

Oracle® Essbase

Configuration Reference for Oracle Essbase



F17648-12
February 2025



Oracle Essbase Configuration Reference for Oracle Essbase,

F17648-12

Copyright © 2019, 2025, Oracle and/or its affiliates.

Primary Author: Essbase Information Development Team

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

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

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

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

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

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

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

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

Contents

1 Configuration Reference Overview

Set Application-Level Configuration Properties	1-1
Set Server-Level Configuration Properties	1-4
Set Provider Services Configuration Properties	1-5

2 Essbase Configuration Settings

Config Settings List	2-1
AGENTLEASEEXPIRATIONTIME	2-5
AGENTLEASERENEWALTIME	2-6
AGENTPORT	2-6
AGENTSECUREPORT	2-7
AGENTTHREADS	2-7
AGGRESSIVEBLKOPTIMIZATION	2-8
APPMAXLOGFILESIZE	2-9
ASOBUFFERCOMMITWAIT	2-9
ASOCACHECONCURRENTCONSUMINGTHREADS	2-10
ASODEFAULTCACHESIZE	2-11
ASODYNAMICAGGINBSO	2-12
ASODYNAMICAGGINBSOFOLDERPATH	2-13
ASODYNHIERASAGG	2-14
ASOLOADBUFFERWAIT	2-14
ASOSAMPLESIZEPERCENT	2-15
AUDITTRAIL	2-17
AUTHENTICATIONMODULE	2-18
AUTOMERGE	2-18
AUTOMERGEMAXSLICENUMBER	2-19
CALCCACHE	2-20
CALCCACHEDEFAULT	2-21
CALCCACHEHIGH	2-22
CALCCACHELOW	2-23
CALCLIMITFORMULARECURSION	2-25
CALCMODE	2-26
CALCNOTICE	2-27

CALCOPTFRMLBOTTOMUP	2-28
CALCPARALLEL	2-29
CALCREUSEDYNALCBLOCKS	2-30
CALCTASKDIMS	2-31
CALCTRACE	2-32
CCTRACK	2-33
CLIENTPREFERREDMODE	2-34
CRASHDUMP	2-34
CRASHDUMPLOCATION	2-35
CUSTOMCALCANDALLOCTHRUINSERT	2-36
DATACACHE SIZE	2-36
DATAERRORLIMIT	2-37
DEFAULTVIEWBUILD	2-37
DEFAULTVIEWBUILDSIZE	2-38
DELIMITEDMSG	2-39
DELIMITER	2-40
DIMBUILDERRORLIMIT	2-40
DIMBUILDSTATSINTERVAL	2-41
DISABLEREPLMISSINGDATA	2-41
DLSINGLETHREADPERSTAGE	2-42
DLTHREADSPREPARE	2-44
DLTHREADSWRITE	2-45
DYNALCCACHEBLKRELEASE	2-47
DYNALCCACHEBLKTIMEOUT	2-48
DYNALCCACHECOMPRBLKBUFSIZE	2-49
DYNALCCACHEMAXSIZE	2-51
DYNALCCACHEONLY	2-52
DYNALCCACHEWAITFORBLK	2-53
ENABLE_DIAG_TRANSPARENT_PARTITION	2-55
ENABLECLEARMODE	2-56
ENABLERTSVLOGGING	2-57
ENABLESECUREMODE	2-57
ESSBASESERVERHOSTNAME	2-58
ESTIMATEDHASHSIZE	2-59
EXCEPTIONLOGOVERWRITE	2-59
EXCLUSIVECALC	2-60
EXPORTTHREADS	2-61
FAILOVERMODE	2-62
FEDERATEDAVCALC	2-62
FILEGOVPATH	2-63
FILELOCKINGMODE	2-65
FORCEALLDENSECALCON2PASSACCOUNTS	2-65

FORCEGRIDEXPANSION	2-66
FORCESHUTDOWNINTERVAL	2-67
GRIDEXPANSION	2-67
GRIDEXPANSIONMESSAGES	2-68
GRIDSUPPRESSINVALID	2-69
HYBRIDBSOINCALCSCRIPT	2-69
IDCS_REQ_LIMIT_PER_SEC	2-71
IDLETIMEMERGE	2-71
IGNORECONSTANTS	2-72
IMPLIED_SHARE_ON_CREATE	2-73
INDEXCACHESIZE	2-74
INPLACEDATAWRITE	2-75
INPLACEDATAWRITEMARGINPERCENT	2-76
JAVAMAXSMARTLISTSPEROUTLINE	2-77
JVMMODULELOCATION	2-77
LOCKTIMEOUT	2-78
LOGMESSAGELEVEL	2-79
LONGCALCTIMETHRESHOLD	2-79
LONGQUERYTIMETHRESHOLD	2-80
LONGREQTIMETHRESHOLD	2-82
LROONSHAREDMBR	2-83
MAXDATE	2-84
MAXERRORMBRVERIFYREPORT	2-85
MAXFORMULACACHESIZE	2-85
MAXLOGINS	2-87
MAXNUMBEROFACTIVEDB	2-87
MAX_REQUEST_GRID_SIZE	2-88
MAX_RESPONSE_GRID_SIZE	2-89
MAX_SIZE_PER_FETCH	2-90
MDXINSERTBUFFERAGGMETHOD	2-90
MDXINSERTREQUESTTIMEOUT	2-92
MDXLIMITFORMULARECURSION	2-92
MDXQRYGOVCOUNT	2-93
MEMORYMAPPEDDATA	2-94
METADATABASEDAGGVIEWSBUILD	2-95
MULTIPLEBITMAPMEMCHECK	2-96
NETBINDRETRYDELAY	2-96
NETDELAY	2-97
NETRETRYCOUNT	2-98
NETSSLHANDSHAKETIMEOUT	2-98
NETTCPCONNECTRETRYCOUNT	2-99
NUMBLOCKSTOEXTEND	2-100

NUMERICPRECISION	2-101
ODBCERRORLOGOFF	2-102
ORACLEHARDWAREACCELERATION	2-102
OUTLINECHANGELOG	2-103
OUTLINECHANGELOGFILESIZE	2-103
PARCALCMULTIPLEBITMAPMEMOPT	2-104
PIPEBUFFERSIZE	2-105
PREFIXID	2-105
QRYGOVEXECBLK	2-106
QRYGOVEXECTIME	2-107
QUERYBOTTOMUP	2-109
QUERYRESULTLIMIT	2-109
QUERYTRACE	2-110
QUERYTRACETHRESHOLD	2-112
REJECTEDRECORDSLIMIT	2-112
RENEGADELOG	2-113
RENEGADELOGLIMIT	2-114
REPLAYSECURITYOPTION	2-115
REPLICATIONASSUMEIDENTICALOUTLINE	2-115
RESTRUCTURETHREADS	2-116
RTDEPCALCOPTIMIZE	2-117
SERVERPORTBEGIN	2-118
SERVERPORTEND	2-119
SERVERTHREADS	2-120
SPLITARCHIVEFILE	2-121
SQLFETCHERRORPOPOP	2-122
SSANCESTORONTOP	2-122
SSAUDIT	2-123
SSAUDITR	2-125
SSBULKGRIDPROCESSING	2-126
SSINVALIDTEXTDETECTION	2-127
SSLCIPHERSUITES	2-127
SSLOGUNKNOWN	2-128
SSMEMBERIDPROCESSING	2-129
SSOPTIMIZEDGRIDPROCESSING	2-130
SSPROCROWLIMIT	2-131
SUPNA	2-132
SVRIDLETIME	2-132
TARGETASOOPT	2-133
TARGETTIMESERIESOPT	2-134
TIMINGMESSAGES	2-134
TRACE_REPORT	2-135

TRANSACTIONLOGDATALOADARCHIVE	2-138
TRANSACTIONLOGLOCATION	2-139
TRIGMAXMEMSIZE	2-141
UPDATECALC	2-142
WALLETPATH	2-143
WORKERTHREADS	2-144
Configuration Settings Categorical List	2-146
Backup and Recovery Configuration Settings	2-147
Calculation Configuration Settings	2-147
Data Import and Export Configuration Settings	2-148
Failover Configuration Settings	2-148
Logging and Error Handling Configuration Settings	2-149
Memory Management Configuration Settings	2-149
Miscellaneous Configuration Settings	2-150
Oracle Exalytics In-Memory Machine Configuration Settings	2-150
Partitioning Configuration Settings	2-150
Ports and Connections Configuration Settings	2-150
Query Management Configuration Settings	2-151
Aggregate Storage and Block Storage Settings Comparison	2-151
Block Storage and Aggregate Storage Configuration Settings	2-152
Aggregate Storage Configuration Settings	2-152
Block Storage Configuration Settings	2-153
Sample Configurations for Threaded Operations	2-153

3 Provider Services Configuration

essbase.jobs.maxCount	3-1
olap.server.netRetryCount	3-1
olap.server.netSocketTimeOut	3-2
olap.server.netSocketTryInfinite	3-2
system.cluster.monitorInterval	3-2

4 Query Logging Configuration

Enable Query Logging	4-1
Query Log Settings File Syntax	4-3
Query Logging Sample File	4-6
Query Logging Sample Output	4-7

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

Configuration Reference Overview

You can use the Oracle Essbase configuration settings to customize the functionality of Essbase.

This reference describes available configuration options for Essbase. This reference is intended for advanced users who need detailed information and examples.

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.

Set Application-Level Configuration Properties

If you have the Service Administrator role, or the Power User role for applications that you created, you can customize Oracle Essbase using application-level configuration properties. Application-level configuration properties apply to all cubes in the application.

One way to specify configuration properties of an application is to do it prior to building the application and cube, using the application workbook. To see an example, go to Files in the Essbase web interface, and download the application workbook `Sample_Basic.xlsx`. It is located in the gallery, in the Demo Samples section (under Block Storage). In this application workbook, go to the Cube.Settings worksheet. Under Application Configuration, the `DATACACHE`SIZE property is set to 3M, and the `INDEXCACHE`SIZE property is set to 1M.

38	Application Configuration	
39		
40	DATACACHE SIZE	3M
41	INDEXCACHE SIZE	1M
42		
43		

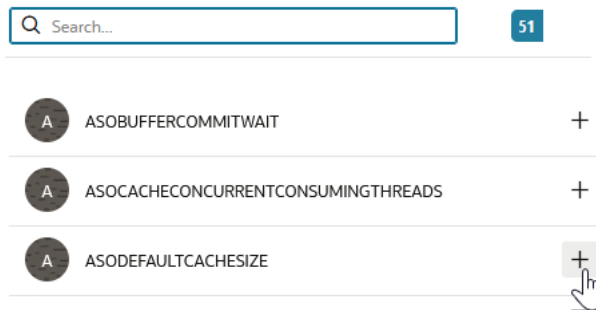
Navigation: Essbase.Cube | **Cube.Settings** | Cube.Generations | D

The following steps tell you how to configure an application that is already deployed, by adding properties and their corresponding values in the Essbase web interface.

- Redwood
- Classic

Redwood

1. On the Applications page, select the application you want to configure.
2. Click **Configuration**.
3. To add a property, click **Add**.
Scroll through the list or search for a property.
4. Click **+** to add the property to the list.



5. Click **x** to close the search tool.



6. In the **Value** column, double click to enter a value.



7. When you're finished making changes, click **Apply and Restart**.

Note:

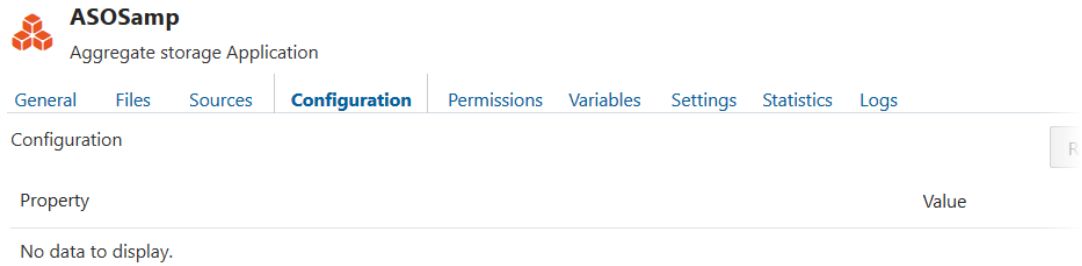
If the application is not started, you are given the option to "Apply" rather than "Apply and Restart." Changes will be applied the next time the application is restarted.

- Wait for the confirmation message.

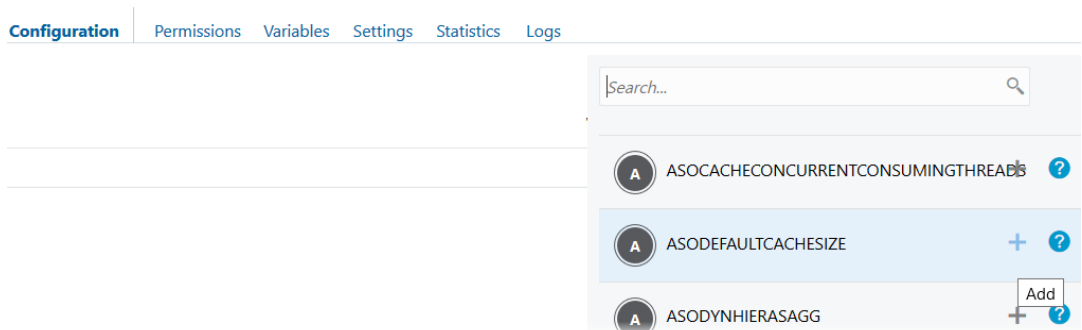
Configuration settings were stored successfully and will be applied when the application is restarted

Classic

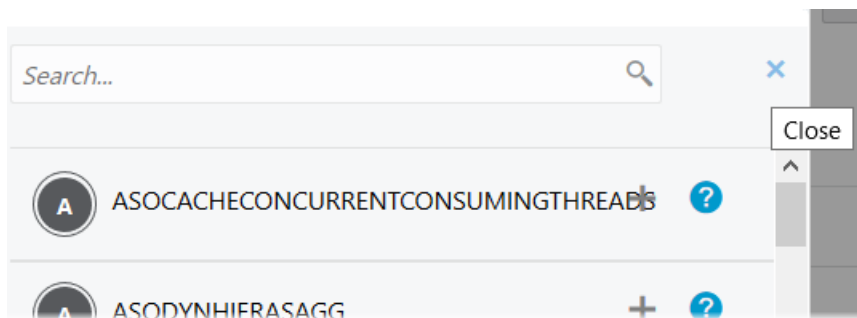
- On the Applications page, select the application you want to configure.
- From the **Actions** menu to the right of the application, click **Inspect**, then click **Configuration**.



- To add a property, click **+**. Scroll through the list or search for a property.
- Click **+** to add the property to the list.



- Click **X** to close the search tool.




- In the **Value** column, double click to enter a value.

