simultaneously, in another MaxL Shell session, load data into a buffer with an ID of 2:

```
alter database ASOsamp.Sample
initialize load_buffer with buffer_id 2 resource_usage 0.5;


import database ASOsamp.Sample data
from data_file "dataload2.txt"
to load_buffer with buffer_id 2
on error abort;
```

3. When the data is fully loaded into the data load buffers, use one MaxL statement to commit the contents of both buffers into the database by using a comma separated list of buffer IDs:

```
import database ASOsamp.Sample data
from load_buffer with buffer_id 1, 2;
```

Using Aggregate Storage Data Load Buffers

Related MaxL Statement:

Query Database (Aggregate Storage)

Use the following MaxL statement to get a list and description of the data load buffers that exist on an aggregate storage database.

```
query database appname.dbname list load_buffers;
```

This statement returns the following information about each existing data load buffer:

Table 3-26 List Load Buffers MaxL Output Columns

Field	Description
buffer_id	ID of a data load buffer (a number between 1 and 4294967296).

Table 3-26 (Cont.) List Load Buffers MaxL Output Columns

Field	Description
internal	A Boolean that specifies whether the data load buffer was created internally by Essbase (TRUE) or by a user (FALSE).
active	A Boolean that specifies whether the data load buffer is currently in use by a data load operation.
resource_usage	The percentage (a number between .01 and 1.0 inclusive) of the aggregate storage cache that the data load buffer is allowed to use.
aggregation method	One of the methods used to combine multiple values for the same cell within the buffer: • AGGREGATE_SUM: Add values when the buffer contains multiple values for the same cell. • AGGREGATE_USE_LAST: Combine duplicate cells by using the value of the cell that was loaded last into the load buffer.
ignore_missings	A Boolean that specifies whether to ignore #MI values in the incoming data stream.
ignore_zeros	A Boolean that specifies whether to ignore zeros in the incoming data stream.

Forcing Deletion of Partitions

The **force** keyword used at the end of the drop partition statement specifies that the source half of a partition definition should be dropped regardless of whether the target half is missing or invalid.

For example, in the following session, assume there is a partition definition between app1.source and app2.target, but the app2.target database has been dropped. An ordinary attempt to drop the partition definition fails:

```
MAXL> drop transparent partition app1.source to app2.target;

     OK/INFO - 1053012 - Object source is locked by user system.
     OK/INFO - 1051034 - Logging in user System.
     OK/INFO - 1051035 - Last login on Friday, January 10, 2005 2:28:09
PM.
       ERROR - 1051032 - Database target does not exist.
     OK/INFO - 1053013 - Object source unlocked by user system.
     OK/INFO - 1051037 - Logging out user system, active for 0 minutes.
```

In the second attempt, the **force** keyword allows the invalid source partition to be dropped:

```
MAXL> drop transparent partition app1.source to app2.target force;

   OK/INFO - 1053012 - Object source is locked by user system.
```

```
   OK/INFO - 1051034 - Logging in user System.
   OK/INFO - 1051035 - Last login on Friday, January 10, 2005 2:31:50
PM.
     ERROR - 1051032 - Database target does not exist.
   OK/INFO - 1051037 - Logging out user system, active for 0 minutes.
   OK/INFO - 1053013 - Object source unlocked by user system.
   OK/INFO - 1241125 - Partition dropped.
```

> **Note:**
>
> The force keyword only works to drop a partition definition when the source half of the partition definition remains valid. In other words, if the source database is deleted, the partition cannot be dropped from the dangling target.

Metadata Filtering

Related MaxL statements: create filter, alter filter.

Metadata filtering provides an additional layer of security in addition to data filtering. With metadata filtering, an administrator can remove outline members from a user's view, providing access only to those members that are of interest to the user.

When a filter is used to apply MetaRead permission on a member,

1. Data for all ancestors of that member are hidden from the filter user's view.
2. Data *and* metadata (member names) for all siblings of that member are hidden from the filter user's view.