The screenshot shows the ASOSamp Configuration interface. At the top, there is a logo for ASOSamp (Aggregate storage Application) and a 'Close' button. Below the logo is a navigation bar with tabs for General, Files, Sources, Configuration (selected), Permissions, Variables, Settings, Statistics, and Logs. Under the Configuration tab, there is a 'Configuration' section with 'Reset' and 'Apply and Restart' buttons. A table below shows the configuration properties:

Property	Value
ASODEFAULTCACHESIZE	200

7. When you're finished making changes, click **Apply and Restart**.
8. Wait for the confirmation message.

 **Configurations are applied successfully and the application is restarted**

For syntax and information about each of the application configuration properties you can use, see Config Settings List. You do not need to use the optional [`<appname>`] syntax when adding properties to the application configuration.

Oracle does not recommend that you modify `essbase.cfg` on the Essbase file system. This configuration is automatically set.

Set Server-Level Configuration Properties

If you have the Service Administrator role, you can configure Oracle Essbase server properties using `essbase.cfg` on the server domain. Server configuration properties apply to all applications and cubes on your Essbase instance.

In most cases, Oracle does not recommend that you modify `essbase.cfg` on the Essbase server machine. Most configurations are best to set at the [application level](#). However, in some cases, you may need to edit the server configuration.

For Essbase 21c, `essbase.cfg` is located in `<Domain Root>/<Domain Name>/config/fmwconfig/essconfig/essbase`. If you do not know where that is in your environment, refer to Environment Locations in the Essbase Platform for an explanation.

The following steps tell you how to configure server-level properties and their corresponding values.

1. Access Essbase server machine.
2. Navigate to `essbase.cfg`. Example: `<Domain Root>/<Domain Name>/config/fmwconfig/essconfig/essbase/essbase.cfg`
3. Open `essbase.cfg` in a text editor.
4. Enter the property name and desired value. For example, `AGENTPORT 1425`.
5. Save `essbase.cfg`.

The configuration changes will take effect next time you stop and restart the Essbase server, as described in Start, Stop, and Check Servers.

For syntax and information about each of the configuration properties you can use, see [Config Settings List](#).

Set Provider Services Configuration Properties

If you have the Service Administrator role, you can customize network-related settings for Oracle Essbase using the Provider Services configuration properties.

To set the values for Provider Services configuration properties,

1. Log in to the Essbase web interface as a Service Administrator.
2. Click **Console**.
3. In the Console, click **Configuration**.
4. On the Provider Services tab, click **Add** to add a new property and set its value. If the property you want to configure is already listed, double-click the **Value** field to edit the value.
5. When you are finished editing properties, click **Save**.

2

Essbase Configuration Settings

You can use a variety of configuration settings to customize the behavior of your Essbase applications, or to change server defaults.

- [Config Settings List](#)
- [Configuration Settings Categorical List](#)
- [Aggregate Storage and Block Storage Settings Comparison](#)
- [Sample Configurations for Threaded Operations](#)

To change application configuration properties, see [Set Application-Level Configuration Properties](#).

To change server-level configuration properties, see [Set Server-Level Configuration Properties](#).

Config Settings List

- [AGENTLEASEEXPIRATIONTIME](#)
- [AGENTLEASERENEWALTIME](#)
- [AGENTPORT](#)
- [AGENTSECUREPORT](#)
- [AGENTTHREADS](#)
- [AGGRESSIVEBLKOPTIMIZATION](#)
- [APPMAXLOGFILESIZE](#)
- [ASOBUFFERCOMMITWAIT](#)
- [ASOCACHECONCURRENTCONSUMINGTHREADS](#)
- [ASODEFAULTCACHESIZE](#)
- [ASODYNAMICAGGINBSO](#)
- [ASODYNAMICAGGINBSOFOLDERPATH](#)
- [ASODYNHIERASAGG](#)
- [ASOLOADBUFFERWAIT](#)
- [ASOSAMPLESIZEPERCENT](#)
- [AUDITTRAIL](#)
- [AUTHENTICATIONMODULE](#)
- [AUTOMERGE](#)
- [AUTOMERGEMAXSLICENUMBER](#)
- [CALCCACHE](#)
- [CALCCACHEDEFAULT](#)
- [CALCCACHEHIGH](#)

- CALCCACHELOW
- CALCLIMITFORMULARECURSION
- CALCMODE
- CALCNOTICE
- CALCOPTFRMLBOTTOMUP
- CALCPARALLEL
- CALCREUSEDYNALCBLOCKS
- CALCTASKDIMS
- CALCTRACE
- CCTRACK
- CLIENTPREFERREDMODE
- CRASHDUMP
- CRASHDUMPLOCATION
- CUSTOMCALCANDALLOCTHRUINSERT
- DATACACHESIZE
- DATAERRORLIMIT
- DEFAULTVIEWBUILD
- DEFAULTVIEWBUILDSIZE
- DELIMITEDMSG
- DELIMITER
- DIMBUILDERRORLIMIT
- DIMBUILDSTATSINTERVAL
- DISABLEREPLMISSINGDATA
- DLSINGLETHREADPERSTAGE
- DLTHREADSPREPARE
- DLTHREADSWRITE
- DYNALCCACHEBLKRELEASE
- DYNALCCACHEBLKTIMEOUT
- DYNALCCACHECOMPRBLKBUFSIZE
- DYNALCCACHEMAXSIZE
- DYNALCCACHEONLY
- DYNALCCACHEWAITFORBLK
- ENABLE_DIAG_TRANSPARENT_PARTITION
- ENABLECLEARMODE
- ENABLERTSVLOGGING
- ENABLESECUREMODE
- ESSBASESERVERHOSTNAME
- ESTIMATEDHASHSIZE

- EXCEPTIONLOGOVERWRITE
- EXCLUSIVECALC
- EXPORTTHREADS
- FAILOVERMODE
- FEDERATEDAVCALC
- FILEGOVPATH
- FILELOCKINGMODE
- FORCEALLDENSECALCON2PASSACCOUNTS
- FORCEGRIDEXPANSION
- FORCESHUTDOWNINTERVAL
- GRIDEXPANSION
- GRIDEXPANSIONMESSAGES
- GRIDSUPPRESSINVALID
- HYBRIDBSOINCALCSCRIPT
- IDCS_REQ_LIMIT_PER_SEC
- IDLETIMEMERGE
- IGNORECONSTANTS
- IMPLIED_SHARE_ON_CREATE
- INDEXCACHESIZE
- INPLACEDATAWRITE
- INPLACEDATAWRITEMARGINPERCENT
- JAVAMAXSMARTLISTSPEROUTLINE
- JVMMODULELOCATION
- LOCKTIMEOUT
- LOGMESSAGELEVEL
- LONGCALCTIMETHRESHOLD
- LONGQUERYTIMETHRESHOLD
- LONGREQTIMETHRESHOLD
- LROONSHAREDMBR
- MAXDATE
- MAXERRORMBRVERIFYREPORT
- MAXFORMULACACHESIZE
- MAXLOGINS
- MAXNUMBEROFACTIVEDB
- MAX_REQUEST_GRID_SIZE
- MAX_RESPONSE_GRID_SIZE
- MAX_SIZE_PER_FETCH
- MDXINSERTBUFFERAGGMETHOD

- MDXINSERTREQUESTTIMEOUT
- MDXLIMITFORMULARECURSION
- MDXQRYGOVCOUNT
- MEMORYMAPPEDDATA
- METADATABASEDAGGVIEWSBUILD
- MULTIPLEBITMAPMEMCHECK
- NETBINDRETRYDELAY
- NETDELAY
- NETRETRYCOUNT
- NETSSLHANDSHAKETIMEOUT
- NETTCPCONNECTRETRYCOUNT
- NUMBLOCKSTOEXTEND
- NUMERICPRECISION
- ODBCERRORLOGOFF
- ORACLEHARDWAREACCELERATION
- OUTLINECHANGELOG
- OUTLINECHANGELOGFILESIZE
- PARCALCMULTIPLEBITMAPMEMOPT
- PIPEBUFFERSIZE
- QRYGOVEXECBLK
- QRYGOVEXECTIME
- QUERYBOTTOMUP
- QUERYRESULTLIMIT
- QUERYTRACE
- QUERYTRACETHRESHOLD
- REJECTEDRECORDSLIMIT
- RENEGADELOG
- RENEGADELOGLIMIT
- REPLAYSECURITYOPTION
- REPLICATIONASSUMEIDENTICALOUTLINE
- RESTRUCTURETHREADS
- RTDEPCALCOPTIMIZE
- SERVERPORTBEGIN
- SERVERPORTEND
- SERVERTHREADS
- SPLITARCHIVEFILE
- SQLFETCHERRORPOPUP
- SSANCESTORONTOP

- [SSAUDIT](#)
- [SSAUDITR](#)
- [SSBULKGRIDPROCESSING](#)
- [SSINVALIDTEXTDETECTION](#)
- [SSLCIPHERSUITES](#)
- [SSLOGUNKNOWN](#)
- [SSMEMBERIDPROCESSING](#)
- [SSOPTIMIZEDGRIDPROCESSING](#)
- [SSPROCROWLIMIT](#)
- [SUPNA](#)
- [SVRIDLETIME](#)
- [TARGETASOOPT](#)
- [TARGETTIMESERIESOPT](#)
- [TIMINGMESSAGES](#)
- [TRACE_REPORT](#)
- [TRANSACTIONLOGDATALOADARCHIVE](#)
- [TRANSACTIONLOGLOCATION](#)
- [TRIGMAXMEMSIZE](#)
- [UPDATECALC](#)
- [WALLETPATH](#)
- [WORKERTHREADS](#)

AGENTLEASEEXPIRATIONTIME

The AGENTLEASEEXPIRATIONTIME configuration setting specifies the maximum amount of time that Essbase Agent can own a lease before the lease is terminated.

AGENTLEASEEXPIRATIONTIME is configurable only for Independent deployment, and should be set in the Essbase Server-level `essbase.cfg` file.

Syntax

```
AGENTLEASEEXPIRATIONTIME n
```

Where *n* is an integer specifying the number of seconds before a lease expires. The default value is 90 seconds.

Example

```
AGENTLEASEEXPIRATIONTIME 100
```

See Also

[AGENTLEASERENEWALTIME](#)

AGENTLEASERENEWALTIME

The AGENTLEASERENEWALTIME configuration setting specifies the time interval, in seconds, after which Essbase Agent attempts to renew a lease. The value must be less than the value set for AGENTLEASEEXPIRATIONTIME.

AGENTLEASERENEWALTIME configuration is configurable only for Independent deployment, and should be set in the Essbase Server-level essbase.cfg file.

Syntax

```
AGENTLEASERENEWALTIME n
```

Where *n* is an integer specifying the number of seconds to reestablish ownership after a lease expires. The default value is 30 seconds.

Example

```
AGENTLEASERENEWALTIME 40
```

See Also

[AGENTLEASEEXPIRATIONTIME](#)

AGENTPORT

The AGENTPORT configuration setting specifies the port that the Essbase Agent uses. You may wish to change the default AGENTPORT if the server port 1423 is inappropriate for your site.

Syntax

```
AGENTPORT n
```

Where *n* specifies the port number for the Essbase Agent. The port number you select should not be in use by any other process. The default value is 1423.

Example

```
AGENTPORT 1478  
SERVERPORTBEGIN 30768  
SERVERPORTEND 31768
```

The above configuration example produces these results:

- AGENTPORT sets the port that the Agent will use at 1478.
- SERVERPORTBEGIN sets the value that the first application server process (ESSSVR) will try to use for a port at 30768.
- SERVERPORTEND sets the highest port number value this Essbase instance can use for application server processes.

See Also[AGENTSECUREPORT](#)[SERVERPORTBEGIN](#)[SERVERPORTEND](#)

AGENTSECUREPORT

The AGENTSECUREPORT configuration setting specifies the port that the Essbase Agent uses for secure communication when in secure connection mode with Transport Layer Security (TLS).

Syntax

```
AGENTSECUREPORT n
```

where *n* specifies the secure port number for the Essbase Agent. The port number you select should not be in use by any other process. The default value is 6423.

Example

```
AGENTSECUREPORT 6425
```

See Also[ENABLESECUREMODE](#)[AGENTPORT](#)

AGENTTHREADS

The AGENTTHREADS configuration setting specifies the maximum number of threads that Essbase can spawn for operations such as logging in and out of Essbase Server, and starting and stopping an application.

Syntax

```
AGENTTHREADS n
```

where *n* specifies the number of threads that Essbase can spawn. The value of *n* can be a number from 5 to 500, inclusive.

The default value is 50 threads.

Notes

- Although the maximum AGENTTHREADS value you can set is 500, the maximum number of threads an operating system can handle might be much lower. Before specifying a value greater than the default value, check with your system administrator, as higher values can significantly consume system resources.
- If you specify an AGENTTHREADS value that is less than 5, over the maximum, or a decimal value, Essbase overrides the value with a closely approximate value of its own.

- One Agent thread is required for each initial connection to an Essbase application and database.
- The AGENTTHREADS configuration setting does not apply to Essbase Java Agent, which uses the WebLogic Server thread pool configuration for the total number of threads that can be spawned at the server and domain levels. The WebLogic Server total thread count is specified in the `config.xml` file. If the value of AGENTTHREADS is less than the value of the WebLogic Server total thread count, Essbase uses the value specified in AGENTTHREADS; if the value of AGENTTHREADS is more than the WebLogic Server total thread count, Essbase uses the value specified in the `config.xml` file.

Example

```
AGENTTHREADS 100
```

Sets the maximum number of threads that Essbase can spawn to 100, assuming that the total thread count specified in the `config.xml` file is 100 or more.

AGGRESSIVEBLKOPTIMIZATION

The AGGRESSIVEBLKOPTIMIZATION configuration setting improves batch calculation time for block storage (BSO) cubes. Do not enable it if your Essbase outline includes stored members which are dependent on dense Dynamic Calc members.

When there are dense Dynamic Calc members in the outline, a batch calculation with formulas uses blocks that contain data cells for all dense Dynamic Calc members. Setting AGGRESSIVEBLKOPTIMIZATION to TRUE, which should be done only if stored members are not dependent on any dense Dynamic Calc members, makes batch calculation work on kernel blocks (smaller blocks) directly, which may improve performance.

Use this setting only if there is no formula dependency on dense Dynamic Calc members; otherwise, the calculation may produce incorrect results.

AGGRESSIVEBLKOPTIMIZATION configuration does not apply to aggregate storage cubes.

Syntax

```
AGGRESSIVEBLKOPTIMIZATION TRUE | FALSE
```

- TRUE—Essbase uses batch calculation on smaller kernel blocks. Use only if stored members are not dependent on any dense Dynamic Calc members.
- FALSE—Essbase does not use batch calculation on smaller kernel blocks. The default value is FALSE.

Example

```
AGGRESSIVEBLKOPTIMIZATION TRUE
```

Improves calculation performance for outlines in which there is no formula dependency on dense Dynamic Calc members.

APPMAXLOGFILESIZE

The APPMAXLOGFILESIZE configuration setting enables you to specify the maximum size of Essbase application log files.

Syntax

```
APPMAXLOGFILESIZE n
```

n—The file size, in bytes.

- The minimum application log file size is 1 MB (1048576 bytes). If you specify a value less than the minimum, it is reset to 1 MB.
- The maximum application log file size is 2 GB (2147482623 bytes). If you specify a value greater than the maximum, it is reset to 2 GB.
- If no value is specified, the default value of 2 GB (2147482623 bytes) is used.

Description

Essbase writes to the application log named `<appname>.log`. When maximum log file size is reached, the file is renamed with incremental numbers at the end: `<appname>.log.<n>` (for example, `Sample.log.0`, `Sample.log.1`, and so on), and a new `<appname>.log` file is created.

Example

```
APPMAXLOGFILESIZE 1500000
```

Sets the maximum application log file size to 1500000 bytes.

ASOBUFFERCOMMITWAIT

The Essbase configuration setting ASOBUFFERCOMMITWAIT defines what should happen if someone attempts to commit an aggregate storage (ASO) data load buffer at the same time as another buffer is being committed. If set to TRUE, concurrent commit attempts should wait. If set to FALSE, they should fail.

Syntax

```
ASOBUFFERCOMMITWAIT [appname] TRUE | FALSE
```

- TRUE—The next concurrent ASO load buffer commit attempt waits until the previous operation completes.
- FALSE—The next concurrent ASO load buffer commit attempt fails and returns an error. This is the default.

Description

When you load data to an ASO cube, you commit the contents of a load buffer to the cube. Creating an ASO load buffer and loading data into the buffer are not exclusive operations (they

can run concurrently with other operations). However, committing the data in the load buffer to the cube is an exclusive operation.

Example

```
ASOBUFFERCOMMITWAIT TRUE
```

See Also

[IDLETIMEMERGE](#)

[ASOCACHECONCURRENTCONSUMINGTHREADS](#)

ASOCACHECONCURRENTCONSUMINGTHREADS

The Essbase configuration setting `ASOCACHECONCURRENTCONSUMINGTHREADS` specifies the maximum number of heavily loaded buffers that can be defined simultaneously for a particular aggregate storage (ASO) application. A heavily loaded buffer is considered to be one that has a specified *resource_usage* value of 1, which is the maximum allowed.

When you load data into ASO cubes, Essbase uses the aggregate storage cache for sorting data. You can control the amount of cache a data load buffer can use by specifying the resource allocation.

Syntax

```
ASOCACHECONCURRENTCONSUMINGTHREADS [appname] n
```

where *n* specifies the number of heavily loaded buffers that the application can create concurrently. The value of *n* can be 1 to 1000, inclusive. The default value is 1.

Description

The aggregate storage cache facilitates memory usage during data loads, aggregations, and retrievals. Optionally use this setting to specify a number of threads across which to divide the cache allocation specified by [ASODEFAULTCACHESIZE](#).

Example

The *specified* *resource_usage* and *actual* resource usage of the aggregate storage cache will be different, depending on how many threads you specify in `ASOCACHECONCURRENTCONSUMINGTHREADS`. The formula to calculate actual resource usage is:

$$(1/\text{ASOCACHECONCURRENTCONSUMINGTHREADS}) * \text{resource_usage}$$

The actual aggregate storage cache size Essbase allocates, in megabytes, is calculated as:

$$\text{ASODEFAULTCACHESIZE} * \text{actual resource usage}$$

Assume the following configurations are set for the cube:

```
ASOCACHECONCURRENTCONSUMINGTHREADS 5  
ASODEFAULTCACHESIZE 500
```

With the above settings, you can initialize up to 5 ASO data load buffers, specifying, for each, the maximum resource_usage of 1.0.

Using the following MaxL statements, you initialize three load buffers:

```
alter database ASOSamp.Basic initialize load_buffer with buffer_id 1
resource_usage 1.0;
alter database ASOSamp.Basic initialize load_buffer with buffer_id 2
resource_usage 0.5;
alter database ASOSamp.Basic initialize load_buffer with buffer_id 3
resource_usage 0.1;
```

The following MaxL statement displays the actual resource usage:

```
query database ASOSamp.Basic list load_buffers;
buffer_id internal active resource_usage aggregation_method ignore_missings
ignore_zeros
-----+-----
1          FALSE   FALSE           0.2  AGGREGATE_SUM      FALSE      FALSE
2          FALSE   FALSE           0.1  AGGREGATE_SUM      FALSE      FALSE
3          FALSE   FALSE           0.02 AGGREGATE_SUM      FALSE      FALSE
```

See Also

[ASOBUFFERCOMMITWAIT](#)

[IDLETIMEMERGE](#)

ASODEFAULTCACHESIZE

The Essbase configuration setting ASODEFAULTCACHESIZE designates the default size for the aggregate storage cache associated with aggregate storage (ASO) cubes. The aggregate storage cache grows dynamically until it reaches this limit.

Syntax

```
ASODEFAULTCACHESIZE [appname] n
```

- *appname*—Optional. Specifies the application to which the ASODEFAULTCACHESIZE setting applies. If omitted, the setting applies to all new applications.
- *n*—An integer value indicating default aggregate storage cache size in megabytes. If ASODEFAULTCACHESIZE is not set, the default size is 100.

Description

The aggregate storage cache facilitates memory usage during data loads, aggregations, and retrievals. When an ASO application is started, Essbase allocates a small area in memory as the aggregate storage cache for the application. As needed, Essbase increases the cache size incrementally, until the maximum cache size specified for the application is reached, or until the operating system denies additional allocations.

Example

```
ASODEFAULTCACHESIZE 200
```

Sets the aggregate storage cache size of all newly created or migrated ASO databases as 200 megabytes.

See Also

[ASOCACHECONCURRENTCONSUMINGTHREADS](#)

ASODYNAMICAGGINBSO

The Essbase configuration setting ASODYNAMICAGGINBSO controls whether block storage (BSO) cubes use hybrid mode for queries. Hybrid mode for queries in BSO means that wherever possible, queries execute with efficiency similar to that of aggregate storage (ASO) cubes.

Hybrid mode is enabled by default for BSO queries (in other words, ASODYNAMICAGGINBSO is implicitly set to FULL). To enable hybrid mode for BSO calculation scripts, you must also enable [HYBRIDBSOINCALCSCRIPT](#) in the application configuration.

The ASODYNAMICAGGINBSO configuration setting applies only to block storage (BSO) cubes.

Syntax

```
ASODYNAMICAGGINBSO [appname [dbname]] NONE | PARTIAL | FULL | ONLY
```

- *appname*—Recommended. Specifies the application for which hybrid mode for queries is applied.
If you specify a value for *appname* and do not specify a value for *dbname*, then the configuration applies to all cubes in the specified application.
To enable the configuration for a specific cube, you must specify an application and cube.
- *dbname*—Recommended. Specifies the cube, in the application specified by *appname*, for which hybrid mode is applied.
If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- NONE—Disable hybrid mode for queries in BSO cubes.
- PARTIAL—Turn on hybrid mode only for simple outline formulas based on the consolidation operators +, -, and ~, but excluding the operators *, /, and %. Leave formulas to be calculated in non-hybrid mode.
- FULL—Turn on hybrid mode for queries and formula calculations. This is the default mode for new BSO cubes in Essbase 21c. If enabled, hybrid mode is in effect for member formulas using any of the supported functions. For a list of supported and unsupported functions, see [Functions Supported in Hybrid Mode](#).
- ONLY—Same as FULL, but if a query is not supported in hybrid mode, return an error instead of defaulting to non-hybrid mode. This can be useful for testing purposes while you are migrating a cube to hybrid mode.

 **Note:**

If ASODYNAMICAGGINBSO is not configured for the current application, and the application uses a feature that requires hybrid mode, then Essbase implicitly sets ASODYNAMICAGGINBSO to ONLY.

Example

```
ASODYNAMICAGGINBSO Sample PARTIAL
```

See Also

Adopt Hybrid Mode for Fast Analytic Processing.

MaxL statements alter application **set cache_size** and query application **get cache_size**, for managing the size of block-storage application cache.

ASODYNAMICAGGINBSOFOLDERPATH

The Essbase configuration setting ASODYNAMICAGGINBSOFOLDERPATH changes the location specification for hybrid mode directories.

ASODYNAMICAGGINBSOFOLDERPATH configuration applies only to block storage (BSO) cubes.

When a block storage cube uses hybrid mode for queries, the following subdirectories are created under the application directory:

- default
- log
- metadata
- temp

These subdirectories are similar to those found in aggregate storage application directories. When the application stops, the directories are removed, and when the application restarts, they are replaced.

Syntax

```
ASODYNAMICAGGINBSOFOLDERPATH [appname] path_to_directory
```

- *appname*—Optional application specification.
If you do not specify an application, the configuration applies to all applications and cubes on the Essbase Server.
- *path_to_directory*—Path to the new directory after you have moved it.

Example

```
ASODYNAMICAGGINBSOFOLDERPATH Sample \\machine-name\directory
```

ASODYNHIERASAGG

The Essbase configuration setting ASODYNHIERASAGG controls whether some upper-level members in aggregate storage (ASO) hierarchies should be aggregated rather than calculated as formulas.

ASODYNHIERASAGG configuration applies only to ASO cubes. Turning this setting on can provide a query performance enhancement for ASO cubes with dynamic hierarchies.

Syntax

```
ASODYNHIERASAGG [appname] OFF | ON
```

- *appname*—Optional. Specifies the application for which hybrid mode is used.
- OFF—This is the default.
- ON—Aggregate upper level members in aggregate storage hierarchies, when they meet the conditions.

Conditions

If ASODYNHIERASAGG is ON, a dynamic hierarchy (or fully dynamic dimension) member will be aggregated, rather calculated as a formula, if all of the following is true for that member:

- does not belong to an Accounts dimension
- does not belong to a typed measures dimension
- has no formula
- is not a dynamic time series member
- solve order is 0
- all children have only one of the following operators: +, ~ or ^
- all consolidating children (not those having only ~ or ^ operators) can be aggregated

Notes

- If the dynamic hierarchy/dimension is not the last one in the outline, the solve order may differ, and the results may differ.
- If the member is a shared member, its prototype is analyzed instead.
- If the member is a label-only member, its first child is analyzed instead.

Example

```
ASODYNHIERASAGG ON
```

ASOLOADBUFFERWAIT

The ASOLOADBUFFERWAIT configuration setting controls the maximum amount of time (in seconds) Essbase waits for aggregate storage cache resources to become available in order

to process ASO load buffer operations. If cache resources do not become available within the specified amount of time, Essbase terminates the load buffer operation.

ASOLOADBUFFERWAIT configuration is applicable when you initialize ASO data load buffers using the `wait_for_resources` option, and applies to allocations, custom calculations, and grid update operations.

This setting applies only to ASO cubes.

Syntax

```
ASOLOADBUFFERWAIT [appname [dbname]] n
```

- *appname*—Optional. Specifies the application for which the `wait_for_resources` option is to be set.

If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified ASO application.

To enable the setting for a specific cube, you must specify an application and cube.

If you do not specify an application, you cannot specify a cube, and the setting applies to all ASO applications and cubes on the Essbase Server.

- *dbname*—Optional. Specifies the cube, in the application specified by *appname*, for which the `wait_for_resources` option is to be set.

If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.

- *n*—Specifies the maximum number of seconds Essbase waits for cache resources to become available.

The default wait value is 10 seconds.

For changes to the configuration to take effect, you must restart the application.

Example

```
ASOLOADBUFFERWAIT ASOsamp Basic 20
```

Sets 20 seconds as the maximum wait time for cache resources to become available on the ASOSamp.Basic cube.

See Also

alter database MaxL statement

ASOSAMPLESIZEPERCENT

The ASOSAMPLESIZEPERCENT configuration setting specifies the number of cells sampled from input-level data in an Essbase aggregate storage (ASO) cube. The sampled data is used to estimate the size of aggregate views.

Larger sample sizes enable Essbase to make increasingly accurate estimates of average view sizes. View selection using a larger sample size enables Essbase to more closely meet the stop size.

Sample sizes are specified as a percentage of input-level data.

 **Note:**

To improve performance of aggregate storage (ASO) cubes in Essbase 21c, you can configure [METADATABASEDAGGVIEWSBUILD](#) to let Essbase automate the creation and maintenance of default aggregate views, based on metadata analysis. You can control the aggregate view size using [DEFAULTVIEWBUILDSIZE](#). Refer also to Optimization for Aggregate View Selection and Generate Aggregate Views Automatically.

Syntax

```
ASOSAMPLESIZEPERCENT [appname [dbname]] n
```

- appname*—**Optional**. ASO application for which sampled data is to be set.

If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.

To enable the setting for a specific cube, you must specify an application and cube.

If you do not specify an application, you cannot specify a cube, and the setting applies to all ASO applications and cubes on Essbase Server.
- dbname*—**Optional**. Specifies the cube, in the application specified by *appname*, for which sampled data is to be set.

If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- n*—A value ranging from 0.0 to 100.0, representing a percentage of input-level cells that are to be used for the ASO cell sample. To specify 0.5% for the sample size, enter 0.5, not 0.005. You do not need to divide by 100.

To calculate the number of sample cells, multiply the number of input-level cells by the percentage specified in *n*. The default, and minimum, sample size is 1 million (1,000,000) cells.

 **Note:**

For ASO databases that have 1 million or more cells, if the percentage specified by ASOSAMPLESIZEPERCENT results in a sample size of fewer than 1 million cells, the setting is ignored and Essbase uses 1 million cells. For cubes that have fewer than 1 million cells, the sample size is the same size as the cube.

Performance Impact

Estimates using larger sample sizes take longer to complete, which may have a significant performance impact on view selection. The recommendation for a cube with more than 1 billion input-level cells is to start with a small ASOSAMPLESIZEPERCENT setting such as 0.1 (meaning 0.1%). Slowly increase this setting until the preferred trade-off between view selection performance and accuracy is reached. The optimal setting for a cube with more than 1 billion cells will probably be less than 3%. Refer to Aggregate View Build Optimization.

To gauge the accuracy of view size estimates for aggregate views that have been built, use the following MaxL query database statement:

```
query database appname.dbname list existing_views
```

Compare the values in the columns named `size_ratio_estimate` and `size_ratio_actual`. The accuracy of each view size estimate differs for each aggregate view.

Example

```
ASOSAMPLESIZEPERCENT ASOsamp.Sample 1
```

AUDITTRAIL

Use the AUDITTRAIL configuration setting to track changes to Essbase.