Example

The following report script for Sample.Basic:

```
//Meta02.rep

<COLUMN (Year, Product)
<CHILDREN Cola

<ROW (Market)
<ICHILDREN West
!
```

under normal unfiltered conditions returns

```
      Year 100-10 Measures Scenario
California          3,498
Oregon                159
Washington            679
Utah                  275
Nevada                (18)
  West              4,593
```

But with the following filter granted to an otherwise read-access user,

```
create or replace filter sample.basic.meta02
 meta_read on '"California","Oregon"'
;
```

the report script then returns:

```
      Year 100-10 Measures Scenario
California          3,498
Oregon                159
  West           #Missing
```

In summary, MetaRead permission on California and Oregon means that:

1. The affected user can see no data for ancestors of California and Oregon members. West data shows only #Missing (or #NoAccess, in a grid client interface).
2. The affected user can see no sibling metadata (or data) for siblings of California and Oregon. In other words, the user sees only the western states for which the filter gives MetaRead permission.

Overlapping Metadata Filter Definitions

You should define a MetaRead filter using multiple rows only when the affected member set in any given row (the metaread members and their ancestors) has no overlap with MetaRead members in other rows. Oracle recommends that you specify one dimension per row in filters that contain MetaRead on multiple rows. However, as long as there is no overlap between the ancestors and MetaRead members, it is still valid to specify different member sets of one dimension into multiple MetaRead rows.

For example, in Sample.Basic, the following filter definition has overlap conflicts:

Table 3-27 Sample Filter with Overlap Conflicts

Access	Member Specification
MetaRead	California
MetaRead	West

In the first row, applying MetaRead to California has the effect of allowing access to California but blocking access to its ancestors. Therefore, the MetaRead access to West is ignored; users who are assigned this filter will have no access to West.

If you wish to assign MetaRead access to West as well as California, then the appropriate method is to combine them into one row:

Table 3-28 Sample Filter with No Overlap Conflicts

Access	Member Specification
MetaRead	California,West

Examples of Triggers

Related MaxL statements: alter trigger, create trigger, display trigger, drop trigger.

The following examples are based on the Sample.Basic database.

> **Note:**
>
> You cannot define a trigger that requires data from Dynamic Calc members or members from another partition.

Example 1: Tracking Sales for January

Example 1 tracks the Actual, Sales value for the following month, product, and region:

- January (Year dimension member Jan)
- Colas (Product dimension member 100)
- In the Eastern region (Market dimension member East)

When the current member being calculated is Jan, and when the Actual, Sales value of Colas for January exceeds 20, the example logs an entry in the file Trigger_jan_Sales.

```
create or replace trigger Sample.Basic.Trigger_Jan_20
Where
 {(Jan,Sales,[100],East,Actual)}
When
 Jan > 20 AND Is(Year.CurrentMember, Jan)
then spool Trigger_Jan_20
end;
```

Example 2: Tracking Sales for Quarter 1

Example 2 tracks the Actual, Sales value for the following months, product, and region:

- January, February, March (The children of Year dimension member Qtr1)
- Colas (Product dimension member 100)
- In the Eastern region (Market dimension member East)

When the current member being calculated is Jan, Feb or Mar, and when the Actual, Sales value of Colas for any of the the months January, February, or March exceeds 20, the example logs an entry in the file Trigger_Jan_Sales_20, Trigger_Feb_Sales_20, or Trigger_Mar_Sales_20.

```
create or replace trigger Sample.Basic.Trigger_Qtr1_Sales
Where
Crossjoin(
 {Qtr1.children},
 {([Measures].[Sales], [Product].[100], [Market].[East], [Scenario].
[Actual])}
```

```
 )
When
 Year.Jan > 20 and is(Year.currentmember, Jan)
then spool Trigger_Jan_Sales_20
When
 Year.Feb > 20 and is(Year.currentmember, Feb)
then spool Trigger_Feb_Sales_20
When
 Year.Mar > 20 and is(Year.currentmember, Mar)
then spool Trigger_Mar_Sales_20
end;
```

Example 3: Tracking Inventory Level

Example 3 tracks the inventory level for the following product, region, and months:

- Colas (product 100)
- In the eastern region (market East)
- For January, February, and March (the children of Qtr1)

If the inventory of Colas in the eastern region falls below 500,000, the example trigger sends an email to recipient@example.com.

```
create or replace trigger Sample.Basic.Inventory_east
where CrossJoin(
 {[Qtr1].children},
 {([East],[100],[Ending Inventory])}
)
when [Ending Inventory] < 500000 then
mail ([smtp_server.example.com],[sender@example.com],
        [recipient@example.com],
[Subject of E-Mail])
end;
```