Syntax

```
AUDITTRAIL DATA | SECURITY
```

- **DATA**—Valid to set only at application level. Track changes to cube data, including Smart View updates, changes to Linked Reporting Objects (LROs), adding notes, attaching files, and referencing URLs. View the audit log in Smart View or in the Essbase web interface. Data auditing is applicable only for block storage cubes. Data audit trail is not enabled by default.
- **SECURITY**—Valid to set only at server-level `essbase.cfg`. Gather information about server-level security, migration, file events, and executed MaxL statements, and consolidate it into a CSV file, based on parameters you specify in a policy XML file. Essbase tracks information about each event, including time, client, user, artifacts affected, and a description. Server-level auditing is applicable for both block storage and aggregate storage cubes. Server-level auditing is not enabled by default.

Example

```
AUDITTRAIL DATA
```

Tracks changes to cube data.

```
AUDITTRAIL SECURITY
```

Gathers information about system level security, migration, file events, and executed MaxL statements.

See Also

Track Data Changes

Audit Security, Artifact Changes, and LCM Events

AUTHENTICATIONMODULE

The AUTHENTICATIONMODULE configuration setting selects a method for Essbase user and group authentication.

When you configure Essbase using the configuration tool, you select an authentication provider for users and groups. The AUTHENTICATIONMODULE configuration setting is automatically added to the Essbase configuration file, with the value corresponding to your selection.

Syntax

```
AUTHENTICATIONMODULE CSS | OPSS_NATIVE
```

- **CSS**—Use EPM Shared Services
- **OPSS_NATIVE**— Use WebLogic security mode in conjunction with your choice of external authentication identity provider

AUTOMERGE

The AUTOMERGE configuration setting specifies whether incremental data slices are automatically merged during a data load to an Essbase aggregate storage (ASO) database.

AUTOMERGE configuration applies only to ASO databases, and does not apply to block storage (BSO) databases.

Syntax

```
AUTOMERGE ALWAYS | NEVER | SELECTIVE
```

- **ALWAYS**—Specifies to automatically merge incremental data slices during a data load to an aggregate storage database. By default, merges are executed once for every four consecutive incremental data slices. If, however, the [AUTOMERGEMAXSLICENUMBER](#) configuration setting is used, the auto-merge process is activated when the AUTOMERGEMAXSLICENUMBER value is exceeded.

The size of the incremental data slices is not a factor in selecting which ones are merged.

The default value is ALWAYS.

- **NEVER**—Specifies to never automatically merge incremental data slices during a data load to an aggregate storage database.

To manually merge incremental data slices, use the alter database MaxL statement with the **merge** clause.

- **SELECTIVE**—Specifies to activate the incremental data slice auto-merge process when the number of incremental data slices specified in the AUTOMERGEMAXSLICENUMBER configuration setting is exceeded. If the number of incremental data slices in the data load does not exceed the value of AUTOMERGEMAXSLICENUMBER, the auto-merge process is not activated.

Example

```
AUTOMERGE SELECTIVE
```

Specifies that the value of the AUTOMERGEMAXSLICENUMBER configuration setting determines whether the process of automatically merging incremental data slices is activated.

See Also

[AUTOMERGEMAXSLICENUMBER](#)

alter database merge ...

AUTOMERGEMAXSLICENUMBER

The Essbase configuration setting AUTOMERGEMAXSLICENUMBER specifies the maximum number of incremental data slices that can exist in a aggregate storage (ASO) data load without activating an automatic merge. When the value of AUTOMERGEMAXSLICENUMBER is exceeded, the automerge is activated.

Note:

To use the AUTOMERGEMAXSLICENUMBER configuration setting, the AUTOMERGE configuration setting must be set to SELECTIVE or ALWAYS.

AUTOMERGEMAXSLICENUMBER configuration applies only to ASO databases, and does not apply to block storage (BSO) databases.

Syntax

```
AUTOMERGEMAXSLICENUMBER n
```

n—Specifies the maximum number of incremental data slices that can exist in an ASO data load without activating an automatic merge.

- When the number of incremental data slices is equal to or less than *n*, the incremental data slices are not merged.
- When the number of incremental data slices is greater than *n*, the automerge process is activated.

The default value of AUTOMERGEMAXSLICENUMBER is 0.

During the automerge process, Essbase determines the maximum size, as a percentage, that any one incremental data slice can contribute to the maximum number of incremental input cells. Essbase counts the number of cells in all committed incremental data slices. Assume that *r* represents the maximum percentage. If the size of an incremental data slice, as a percentage, is:

- Equal to or less than *r*, the slice is added to the automerge list.
- Greater than *r*, the slice is not added to the automerge list.

Example

```
AUTOMERGEMAXSLICENUMBER 5
```


Activates the automerge process when the number of incremental data slices exceeds 5.

See Also[AUTOMERGE](#)

CALCCACHE

The CALCCACHE configuration setting specifies whether Essbase uses a calculator cache to manage data blocks during calculation. Using the calculator cache improves performance. You can specify the size of the cache using the SET CACHE command in a calculation script, or the CALCCACHE {HIGH | DEFAULT | LOW} configuration settings.

CALCCACHE configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CALCCACHE [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the application for which the setting applies.
If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.
To enable the setting for a specific cube, you must specify an application and cube.
If you do not specify an application, you cannot specify a cube, and the setting applies to all applications and cubes on the Essbase instance.
- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which the setting applies.
If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- TRUE—Essbase uses a calculator cache when calculating the cube. This is the default.
- FALSE—Essbase does not use a calculator cache when calculating the cube.

Description

If required during a calculation, you can override this default setting using the SET CACHE command in a calculation script.

When CALCCACHE is set to TRUE, Essbase uses the calculator cache, providing that:

- The cube has at least two sparse dimensions.
- You calculate at least one full sparse dimension (unless you specify the SET CACHE ALL option in a calculation script).

Example

```
CALCCACHE Sample Basic FALSE
```

See Also[SET CACHE command](#)

CALCCACHEHIGH

CALCCACHEDEFAULT

CALCCACHELOW

CALCCACHEDEFAULT

Using the Essbase calculator cache improves calculation performance in block storage (BSO) cubes.

Use CALCCACHEDEFAULT to set a default value for the calculation script SET CACHE command.

CALCCACHEDEFAULT configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CALCCACHEDEFAULT [appname [dbname]] n
```

- *appname*—Optional. Specifies the application for which the setting applies.
If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application.
To enable the setting for a specific database, you must specify an application and database.
If you do not specify an application, you cannot specify a database, and the setting applies to all applications and databases on Essbase Server.
- *dbname*—Optional. Specifies the database, in the application specified by *appname*, for which the setting applies.
If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- *n*—The default calculator cache size, in bytes. If you do not set a default value, it is 200,000 bytes.

Description

Using the calculator cache significantly improves calculation performance in BSO cubes. The size of the performance improvement depends on the database configuration.

You can specify whether Essbase uses a calculator cache by default using the CALCCACHE setting. If required during a calculation, override this default setting using the SET CACHE command in a calculation script.

Example

Assume the Essbase application configuration specifies these settings:

```
CALCCACHEHIGH 1000000  
CALCCACHEDEFAULT 300000  
CALCCACHELOW 200000
```

 **Note:**

In the Essbase configuration, a parameter is not followed by a semicolon; in a calculation script, a parameter must be followed by a semicolon.

You could then use the following SET CACHE commands in a calculation script:

```
SET CACHE HIGH;
```

sets a calculator cache of 1,000,000 bytes for the duration of the calculation script.

```
SET CACHE DEFAULT;
```

sets a calculator cache of 300,000 bytes for the duration of the calculation script.

```
SET CACHE LOW;
```

sets a calculator cache of 200,000 bytes for the duration of the calculation script.

See Also

SET CACHE command

[CALCCACHE](#) configuration setting

CALCCACHEHIGH

Using the Essbase calculator cache improves calculation performance in block storage (BSO) cubes.

Use CALCCACHEHIGH to set the high value for the calculation script SET CACHE command.

CALCCACHEHIGH configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CALCCACHEHIGH [appname [dbname]] n
```

- *appname*—Optional. Specifies the application for which the setting applies.
If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.
To enable the setting for a specific cube, you must specify an application and cube.
If you do not specify an application, you cannot specify a cube, and the setting applies to all applications and cubes on Essbase Server.
- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which the setting applies.
If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.

- n —The maximum calculator cache size, in bytes (not to exceed 200,000,000 bytes). By default, the maximum is 200,000.

Description

Essbase uses the calculator cache to create and track data blocks during calculation. Using the calculator cache significantly improves calculation performance. The size of the performance improvement depends on the cube configuration.

You can specify whether Essbase uses a calculator cache by default using the CALCCACHE setting. If required during a calculation, override this default setting using the SET CACHE command in a calculation script.

Example

Assume the Essbase configuration contains these settings:

```
CALCCACHEHIGH 1000000
CALCCACHEDEFAULT 300000
CALCCACHELOW 200000
```



Note:

In the Essbase configuration, a parameter is not followed by a semicolon; in a calculation script, a parameter must be followed by a semicolon.

You could then use the following SET CACHE commands in a calculation script:

```
SET CACHE HIGH;
```

sets a calculator cache of 1,000,000 bytes for the duration of the calculation script.

```
SET CACHE DEFAULT;
```

sets a calculator cache of 300,000 bytes for the duration of the calculation script.

```
SET CACHE LOW;
```

sets a calculator cache of 200,000 bytes for the duration of the calculation script.

See Also

SET CACHE command

[CALCCACHE](#) configuration setting

CALCCACHELOW

Using the Essbase calculator cache improves calculation performance in block storage (BSO) cubes.

Use CALCCACHELOW to set the low value for the calculation script SET CACHE command.

CALCCACHELOW configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CALCCACHELOW [appname [dbname]] n
```

- *appname*—Optional. Specifies the application for which the setting applies.
If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.
To enable the setting for a specific cube, you must specify an application and cube.
If you do not specify an application, you cannot specify a database, and the setting applies to all applications and cubes on Essbase Server.
- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which the setting applies.
If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- *n*—The minimum calculator cache size, in bytes. By default, the minimum is 200,000.

Description

Essbase uses the calculator cache to create and track data blocks during calculation. Using the calculator cache significantly improves calculation performance. The size of the performance improvement depends on the cube configuration.

You can specify whether Essbase uses a calculator cache by default using the CALCCACHE configuration setting. If required during a calculation, override this default setting using the SET CACHE command in a calculation script.

Example

Assume the Essbase configuration specifies these settings:

```
CALCCACHEHIGH 1000000  
CALCCACHEDEFAULT 300000  
CALCCACHELOW 200000
```



Note:

In the Essbase configuration, a parameter is not followed by a semicolon; in a calculation script, a parameter must be followed by a semicolon.

You could then use the following SET CACHE commands in a calculation script:

```
SET CACHE HIGH;
```

sets a calculator cache of 1,000,000 bytes for the duration of the calculation script.

```
SET CACHE DEFAULT;
```

sets a calculator cache of 300,000 bytes for the duration of the calculation script.

```
SET CACHE LOW;
```

sets a calculator cache of 200,000 bytes for the duration of the calculation script.

See Also

SET CACHE command

[CALCCACHE](#) configuration setting

CALCLIMITFORMULARECURSION

When set to true, the CALCLIMITFORMULARECURSION configuration setting prevents the Essbase Server from going beyond 128 formula execution levels.

CALCLIMITFORMULARECURSION configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CALCLIMITFORMULARECURSION TRUE | FALSE
```

- **TRUE**—Imposes a limit of 128 on the number of formula execution levels. This is the default.
- **FALSE**—Imposes no limit on the number of formula execution levels.

Description

CALCLIMITFORMULARECURSION limits the number of execution levels of Essbase formulas. If a calculation involves formulas referencing one or more members from sparse dimensions and there are formulas along dense dimension members, the formula execution may be recursive (have multiple execution levels). Formulas with excessive execution levels may crash the Essbase Server. Setting CALCLIMITFORMULARECURSION to TRUE prevents excessive execution levels.

If a formula reaches 128 execution levels and CALCLIMITFORMULARECURSION is set to TRUE (or default), Essbase stops processing that formula and writes error messages in the application log. If a formula reaches 128 execution levels and CALCLIMITFORMULARECURSION is set to FALSE, Essbase continues processing that formula and writes an information message in the application log.

Note:

This setting does not affect formulas in MDX queries (for example, calculated members).

Example

If you added a member named Payroll2 to the Measure dimension in Sample Basic and used the following formula to calculate it, you would get a recursion error if Market has more than 128 members:

```
Payroll / @SUMRANGE(Payroll, @IRDESCENDANTS(Market))
```

CALCMODE

The CALCMODE configuration setting enables you to globally set the Essbase formula execution mode to block mode or bottom-up mode, instead of selecting these modes in individual calculation scripts.

By default, block mode and bottom up mode are off. Hybrid mode is the default calculation and query mode for BSO cubes.

CALCMODE configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CALCMODE [appname [dbname]] [BLOCK | BOTTOMUP]
```

- *appname*—Optional. If you specify an application, all the cubes in that application are affected by the CALCMODE setting. If you leave out the application and database name parameters, the CALCMODE setting applies to the entire Essbase Server.
- *dbname*—Optional. If you specify an application and cube, the cube you specify is affected by the CALCMODE setting. If you do not specify an application with the cube, the CALCMODE setting will fail.
- BLOCK—Turns on block calculation mode.
- BOTTOMUP—Turns on bottom-up calculation mode.

Description

CALCMODE configuration setting allows you to set the calculation mode at the Essbase Server, application, or database (cube) level instead of indicating it in a calculation script using @CALCMODE.

Example

```
CALCMODE BLOCK
```

Turns on block calculation mode for all cubes and applications in the Essbase Server.

See Also

@CALCMODE function

CALCNOTICE

The CALCNOTICE configuration settings enable you to customize the number of notifications Essbase generates for the HIGH, DEFAULT, and LOW values of the SET NOTICE calculation command. SET NOTICE displays completion notices about the progress of the calculation.

CALCNOTICE configurations apply only to block storage (BSO) databases, and do not apply to aggregate storage (ASO) databases.

Syntax

```
CALCNOTICEHIGH | CALCNOTICEDEFAULT | CALCNOTICELOW n
```

- CALCNOTICEHIGH—The level with the maximum number of completion notices.
- CALCNOTICEDEFAULT—The level with the default number of completion notices.
- CALCNOTICELOW—The level with the minimum number of completion notices.
- *n*—Integer value for each level. It represents the number of notices to be displayed at set intervals during the calculation. The default is 10.

Description

CALCNOTICE defines the values for each of the three levels of the SET NOTICE calculation command.

SET NOTICE HIGH | DEFAULT | LOW generates completion notices during a calculation. The frequency and number of completion notices depends on the level specified.

The interval between notices is approximate. Essbase measures the interval by taking the number of data blocks already calculated as a percentage of the total number of possible data blocks in your cube.

For partial calculations and calculations with multiple passes through your cube, the interval between completion notices is approximate.

Notes

- The intervals between completion notices are approximate.
- Completion notices do not significantly reduce the calculation performance, except when used with a very small cube.

Example

If you apply the following configurations:

```
CALCNOTICEHIGH 50  
CALCNOTICEDEFAULT 20  
CALCNOTICELOW 5
```

Then SET NOTICE commands in a calculation script produce the following results:

```
SET NOTICE HIGH;
```


Displays 50 completion notices at 2% intervals.

```
SET NOTICE DEFAULT;
```

Displays 20 completion notices at 5% intervals.

```
SET NOTICE LOW;
```

Displays 5 completion notices at 20% intervals.

See Also

SET NOTICE (calculation command)

CALCOPTFRMLBOTTOMUP

The CALCOPTFRMLBOTTOMUP configuration setting specifies whether Essbase optimizes the calculation of complex formulas on sparse dimensions in large cube outlines. If enabled, Essbase performs a bottom-up calculation on formulas that would otherwise require a top-down calculation.

CALCOPTFRMLBOTTOMUP configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CALCOPTFRMLBOTTOMUP TRUE | FALSE
```

- TRUE—Optimizes the calculation of formulas on sparse dimensions in large outlines by forcing a bottom-up calculation.
- FALSE—Does not force a bottom-up calculation for formulas on sparse dimensions in large outlines. This is the default.

Description

CALCOPTFRMLBOTTOMUP tells Essbase whether to optimize the calculation of formulas on sparse dimensions in large cube outlines, so that you can efficiently use CALC ALL and CALC DIM commands to calculate the cube.

You can override the CALCOPTFRMLBOTTOMUP setting by using the SET FRMLBOTTOMUP command in a calculation script.

Notes

- For information on complex formulas and top-down calculations, see Bottom-Up and Top-Down Calculation.
- Forcing a bottom-up calculation on a formula may produce results that are inconsistent with a top-down calculation if:
 - The formula contains complex functions (for example, range functions)
 - The formula's dependencies are not straightforward

- Before using the CALCOPTFRMLBOTTOMUP setting in a production environment, be sure to check the validity of calculation results produced when the setting is enabled (set to TRUE).
- The SET CREATENONMISSINGBLK calculation command can force top-down calculations, regardless of the value of the CALCOPTFRMLBOTTOMUP setting.

Example

```
CALCOPTFRMLBOTTOMUP TRUE
```

See Also

SET FRMLBOTTOMUP (calculation command)

SET CREATENONMISSINGBLK (calculation command)

CALCPARALLEL

The CALCPARALLEL configuration setting enables an older method of parallel calculation. Parallel calculation lets you define the number of processing threads Essbase uses for calculation tasks.

CALCPARALLEL configuration can be applied to both block storage (BSO) and aggregate storage (ASO) databases. By default, BSO databases are calculated serially, and ASO databases are calculated using two threads.

Syntax

```
CALCPARALLEL [appname [dbname]] n
```

- *appname*—Optional. Specifies that parallel calculation applies to all cubes on the application. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the application. If you do not specify an application, you cannot specify a cube, and the setting applies to all applications and cubes.
- *dbname*—Optional. Specifies that parallel calculation applies only to this cube. If you specify a value for *dbname* but do not include *appname*, the parameter is ignored, and parallel calculation is enabled for all applications and cubes.
- *n*—A required parameter that specifies the number of threads to be made available for parallel calculation.
 - For block storage, an integer between 1-128. The default value, 1, specifies serial calculation: no parallel calculation takes place.
 - For aggregate storage, an integer from 1-128. 2 is the default value.

A value less than 1 is interpreted as the default size. A value greater than the maximum size is interpreted as the maximum size.

You must restart Essbase to initialize any change to the configuration.

Description

This setting enables CALCPARALLEL parallel calculation. For block storage databases, Essbase analyzes each pass of a calculation to determine whether parallel calculation would optimize the calculation. If it would not, Essbase uses serial calculation even if CALCPARALLEL is set to a number greater than 1.

Notes

- With block storage cubes, Essbase dynamically calculates the number of task dimensions for parallel calculation by starting with a value of 1, determining how many potential tasks are generated, and increasing the number of task dimensions until an optimal limit is reached. If CALCTASKDIMS has been used to increase the number of tasks and to decrease the size of each task identified for parallel calculation, the number of sparse dimensions set with CALCTASKDIMS is used. See Identifying Additional Tasks for Parallel Calculation for more information about what kind of outlines or calculation scripts generate many empty tasks.
- If you increase the number of calculation threads for aggregate storage databases, since the aggregate storage cache is split up amongst the threads, consider increasing the size of aggregate storage memory cache.
- When running a parallel calculation that includes the @XREF calculation function, the application returns a timeout error if the number of threads specified for CALCPARALLEL is higher than the number of threads specified by the SERVERTHREADS setting. For example, the default value of SERVERTHREADS is 20. If you set CALCPARALLEL to 25, an application timeout error is generated.
- To learn about a newer type of parallel calculation, see the FIXPARALLEL calculation command.

Example

```
CALCPARALLEL 3
```

Enables up to three threads to perform calculation tasks at the same time.

See Also

Using CALCPARALLEL Parallel Calculation

SET CALCPARALLEL

SET CALCTASKDIMS

[SERVERTHREADS](#)

FIXPARALLEL...ENDFIXPARALLEL

CALCREUSEDYNCALCBLOCKS

The CALCREUSEDYNCALCBLOCKS configuration setting for Essbase controls whether dynamically calculated values are re-used during retrievals.

CALCREUSEDYNCALCBLOCKS configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CALCREUSEDYNCALCBLOCKS TRUE | FALSE
```

- TRUE—Dynamically calculated values are re-used. This is the default.

- FALSE—Dynamically calculated values are not re-used.

Description

By default, Essbase re-uses dynamically calculated values during retrievals. This can speed up retrievals that involve a large number of dynamically calculated blocks that are each required to compute several other blocks, such as when there is a large hierarchy of sparse Dynamic Calc members. However, a large dynamic calculator cache size or a large value for the CALCLOCKBLOCK may adversely affect the retrieval performance when this method is used. In such cases, CALCREUSEDYNCALCBLOCKS should be set to FALSE.

Example

```
CALCREUSEDYNCALCBLOCKS TRUE
```

CALCTASKDIMS

The CALCTASKDIMS configuration setting enables you to specify how many of the sparse dimensions in an Essbase outline should be used to identify potential calculation tasks that can be run in parallel.

CALCTASKDIMS configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CALCTASKDIMS [appname [dbname]] n
```

- *appname*—Optional. CALCTASKDIMS applies to all cubes on the application. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the application. If you do not specify an application, you cannot specify a cube, and the setting applies to all applications and cubes.
- *dbname*—Optional. Cube name to which CALCTASKDIMS applies. If you specify a value for *dbname* but do not include *appname*, the parameter is ignored and the setting applies to all applications and cubes.
- *n*—Required. An integer specifying the number of sparse dimensions to be included when Essbase identifies tasks that can be performed at the same time.

A value of 1 indicates that only the last sparse dimension in the outline is used to identify tasks. A value of 2, for example, indicates that the last and second-to-last sparse dimensions in the outline are used. Because each unique combination of members from selected sparse dimensions is a potential task, the potential number of parallel tasks is the product of the number of members of the selected dimensions. The maximum value is the number of sparse dimensions in the outline.

Any value less than 1 is interpreted as 1, any value greater than the number of sparse dimensions in the outline is converted to the largest valid value.

You must restart Essbase to initialize any change to the configuration.

Notes

- If you do not notice an improvement in calculation performance after increasing the value of CALCTASKDIMS, see the note in the SET CALCTASKDIMS topic.

- Use the CALCTASKDIMS configuration setting only if your outline generates many empty tasks, thus reducing opportunities for parallel calculation. See Identifying Additional Tasks for Parallel Calculation for more information about what kind of outlines or calculation scripts generate many empty tasks.

Example

CALCTASKDIMS Sample Basic 2

Specifies that for application Sample and database Basic, the last two sparse dimensions in an outline will be used to identify potential tasks to perform at the same time during a calculation pass.

See Also

[CALCPARALLEL](#)

SET CALCPARALLEL calculation command

SET CALCTASKDIMS calculation command

CALCTRACE

The CALCTRACE configuration setting enables calculation tracing to help you debug Essbase calculation scripts. You can specify a single cell to trace from Smart View, or use SET TRACE commands in calculation scripts if you need to trace multiple cells.

CALCTRACE configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

The calculation tracing is done on the cell specified by using the SET TRACE calculation command, or by selecting the cell in Smart View. The output is available in Smart View, as well as in a file, `calc_trace.txt`, located in the cube directory.

Syntax

CALCTRACE OFF | ON

- OFF – Calculations are not traced. Any SET TRACE commands in calculation scripts are ignored. This is the default.
- ON – Calculations can be traced. You can specify a cell to be traced in Smart View by selecting a cell in the grid before executing a calculation script. You can also use SET TRACE commands in calculation scripts if you need to trace multiple cells.

Example

CALCTRACE ON

See Also

SET TRACE

Trace Calculations

CCTRACK

The CCTRACK configuration setting controls whether exchange rates are tracked as Essbase calculates currency conversions.

CCTRACK configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
CCTRACK TRUE | FALSE
```

- TRUE—Exchange rates are tracked while conversions are calculated. This is the default.
- FALSE—Turns off the tracking system.

Description

Tracking exchange rates has the following advantages:

- Allows conversion to occur at report time through the grid client or Report Writer
- Allows you to convert a converted currency back to its original, local rate using the command
- Prevents data inaccuracies due to accidental reconversion of data during a calculation

After loading data, you can clear the tracked exchange rates for the new data using the CLEARCCTRACK calculation command. During a calculation, you can enable or disable CCTRACK using the SET CCTRACKCALC calculation command.

Notes

- When CCTRACK is turned on, the following restrictions apply:
 - If you are using currency partitions, you cannot use a CCONV command with a FIX statement to convert a subset of a currency partition (a calculation script attempting such a FIX will not validate).
 - If you are not using currency partitions, you must use CCONV with a FIX statement.
- Setting CCTRACK to FALSE turns off the tracking system with the following results:
 - The CCONV command assumes that the data is unconverted (in local currency). If you accidentally run the CCONV command multiple times on the same data, the resulting data will be inaccurate.
 - Similarly, the currency report options assume that the data is unconverted (in local currency). If the data has already been converted in the database, it is reconverted at report time, resulting in inaccurate data.
 - The restrictions on using the FIX...ENDFIX and DATACOPY calculation commands in currency conversions do not apply. For example, if you are using currency partitions, you can now use the FIX command with the CCONV command to calculate a subset of a currency partition. If you are not using currency partitions, you can use CCONV without a FIX statement.

Example

```
CCTRACK TRUE
```

See Also

SET UPTOLOCAL (calculation command)

SET CCTRACKCALC (calculation command)

CLEARCCTRACK (calculation command)

CLIENTPREFERREDMODE

The CLIENTPREFERREDMODE configuration setting determines whether Essbase allows only secure TLS (SSL) connectivity for clients.

For information on implementing TLS, see *Secure Your Communication and Network with TLS and Certificates*.

Syntax

```
CLIENTPREFERREDMODE SECURE | CLEAR
```

- **SECURE**—Essbase communicates with clients using only secure connection mode with Transport Layer Security (TLS).
- **CLEAR**—Client sessions are based on the transport specified in the login API. If the secure transport is specified, then the session uses secure mode; otherwise, the session uses clear. The default value is CLEAR.

Example

```
CLIENTPREFERREDMODE SECURE
```

See Also

[AGENTSECUREPORT](#)

[ENABLECLEARMODE](#)

[ENABLESECUREMODE](#)

For information on implementing TLS, see *Secure Your Communication and Network with TLS and Certificates*.

CRASHDUMP

The CRASHDUMP configuration setting indicates whether Essbase saves a core dump to a file when an abnormal termination of an application server process occurs.

Syntax

```
CRASHDUMP TRUE | FALSE
```

- **TRUE**—Creates a directory containing a core file for each abnormal termination. This is the default.
- **FALSE**—No core file is created.

Description

CRASHDUMP helps diagnose abnormal program terminations.

In each instance of an application server crash, when CRASHDUMP is set to TRUE, Essbase creates the core file in a directory under `tmp`. The name of the new directory is `ESSSVR.timestamp`, where *timestamp* displays the date and time. For example:

```
/u01/tmp/ESSSVR.Tue_Dec_1_18_16_17_2020/core
```

If a server process is automatically shut down, the core file contains a core dump of that moment. If a server process is shut down manually, the core file may be empty.

Look for the core file any time you experience abnormal Essbase program terminations. If the file is not empty, provide it to Support and then remove it and its directory from the computer. If the core file is empty, remove it and its directory from the computer.

In normal operations without abnormal terminations, core files are not created.

Example

```
CRASHDUMP TRUE
```

See Also

[CRASHDUMPLOCATION](#)

CRASHDUMPLOCATION

The CRASHDUMPLOCATION configuration setting specifies the location where Essbase saves a core dump file when an abnormal application termination occurs. This configuration is applicable only if CRASHDUMP is set to TRUE.

Syntax

```
CRASHDUMPLOCATION path
```

where *path* is the fully-qualified path to the directory where Essbase should save the core dump file.

Description

In the event of abnormal program terminations, if CRASHDUMP is set to true and a CRASHDUMPLOCATION path is specified, the core files are generated in a uniquely named core file directory under the specified path.

If the location specified by *path* does not exist or does not have write permissions, the core files are generated in the default location, and an error message is logged in the server log files. The default location is described in [CRASHDUMP](#).

Example

```
CRASHDUMP /u01/crash
```


See Also[CRASHDUMP](#)

CUSTOMCALCANDALLOCTHRUINSERT

The CUSTOMCALCANDALLOCTHRUINSERT configuration setting for Essbase enables execution of aggregate storage custom calculations and allocations through MDX Insert.

Syntax

```
CUSTOMCALCANDALLOCTHRUINSERT [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the aggregate storage application to which the configuration applies.
- *dbname*—Optional. Specifies the aggregate storage cube to which the configuration applies.
- TRUE—The execution of aggregate storage custom calculations and allocations goes through MDX Insert.
- FALSE—Custom calculations and allocations do not execute through MDX Insert. This is the default.

See Also

- Performing Custom Calculations and Allocations on Aggregate Storage Databases
- USE_MDX_INSERT

DATACACHE SIZE

The DATACACHE SIZE configuration setting defines the data cache size for Essbase block storage cubes. The data cache is a buffer in memory that holds data blocks. Essbase allocates this memory during data load, calculation, and retrieval operations, as needed.

This setting does not apply to aggregate storage cubes.

Syntax

```
DATACACHE SIZE n
```

n—An integer value expressed in bytes (B), kilobytes (K), megabytes (M), or gigabytes (G):

- Minimum value: 3 megabytes (3M)
- Maximum value: 2 gigabytes (2G)
- Default value: 100 megabytes (100M)

If a value is given without a B, K, M, or G qualifier, it is assumed the value is in bytes.

The qualifier can be in upper or lowercase and can be entered adjacent to the value (10M) or separated by a space (10 M).

Description

DATACACHE SIZE specifies, in bytes, kilobytes, megabytes, or gigabytes, the size of the data cache for cubes.

Example

```
DATACACHE SIZE 90M
```

Sets the data cache size of databases to 90 megabytes.

DATAERRORLIMIT

The DATAERRORLIMIT configuration setting for Essbase determines the number of records that can be written to an error log during a data load operation.

Syntax

```
DATAERRORLIMIT n
```

n—The number of records, per data load or dimension build, that can be written to the error log. Default: 1000. Maximum: 65,000.

Description

DATAERRORLIMIT determines the number of records that can be written to the error log during data load or dimension build operations.

After the specified number of errors have been recorded, Essbase fails the operation and issues an error message.

Essbase logs data load and dimension build errors in the cube directory. The file format is `err_<database name>_<number>`.

Example

```
DATAERRORLIMIT 1000
```

See Also

Check Error Logs

[REJECTEDRECORDSLIMIT](#)

DEFAULTVIEWBUILD

The DEFAULTVIEWBUILD configuration setting applies only to aggregate storage (ASO) cubes in Essbase 19c, when you want Essbase to automate the creation and maintenance of default aggregate views.

In releases prior to Essbase 19c, Essbase created default aggregate views by using internal analysis based on data sampling. Starting in Essbase 19c, default view selection can be based on metadata analysis. In some cases, this algorithm is better for query performance.

Note: In Essbase 21c or later, this configuration is replaced by [METADATABASEDAGGVIEWSBUILD](#).

This configuration applies only to aggregate storage cubes. Use the [DEFAULTVIEWBUILDSIZE](#) configuration setting to limit the growth of cubes as a result of automatic aggregations.

Syntax

```
DEFAULTVIEWBUILD appname TRUE | FALSE
```

- *appname*—Application name. The configuration applies to all cubes within the named application. If unspecified, the configuration applies to all aggregate storage cubes on the Essbase Server.
- TRUE—Essbase performs automatic default aggregate views selection, build, and maintenance, meaning that default aggregate view selection and materialization are recreated automatically in response to the following qualifying cube events:
 - data is updated (loaded, calculated, allocated, deleted, or submitted from Smart View)
 - there is a metadata change that would cause an update of data indexes, requiring a redesign or rebuilding of aggregate views

For its view selection algorithm, Essbase uses database metadata analysis.

Use [DEFAULTVIEWBUILDSIZE](#) to limit the growth of cubes as a result of automatic views maintenance.

- FALSE—This is the default. Creation and maintenance of default aggregate views are not automated. Instead, you select, build, and maintain the aggregate views yourself. For its view selection algorithm, Essbase uses internal analysis based on data sampling.

Example

```
DEFAULTVIEWBUILD TRUE
```

See Also

[DEFAULTVIEWBUILDSIZE](#) to limit the growth of cubes as a result of automatic views maintenance

Generate Aggregate Views Automatically

Optimization for Aggregate View Selection

DEFAULTVIEWBUILDSIZE

The DEFAULTVIEWBUILDSIZE configuration setting enables you to specify the maximum allowed growth ratio of the aggregate storage (ASO) cube when you allow Essbase to automatically maintain default aggregate view selection and materialization.

This configuration applies only to aggregate storage cubes for which [METADATABASEDAGGVIEWSBUILD](#) is set to ON or AUTO (in Essbase 21c or later), or when [DEFAULTVIEWBUILD](#) configuration is set to TRUE (in releases prior to Essbase 21c).

Syntax

```
DEFAULTVIEWBUILDSIZE appname n
```

- *appname*—Application name. The configuration applies to all cubes within the named application.
- *n*—Total size ratio limiting the resulting aggregation views when METADATABASEDAGGVIEWSBUILD (or DEFAULTVIEWBUILD) configuration is enabled. The maximum growth of the aggregated cube must not exceed this ratio. For example, setting the the total size ratio to 1.3 means that the cube can grow by no more than 30% as a result of the automated aggregation.

Example

```
DEFAULTVIEWBUILDSIZE ASOSamp 1.3
```

See Also

Generate Aggregate Views Automatically

Optimization for Aggregate View Selection

DELIMITEDMSG

The DELIMITEDMSG configuration setting enables you to set whether fields in Essbase application logs are delimited.

Syntax

```
DELIMITEDMSG [TRUE | FALSE]
```

The default is FALSE (application log fields are undelimited).

Description

If DELIMITEDMSG is set to TRUE, and no value for [DELIMITER](#) is supplied, the default tilde (~) is used to delimit fields. If set to FALSE, any value specified in DELIMITER is ignored, and no special delimiter is used for logs.

Example

```
DELIMITEDMSG TRUE  
DELIMITER *
```

Essbase produces logs that use the asterisk (*) symbol as a delimiter between fields in a log.

See Also

[DELIMITER](#)

DELIMITER

The DELIMITER configuration setting enables you to select one of five symbols to delimit fields in Essbase application logs.

Syntax

```
DELIMITER [ ~ | ^ | * | : | & ]
```

Description

DELIMITER specifies which of five symbols that Essbase will use to delimit fields in logs. DELIMITER is ignored unless DELIMITEDMSG TRUE is also present in the configuration. If DELIMITEDMSG is TRUE but the delimiter is unspecified, the default delimiter is ~.

Example

```
DELIMITEDMSG TRUE  
DELIMITER *
```

Essbase produces logs that use the asterisk (*) symbol as a delimiter between fields in a log.

See Also

[DELIMITEDMSG](#)

DIMBUILDERRORLIMIT

The DIMBUILDERRORLIMIT configuration setting determines the number of records that can be written to an error log during a dimension build operation. After the specified number of errors have been recorded, Essbase no longer records any more errors, but continues the dimension build process.

Syntax

```
DIMBUILDERRORLIMIT n
```

n—The number of records, per dimension build, that can be written to the error log. Default: 20,000. Maximum: 65,000.

Notes

Essbase logs data load and dimension build errors in the cube directory. The file format is `err_<tablename>_<number>`.

Example

```
DIMBUILDERRORLIMIT 40000
```

See Also

Check Error Logs

[DATAERRRLIMIT](#)

DIMBUILDSTATSINTERVAL

The DIMBUILDSTATSINTERVAL configuration setting specifies the number of records to process before reporting on dimension build progress. Load status information is written to the Essbase log file.

Syntax

```
DIMBUILDSTATSINTERVAL [n]
```

n—Required. An integer specifying the number of records to process before updating the dimension build progress information. The default value is 20000.

Example

```
DIMBUILDSTATSINTERVAL 20000
```

If there are 50000 records to process in the data source, and DIMBUILDSTATSINTERVAL is defined at 20000, Essbase updates the dimension build progress after processing 20000 records, and then 40000 records.

DISABLEREPLMISSINGDATA

The DISABLEREPLMISSINGDATA configuration setting instructs Essbase not to replicate #MISSING values to a target partition, thus improving performance, potentially with less accurate data.

Syntax

```
DISABLEREPLMISSINGDATA [appname [dbname]] TRUE | FALSE
```

- *appname*—Application name. Optional parameter for applying the TRUE or FALSE setting to one or all cubes within the application. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application. If you do not specify an application, you cannot specify a cube, and the setting applies to all applications and cubes on the Essbase Server.
- *dbname*—Database (cube) name. Optional parameter for applying the TRUE or FALSE setting to the specified cubes within the specified application. If you do not specify a value for *dbname*, the setting applies to all cubes within the specified application. If *appname* is not specified, you cannot specify *dbname*.
- TRUE—#MISSING values are not replicated to the target for those applications and cubes specified through the *appname* and *dbname* parameters.

- FALSE—#MISSING values are replicated to the target for those applications and cubes specified through the *appname* and *dbname* parameters. This is the default.

Notes

DISABLEREPLMISSINGDATA applies only to replicated partitions on block storage cubes.

When #MISSING data is not replicated, a warning message is logged in the application log file.

Example

Assume a partition exists from Sample1 Basic (source) to Sample2 Basic (target). To prevent replication of #MISSING data, configure the following:

```
DISABLEREPLMISSINGDATA Sample1 Basic TRUE
DISABLEREPLMISSINGDATA Sample2 Basic TRUE
```

DLSINGLETHREADPERSTAGE

The DLSINGLETHREADPERSTAGE configuration setting instructs Essbase to load data using a single thread per processing stage, or to use the thread values specified in the DLTHREADSPREPARE and DLTHREADSWRITE configuration settings. These three configuration settings help you test and improve block storage data load performance.

Syntax

```
DLSINGLETHREADPERSTAGE [appname [dbname]] TRUE | FALSE
```

- *appname*—Application name. Optional parameter for applying the TRUE or FALSE setting to one or all databases within the application. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application. If you do not specify an application, you cannot specify a database and the setting applies to all applications and databases on the Essbase Server.
- *dbname*—Database name. Optional parameter for applying the TRUE or FALSE setting to a specific database within the specified application. If you do not specify a value for *dbname*, the setting applies to all databases within the specified application. If *appname* is not specified, you cannot specify *dbname*.
- TRUE—Tells Essbase **not** to use the values in the DLTHREADSPREPARE and DLTHREADSWRITE configuration settings when it performs a data load. Consequently, it performs all data load processes in single-thread stages.
- FALSE—Tells Essbase to use the thread values specified in the configuration settings DLTHREADSPREPARE and DLTHREADSWRITE as the numbers of threads to use in the preparation and write stages of data load processing. The default value is FALSE.

Description

This setting, and related settings [DLTHREADSPREPARE](#) and [DLTHREADSWRITE](#), are related to parallel data load processing. Data load processing is divided up into stages that are performed by Essbase using separate processing threads for each stage. By default, a single thread is used for each stage. Taking advantage of the multithreading capabilities of the server machine, the separate single-thread stages can be performed in parallel.

To improve data load performance by maximizing use of processor resource for your situation, you can use these settings to enable additional multiple-thread processing within the preparation and write stages of data load processing.

Notes

- While testing thread values for the DLTHREADSPREPARE and DLTHREADSWRITE configuration settings, you can use the DLSINGLETHREADPERSTAGE setting to quickly revert to using a single thread per stage.
- Enabling use of multiple threads during the preparation and write stages may produce little if any benefit on a single-processor machine.
- Optimizing factors such as the content and organization of the data source can enhance performance more than increasing the numbers of threads to be used.

Examples

Example 1

```
DLSINGLETHREADPERSTAGE Sample Basic TRUE
DLTHREADSPREPARE Sample Basic 3
DLTHREADSWRITE Sample Basic 4
```

Essbase ignores any values specified by DLTHREADSPREPARE and DLTHREADSWRITE while loading data to the Sample Basic application and database. As a result, Essbase uses single threads in each stage.

Example 2

```
DLSINGLETHREADPERSTAGE FALSE
DLTHREADSPREPARE Sample Basic 3
DLTHREADSWRITE Sample Basic 4
```

Based on the first setting, Essbase uses the number of threads specified by the DLTHREADSPREPARE and DLTHREADSWRITE configuration settings for all databases on the server. The settings on the second and third lines specify use of 3 processing threads for the preparation stages and 4 processing threads for the write stages when loading the Sample Basic application and database. Assuming that there are no further related settings, the default value 1 (one) is assumed for all other applications and databases on the server.

Example 3

```
DLSINGLETHREADPERSTAGE Sample FALSE
DLTHREADSWRITE Sample Basic 3
DLTHREADSWRITE Sample Interntl 4
```

In the above example, Essbase uses the number of threads specified by the DLTHREADSPREPARE and DLTHREADSWRITE configuration settings for all databases within the application named Sample. To enable usage of different numbers of threads for the write stage for the two different databases, two DLTHREADSWRITE settings are included with different thread values for each specific database. Because no

DLTHREADSPREPARE setting is specified, the preparation stage is single-threaded.

See Also

[DLTHREADSPREPARE](#)

[DLTHREADSWRITE](#)

[WORKERTHREADS](#)

Optimizing Data Loads

DLTHREADSPREPARE

The DLTHREADSPREPARE configuration setting specifies how many threads Essbase may use during the block storage data load preparation stage, which organizes the source data in memory in preparation for storing the data into blocks. Multiple threads, processing in parallel, may improve data load performance.

In order for Essbase to use the value specified for this setting, the DLSINGLETHREADPERSTAGE setting must be set to FALSE.

Syntax

```
DLTHREADSPREPARE [appname [dbname]] n
```

- *appname*—Application name. Optional parameter for using the specified number of threads in one or all databases within the application. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application. If you do not specify an application, you cannot specify a database and the setting applies to all applications and databases on the Essbase Server.
- *dbname*—Database name. Optional parameter for using the specified number of threads when loading the specified database within the specified application. If you do not specify a value for *dbname*, the setting applies to all databases within the specified application. If *appname* is not specified, you cannot specify *dbname*.
- *n*—The number of threads the data load process may use for preparing the data to be loaded. Specify an integer between 1 and 32. The default value is 1.

If *n* is greater than the maximum or a negative number, the value is assumed to be 32.

Description

DLTHREADSPREPARE, and related settings [DLTHREADSWRITE](#) and [DLSINGLETHREADPERSTAGE](#), are related to parallel data load processing. The concept of a *pipeline* is relevant to Essbase data loads. A pipeline is a series of data processing elements in memory that may be executed serially or in parallel. An Essbase data load operation uses a pipeline consisting of 5 stages. By default, a single thread is used for each stage. Therefore, all data load operations need a minimum of 5 threads.

To improve data load performance by maximizing use of processor resource for your situation, you can use these settings to enable additional multiple-thread processing within the preparation and write stages of data load processing. For more information about parallel thread processing in block storage data loads, see [Optimizing Data Loads](#).

Notes

- You can use another configuration setting, `DLTHREADSWRITE`, to specify the number of threads for the write stage of data load processing.
- Many factors affect the possible optimal values for `DLTHREADSPREPARE` including the number of processors on the machine and the number of other processes running on the machine. If you want to set this setting to a value higher than the default (1), check with your system administrator, as higher values can consume considerable system resources. As a rule of thumb, do not expect performance advantages if the number of threads for this setting is greater than the number of processors on the server machine.
- Setting the value for `DLTHREADSPREPARE` to be greater than 1 (one) may produce little if any benefit on a single-processor machine.

Example

```
DLSINGLETHREADPERSTAGE Sample Basic FALSE
DLTHREADSPREPARE Sample Basic 3
```

Because `DLSINGLETHREADPERSTAGE` is set to `FALSE` for the Sample Basic application and database, Essbase uses 3 parallel threads during the preparation stage when loading data to Sample Basic.

See Also

[DLTHREADSWRITE](#)

[DLSINGLETHREADPERSTAGE](#)

[WORKERTHREADS](#)

Optimizing Data Loads

DLTHREADSWRITE

The `DLTHREADSWRITE` configuration setting specifies how many threads Essbase may use during the stage of the block storage data load process that writes blocks on the disk. Multiple threads, processing in parallel, may improve data load performance.

Since Essbase uses a single thread during the write stage of the aggregate storage (ASO) data load process, `DLTHREADSWRITE` does not apply to ASO databases.

Syntax

```
DLTHREADSWRITE [appname [dbname]] n
```

- *appname*—Application name. Optional parameter for using the specified number of threads in one or all databases within the application. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application. If you do not specify an application, you cannot specify a database and the setting applies to all applications and databases on the Essbase Server.
- *dbname*—Database name. Optional parameter for using the specified number of threads when loading the specified database within the specified application. If you do not specify a

value for *dbname*, the setting applies to all databases within the specified application. If *appName* is not specified, you cannot specify *dbname*

- *n*—The number of threads the data load process may use for writing data blocks to the disk. Specify an integer between 1 and 32. The default value is 1. If *n* > 32, or a negative number, the value is assumed to be 32.

Description

This setting, along with [DLTHREADSPREPARE](#) and [DLSINGLETHREADPERSTAGE](#), are related to parallel data load processing. The concept of a *pipeline* is relevant to Essbase data loads. A pipeline is a series of data processing elements in memory that may be executed serially or in parallel. An Essbase data load operation uses a pipeline consisting of 5 stages. By default, a single thread is used for each stage. Therefore, all data load operations need a minimum of 5 threads.

To improve data load performance by maximizing use of processor resource for your situation, you can use these settings to enable additional multiple-thread processing within the preparation and write stages of data load processing.

You can specify [DLTHREADSWRITE](#) for individual databases, all databases within an application, or for all applications and databases on the server.

In order for Essbase to use the value specified for [DLTHREADSWRITE](#), the configuration setting [DLSINGLETHREADPERSTAGE](#) must be set to `FALSE`.

For more information about parallel thread processing in block storage data loads, see [Optimizing Data Loads](#).

Notes

- You can use another configuration setting, [DLTHREADSPREPARE](#), to specify the number of threads for the preparation stage of data load processing.
- Many factors affect the possible optimal values for [DLTHREADSWRITE](#) including the number of processors on the machine and the number of other processes running on the machine. If you want to set this setting to a value higher than the default (1), check with your system administrator, as higher values can consume considerable system resources. Do not expect performance advantages if the number of threads for this setting is greater than the number of processors on the server machine.
- Setting the value for [DLTHREADSWRITE](#) to be greater than 1 (one) may produce little if any benefit on a single-processor machine.

Example

```
DLSINGLETHREADPERSTAGE Sample Basic FALSE
DLTHREADSWRITE Sample Basic 3
```

Because [DLSINGLETHREADPERSTAGE](#) is set to `FALSE` for the Sample Basic application and database, Essbase uses 3 parallel threads during the write stage when loading data to Sample Basic.

See Also

[DLTHREADSPREPARE](#)

[DLSINGLETHREADPERSTAGE](#)

WORKERTHREADS

Optimizing Data Loads

DYNCALCCACHEBLKRELEASE

The DYNCALCCACHEBLKRELEASE configuration setting enables Essbase to create a temporary buffer for dynamic calculations in cases where the wait for space in the dynamic calculator cache has exceeded the specified wait time.

DYNCALCCACHEBLKRELEASE configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
DYNCALCCACHEBLKRELEASE [appname [dbname]] TRUE | FALSE
```

- *appname*—If you specify an application name, the setting applies to all cubes within the application. If you do not specify an application name, the setting applies to all applications and cubes on the server.
- *dbname*—If you specify a database (cube) name, the setting applies only to this cube. If you do not also specify an application name, the setting applies to all applications and cubes on the server.
- TRUE—Tells Essbase to make room available in the dynamic calculator cache by temporarily storing inactive blocks in a separate, compressed-block buffer.
- FALSE—This is the default value. Tells Essbase not to find room in the dynamic calculator cache for a different set of blocks. Instead, if allowed by the DYNCALCCACHEONLY setting, Essbase attempts to perform calculations on these blocks in memory outside the dynamic calculator cache.

Description

The dynamic calculator cache is a memory buffer that holds data blocks that are expanded to include dynamically calculated members. Essbase allocates memory in the dynamic calculator cache to store these blocks during retrievals or calculations that involve dynamically calculated members.

Using the dynamic calculator cache may improve retrieval performance by reducing the number of calls to the operating system to do memory allocations. The size of the improvement depends on your configuration.

Use DYNCALCCACHEBLKRELEASE to tell Essbase to make room available in the dynamic calculator cache, if needed, by compressing inactive blocks from that cache and attempting to temporarily store them in a separate, compressed-block buffer.

Notes

The following sequence of events must occur and settings must be defined before Essbase releases space in the dynamic calculator cache:

- The area allocated in the dynamic calculator cache has reached the maximum allowed (specified by DYNCALCCACHEMAXSIZE).
- DYNCALCCACHEWAITFORBLK is set as TRUE and the wait period specified by DYNCALCCACHEBLKTIMEOUT has been reached.

- DYNALCCACHEBLKRELEASE is set to TRUE. Essbase releases an area in the dynamic calculator cache by compressing blocks from this cache and attempting to store them temporarily in a compressed-block buffer. The size of this buffer is defined by the DYNALCCACHECOMPRBLKBUFSIZE configuration setting.

Example

```
DYNALCCACHEBLKRELEASE TRUE
```

Essbase makes needed space available in the dynamic calculator cache by compressing inactive blocks and temporarily storing them in a dynamic calculator cache compressed-block buffer.

See Also

[DYNALCCACHEMAXSIZE](#)

[DYNALCCACHEWAITFORBLK](#)

[DYNALCCACHEBLKTIMEOUT](#)

[DYNALCCACHEONLY](#)

[DYNALCCACHECOMPRBLKBUFSIZE](#)

DYNALCCACHEBLKTIMEOUT

Use the DYNALCCACHEBLKTIMEOUT configuration setting to specify the maximum number of seconds that Essbase should wait for space in the dynamic calculator cache in order to perform the requested calculation there.

DYNALCCACHEBLKTIMEOUT configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
DYNALCCACHEBLKTIMEOUT [appname [dbname]] n
```

- *appname*—If you specify an application name, the setting applies to all cubes within the application. If you do not specify an application name, the setting applies to all applications and cubes on the server.
- *dbname*—If you specify a database (cube) name, the setting applies only to this cube. If you do not also specify an application name, the setting applies to all applications and cubes on the server.
- *n*—A number of milliseconds. May or may not include a decimal point. Any number less than 1 will be treated as 1. The default value is 10000 milliseconds (10 seconds).

Description

If Essbase waits the entire number of seconds specified by DYNALCCACHEBLKTIMEOUT, it then checks the DYNALCCACHEBLKRELEASE setting to determine what to do next:

- To make room in the dynamic calculator cache by temporarily swapping out blocks in the dynamic calculator cache that are inactive

- If DYNCALCCACHEONLY is FALSE, to write and calculate the blocks in memory outside the dynamic calculator cache

The dynamic calculator cache is a memory buffer that holds data blocks that are expanded to include dynamically calculated members. Essbase allocates memory in the dynamic calculator cache to store these blocks during retrievals or calculations that involve dynamically calculated members.

The dynamic calculator cache can improve retrieval performance by reducing the number of calls to the operating system for memory allocations. The size of the performance improvement from using the dynamic calculator cache depends on your database configuration.

Notes

- Use the DYNCALCCACHEBLKRELEASE setting to tell Essbase where to store and calculate data blocks containing Dynamic Calc members if the wait for space in the dynamic calculator cache has exceeded the specified wait time.
- The DYNCALCCACHEBLKTIMEOUT configuration setting is meaningful only when the DYNCALCCACHEWAITFORBLK configuration setting is set to TRUE.

Example

```
DYNCALCCACHEBLKTIMEOUT 20000
```

Essbase waits up to 20 seconds for space in the dynamic calculator cache before checking the DYNCALCCACHEBLKRELEASE setting to determine the next step to take before performing the requested calculation.

See Also

[DYNCALCCACHEMAXSIZE](#)

[DYNCALCCACHEONLY](#)

[DYNCALCCACHEWAITFORBLK](#)

[DYNCALCCACHEBLKRELEASE](#)

[DYNCALCCACHECOMPRBLKBUFSIZE](#)

DYNCALCCACHECOMPRBLKBUFSIZE

The DYNCALCCACHECOMPRBLKBUFSIZE configuration setting specifies the size of a temporary buffer for storing compressed blocks in order to make more space in the Essbase dynamic calculator cache.

DYNCALCCACHECOMPRBLKBUFSIZE configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
DYNCALCCACHECOMPRBLKBUFSIZE [appname [dbname]] n
```

- *appname*—If you specify an application name, the setting applies to all cubes within the application. If you do not specify an application name, the setting applies to all applications and cubes on the server.
- *dbname*—If you specify a database (cube) name, the setting applies only to this cube. If you do not also specify an application name, the setting applies to all applications and cubes on the server.
- *n*—An integer expressed in bytes (B), kilobytes (K), megabytes (M), or gigabytes (G)
 - Minimum value: 0 megabytes (0 M). If the value is 0, Essbase does not use the compressed block buffer.
 - Default value: 1 megabyte (1M, which is ~1,000,000 bytes)
 - If a value is given without a B, K, M, or G qualifier, it is assumed the value is in bytes.
 - The qualifier can be in upper or lowercase and can be entered adjacent to the value (10M) or separated by a space (1M)

Description

Using the dynamic calculator cache may improve retrieval performance by reducing the number of calls to the operating system to do memory allocations. The size of the improvement depends on your configuration.

The dynamic calculator cache is a memory buffer that holds data blocks that are expanded to include dynamically calculated members. Essbase allocates memory in the dynamic calculator cache to store these blocks during retrievals or calculations that involve dynamically calculated members.

The dynamic calculator cache compressed-block buffer is an area in memory where Essbase compresses and temporarily stores blocks from the dynamic calculator cache to free space for other blocks for other calculations. When space is again available, Essbase decompresses blocks stored in the compressed-block buffer and returns them to the dynamic calculator cache.

In order to make space available in the dynamic calculator cache, Essbase uses the value specified by the `DYNCALCCACHECOMPRBLKBUFSIZE` configuration to size the dynamic calculator cache compressed-block buffer. Essbase temporarily stores compressed blocks from the dynamic calculator cache into this buffer under the following circumstances:

- The area allocated in the dynamic calculator cache has reached the maximum allowed (specified by `DYNCALCCACHEMAXSIZE`) and Essbase requires additional space for blocks to be calculated in the current query.
- `DYNCALCCACHEWAITFORBLK` is set to `TRUE` and the wait period specified by `DYNCALCCACHEBLKTIMEOUT` has been reached.
- `DYNCALCCACHEBLKRELEASE` is set to `TRUE`, indicating Essbase should release dynamic calculator cache area.

Notes

Essbase uses the temporary compressed-block buffer only when the `DYNCALCCACHEBLKRELEASE` configuration parameter is set to `TRUE` and the `DYNCALCCACHECOMPRBLKBUFSIZE` setting is greater than 0.

Example

```
DYNCALCCACHECOMPRBLKBUFSIZE 1000000
```

Sets 1,000,000 (one million) bytes as the size for the dynamic calculator cache compressed-block buffer.

See Also[DYNALCCACHEMAXSIZE](#)[DYNALCCACHEONLY](#)[DYNALCCACHEWAITFORBLK](#)[DYNALCCACHEBLKTIMEOUT](#)[DYNALCCACHEBLKRELEASE](#)

DYNALCCACHEMAXSIZE

The DYNALCCACHEMAXSIZE configuration setting specifies the maximum amount of memory allocated for the dynamic calculator cache for an Essbase block storage database. The specified value takes effect for all databases that are opened after the server is started.

The dynamic calculator cache is a memory buffer that holds data blocks that are expanded to include dynamically calculated members. Essbase allocates memory in the dynamic calculator cache to store these blocks during retrievals or calculations that involve dynamically calculated members.

Using dynamic calculator cache may improve retrieval performance by reducing the number of calls to the operating system to do memory allocations.

DYNALCCACHEMAXSIZE configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
DYNALCCACHEMAXSIZE [appname [dbname]] n
```

- *appname*—If you specify an application name, the setting applies to all databases within the application. If you do not specify an application name, the setting applies to all applications and databases on the Essbase Server.
- *dbname*—If you specify a database name, the setting applies only to the database. If you do not also specify an application name, the setting applies to all applications and databases on the Essbase Server.
- *n*—An integer expressed in bytes (B), kilobytes (K), megabytes (M), or gigabytes (G).
 - Minimum value: 0 megabytes (0 M). If the value is 0, Essbase does not use dynamic calculator cache.
 - Default value: 20 megabytes (20M, which is 20,971,520 bytes)
 - The maximum amount of memory that can be allocated is 256 GB. If a value is given without a B, K, M, or G qualifier, it is assumed the value is in bytes. The qualifier can be in upper or lowercase and can be entered adjacent to the value (10M) or separated by a space (10 M).

Example

```
DYNALCCACHEMAXSIZE 30M
```


Sets 30 megabytes as the maximum size for the dynamic calculator cache.

DYNCALCCACHEONLY

The Essbase configuration setting DYNCALCCACHEONLY specifies whether dynamic calculations can use memory outside the dynamic calculator cache in the case that it is full.

DYNCALCCACHEONLY configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
DYNCALCCACHEONLY [appname [dbname]] TRUE | FALSE
```

- *appname*—If you specify an application name, the setting applies to all cubes within the application. If you do not specify an application name, the setting applies to all applications and cubes on the server.
- *dbname*—If you specify a database (cube) name, the setting applies only to the cube. If you do not also specify an application name, the setting applies to all applications and cubes on the server.
- TRUE—Disallows the use of memory outside the dynamic calculator cache. If space for blocks with dynamically calculated members cannot be obtained from the dynamic calculator cache, Essbase generates an error message.
- FALSE—Allows the use of memory outside the dynamic calculator cache, if necessary, for blocks containing dynamically calculated members. The default value is FALSE.

Description

The dynamic calculator cache is a memory buffer that holds data blocks that are expanded to include dynamically calculated members. Essbase allocates memory in the dynamic calculator cache to store these blocks during retrievals or calculations that involve dynamically calculated members.

Using the dynamic calculator cache may improve retrieval performance by reducing the number of calls to the operating system to do memory allocations. The size of the improvement depends on your configuration.

When no room is available in the dynamic calculator cache, the DYNCALCCACHEWAITFORBLK and DYNCALCCACHECOMPRBLKBUFSIZE configuration settings provide options that could result in Essbase using memory outside the dynamic calculator cache to store blocks that contain dynamically calculated members. If you are experiencing a severe memory shortage, you can use the DYNCALCCACHEONLY setting to disallow the use of memory outside the dynamic calculator cache. If DYNCALCCACHEONLY is set to TRUE, instead of using memory outside the dynamic calculator cache, Essbase generates the error message, "Allocation outside the dynamic calculator cache is disallowed."

Notes

The default value of DYNCALCCACHEONLY is FALSE. Only set DYNCALCCACHEONLY to TRUE for one or more of the following circumstances:

- The operating system is not properly reclaiming memory outside the dynamic calculator cache.

- There is a severe memory shortage
- Tighter control is required over memory usage for dynamic calculations

Example

```
DYNCALCCACHEONLY TRUE
```

Specifies that the dynamic calculator cache is the only memory area that Essbase may use to store blocks that contain dynamically calculated blocks. If a retrieval requires space that is not available in the dynamic calculator cache, the execution of the retrieval is terminated. The user sees an error message that is also posted to the application log.

See Also

[DYNCALCCACHEMAXSIZE](#)

[DYNCALCCACHEWAITFORBLK](#)

[DYNCALCCACHEBLKTIMEOUT](#)

[DYNCALCCACHECOMPRBLKBUFSIZE](#)

[DYNCALCCACHEBLKRELEASE](#)

DYNCALCCACHEWAITFORBLK

The DYNCALCCACHEWAITFORBLK configuration setting specifies whether Essbase should wait for memory to be freed in the dynamic calculator cache, or use outside memory.

DYNCALCCACHEWAITFORBLK configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
DYNCALCCACHEWAITFORBLK [appname [dbname]] TRUE | FALSE
```

- *appname*—If you specify an application name, the setting applies to all cubes within the application. If you do not specify an application name, the setting applies to all applications and cubes on the server.
- *dbname*—If you specify a database (cube) name, the setting applies only to this cube. If you do not also specify an application name, the setting applies to all applications and cubes on the server.
- TRUE—Tells Essbase to wait for memory to be freed in the dynamic calculator cache.
- FALSE—This is the default. If allowed by the DYNCALCCACHEONLY setting, tells Essbase attempt to perform calculations on these blocks in memory outside the dynamic calculator cache.

If the DYNCALCCACHEONLY setting is TRUE, tells Essbase to generate an error message instead of using memory outside the dynamic calculator cache.

Description

The dynamic calculator cache is a memory buffer that holds data blocks that are expanded to include dynamically calculated members. Essbase allocates memory in the dynamic calculator cache to store these blocks during retrievals or calculations that involve dynamically calculated members.

Using the dynamic calculator cache may improve retrieval performance by reducing the number of calls to the operating system to do memory allocations. The size of the improvement depends on your configuration.

Use `DYNALCCACHEWAITFORBLK` to set or change how Essbase handles the situation when it needs additional memory to store blocks in the dynamic calculator cache for the cube.

When `DYNALCCACHEWAITFORBLK` is `TRUE`, Essbase waits to store and calculate data blocks in the dynamic-calculator-cache area that is currently in use by other queries.

When `DYNALCCACHEWAITFORBLK` is `FALSE`, if `DYNALCCACHEONLY` is also `FALSE`, instead of waiting for area in the dynamic calculator cache, Essbase attempts to store and calculate data blocks for the current query in memory outside the dynamic calculator cache. If `DYNALCCACHEONLY` is `TRUE`, Essbase generates an error message instead of using memory outside the dynamic calculator cache.

Notes

Use `DYNALCCACHEBLKTIMEOUT` to specify the maximum number of seconds that Essbase waits for space in the dynamic calculator cache.

Example

```
DYNALCCACHEONLY FALSE
DYNALCCACHEWAITFORBLK FALSE
```

Essbase attempts to perform the block calculation in memory outside the dynamic calculator cache, instead of waiting for space to become available in the dynamic calculator cache.

See Also

[DYNALCCACHEMAXSIZE](#)

[DYNALCCACHEONLY](#)

[DYNALCCACHEBLKTIMEOUT](#)

[DYNALCCACHEBLKRELEASE](#)

[DYNALCCACHECOMPRBLKBUFSIZE](#)

ENABLE_DIAG_TRANSPARENT_PARTITION

The Essbase configuration setting `ENABLE_DIAG_TRANSPARENT_PARTITION` specifies whether to log transaction response times for requests sent from a data source to a transparent partition target.

The transparent partition target can be either a block storage or aggregate storage cube. Logging these diagnostic messages is helpful when troubleshooting response times that are too slow.

Syntax

```
ENABLE_DIAG_TRANSPARENT_PARTITION [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the application for which logging diagnostic messages is to be enabled.

If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.

To enable the setting for a specific cube, you must specify an application and cube.

If you do not specify an application, you cannot specify a cube, and the setting applies to all applications and cubes on Essbase Server.

- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which logging diagnostic messages is to be enabled.

If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored, and logging diagnostic messages is enabled for all applications and cubes on Essbase Server.

- `TRUE`—Log transaction response times for requests to a transparent partition.
- `FALSE`—Do not log transaction response times for requests to a transparent partition. This is the default.

You must restart Essbase Server to initialize any change to the configuration file.

Description

When logging is enabled, Essbase writes messages to the source and target database log files during querying.

For every partial response sent to the target from the source, Essbase logs these messages:

- In the source cube's log file, the following message, of type INFO, provides the size of the response grid:

```
Sending response grid of size xxxxx.
```

- In the target cube's log file, the following message provides the size of the request grid issued to the source and an estimated response time:

```
Waiting for data from source system:application:database grid size sizeOfRequestGrid. Approximately one second is needed to fetch a grid of size one million cells with non-missing cell density of 7% from the source.
```

For every partial grid received from the source, Essbase logs the following message about the density of the grid to the target cube's log file:

```
Density of the grid xxxxxx of fetch size xxxxxx.
```

When an aggregate storage cube is the target of a transparent partition, you can set the request and response grid size.

Example

```
ENABLE_DIAG_TRANSPARENT_PARTITION ASOSamp TRUE
```

Enables logging of transaction response times for all cubes associated with the ASOSamp application.

See Also

[MAX_REQUEST_GRID_SIZE](#) configuration setting

[MAX_RESPONSE_GRID_SIZE](#) configuration setting

ENABLECLEARMODE

The ENABLECLEARMODE configuration setting determines whether Essbase allows clear connection mode (as opposed to secure mode with TLS / SSL). This setting applies to the Essbase Agent and applications.

Syntax

```
ENABLECLEARMODE TRUE | FALSE
```

- TRUE—Essbase handles plain TCP requests. The default value is TRUE.
- FALSE—Essbase handles only secure TLS requests, not plain TCP requests.

Description

When you configure Essbase using the configuration tool, you select whether to use secure communication mode. If you do not select secure mode, then clear mode is used. ENABLESECMODE and ENABLECLEARMODE configuration settings are automatically added to the Essbase configuration file, with the values corresponding to your selection.

Example

```
ENABLECLEARMODE FALSE
```

See Also

[AGENTSECUREPORT](#)

[CLIENTPREFERREDMODE](#)

[ENABLESECMODE](#)

For information on implementing TLS, see *Secure Your Communication and Network with TLS and Certificates*.

ENBLERTSVLOGGING

The ENBLERTSVLOGGING configuration setting determines whether Essbase logs runtime substitution variables that are used in a calculation script.

Runtime substitution variable log entries are written to the application log file.

Syntax

```
ENBLERTSVLOGGING [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the application for which runtime substitution variable logging is to be set.

If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application.

To enable the setting for a specific database, you must specify an application and database.

If you do not specify an application, you cannot specify a database, and the setting applies to all applications and databases on Essbase Server.
- *dbname*—Optional. Specifies the database, in the application specified by *appname*, for which runtime substitution variable logging is to be set.

If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- TRUE—Runtime substitution variables that are used in a calculation script are logged. For information about the format of these log entries, see *Logging Runtime Substitution Variables*.
- FALSE—Runtime substitution variables that are used in a calculation script are not logged. The default value is FALSE.

Example

```
ENBLERTSVLOGGING TRUE
```

See Also

SET RUNTIMESUBVARS calculation command

ENABLESECUREMODE

The ENABLESECUREMODE configuration setting determines whether Essbase allows secure connection mode with Transport Layer Security (TLS / SSL). This setting applies to the Essbase Agent and applications.

Syntax

```
ENABLESECUREMODE TRUE | FALSE
```

- TRUE—Secure mode is enabled. Essbase can handle TLS (SSL) requests.
- FALSE—Secure mode is not used. The default value is FALSE.

Description

When you configure Essbase using the configuration tool, you select whether to use secure communication mode. If you do not select secure mode, then clear mode is used. `ENABLESECUREMODE` and `ENABLECLEARMODE` configuration settings are automatically added to the Essbase configuration file, with the values corresponding to your selection.

Example

```
ENABLESECUREMODE TRUE
```

See Also

[AGENTSECUREPORT](#)

[CLIENTPREFERREDMODE](#)

[ENABLECLEARMODE](#)

For information on implementing TLS, see *Secure Your Communication and Network with TLS and Certificates*.

ESSBASESERVERHOSTNAME

The `ESSBASESERVERHOSTNAME` configuration setting specifies the computer host name of the Essbase Agent and Essbase Server, and where an Essbase application process runs.

Syntax

```
ESSBASESERVERHOSTNAME server_name
```

where *server_name* is the name of the host where your Essbase application process runs. `ESSBASESERVERHOSTNAME` uses the current server by default.

Description

`ESSBASESERVERHOSTNAME` identifies the host where your Essbase application process runs. The value must be a valid host name and must map to an IP address assigned to the computer.

Notes

- You can use `ESSBASESERVERHOSTNAME` to restrict the network interface provider on which Essbase and the applications listen when multiple instances are running on the same host.
- The behavior of the client is not necessarily tied to this configuration setting. For example, the MaxL client always uses localhost as the default, irrespective of this configuration setting.
- In Report Writer, you can display `ESSBASESERVERHOSTNAME` values on a report. For example, you can use the `*MACHINE` replacement value in the Report Writer `{Mask}` command to display the `ESSBASESERVERHOSTNAME` value as the server name.

- In a clustered environment, the IP address must be accessible from both nodes of the cluster.

Example

```
ESSBASESERVERHOSTNAME essbase_server1
```

ESTIMATEDHASHSIZE

The ESTIMATEDHASHSIZE configuration setting specifies, in millions, the estimated number of Essbase member name and alias name strings that are loaded into memory for optimal performance of name lookup and name insertion during dimension build and outline editing.

ESTIMATEDHASHSIZE allows you to configure a new hash-table implementation, which has an increased memory footprint. The value that you set for this configuration setting affects the amount of memory used when editing an outline. If you set the value to a number that is lower than the estimated number of strings, dimension build performance might be impacted.

This configuration setting applies to block storage and aggregate storage applications.

Syntax

```
ESTIMATEDHASHSIZE [appname] n
```

- *appname*—Optional. Specifies the application for which the estimated hash size applies. If you do not specify an application, the setting applies to all applications on Essbase Server.
- *n*—Specifies the estimated number of strings that are populated in an extended hash table. The value must be an integer between 1 and 256. A value of 1 represents 1 million strings; a value of 256 represents 256 million strings.

The default value is 5 (5 million strings).

Example

```
ESTIMATEDHASHSIZE Sample 50
```

Sets the estimated number of member name and alias name strings that are loaded into memory to 50 million for the Sample application.

EXCEPTIONLOGOVERWRITE

The EXCEPTIONLOGOVERWRITE configuration setting determines whether Essbase overwrites the existing exception log or creates a new exception log.

Syntax

```
EXCEPTIONLOGOVERWRITE TRUE | FALSE
```

- TRUE—Essbase overwrites the existing exception log.
- FALSE—Essbase keeps the existing exception log and creates new logs for every exception. This is the default.

Description

EXCEPTIONLOGOVERWRITE determines whether Essbase overwrites existing exception log data or creates a new log for each exception condition. The exception log name is normally `log00001.xcp`.

When EXCEPTIONLOGOVERWRITE is FALSE:

- Essbase creates a new log instead of overwriting the previous one.
- Subsequent logs are numbered sequentially; for example, if `log00001.xcp` exists, the next log has the file name `log00002.xcp`, and the next has `log00003.xcp`, and so on.

The Essbase exception handler writes the information into the exception log on the local disk in a text file as follows:

- If the server crashed, the log is written in the directory pointed to by `<Essbase Path>`.
- If the application crashed and the application name is unknown, the log is written into the `app` subdirectory under the directory pointed to by `<Application Directory>`.
- If the application crashed and the application name is known, but the cube name is unknown, the log is written to the appropriate application directory; for example, `<Application Directory>/app/app1`.
- If the application crashed and both the application and cube names are known, the log is written to the appropriate cube directory; for example, `<Application Directory>/app/app1/cube1`.

See Environment Locations in the Essbase Platform for information about directory locations in Essbase.

Example

```
EXCEPTIONLOGOVERWRITE FALSE
```

EXCLUSIVECALC

The EXCLUSIVECALC configuration setting determines whether Essbase allows calculations to run concurrently in the same cube, or prevents it. By default, concurrent calculations are allowed.

EXCLUSIVECALC configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
EXCLUSIVECALC TRUE | FALSE
```

- TRUE—If a calculation operation (command or script) is running, Essbase fails any other calculation operations.
- FALSE—Essbase allows concurrent calculation operations. This is the default.

Example


```
EXCLUSIVECALC TRUE
```

EXPORTTHREADS

The EXPORTTHREADS configuration setting for Essbase defines the default number of threads that can be produced during parallel data export.

Syntax

```
EXPORTTHREADS appname dbname n
```

- *appname*—This is the name of the application. You can also use `xxxxx` as a wildcard to indicate all application names.
 - *dbname*—This is the name of the database. You can also use `xxxxx` as a wildcard to indicate all database names.
 - *n*—This integer sets the default for the number of export threads that can be used to export data. The default is 1.
 - **Block storage cubes:** The number of threads is an integer, between 1 and 1024, inclusive. The number of available block-address ranges limits the number of export threads. Essbase divides the number of actual data blocks by the specified number of 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. 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.
- **Aggregate storage cubes:** The number of threads is an integer, between 1 and 8, inclusive. This number should generally be equal to the number of processors on the machine that you wish to commit to doing parallel export. However, for parallel export on a very small aggregate storage cube 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.

Description

EXPORTTHREADS enables you to specify the number of threads that can be used to export data. The export process is then executed in parallel, and multiple threads can retrieve data and write to their corresponding export files concurrently. If EXPORTTHREADS is not specified, or is not followed by its arguments, then the default value of 1 is used.

Example

```
EXPORTTHREADS Sample Basic 4
```

See Also

Export Data (MaxL)

[WORKERTHREADS](#)

FAILOVERMODE

The FAILOVERMODE configuration setting determines whether Essbase is deployed as a failover cluster that is managed by WebLogic Server, or as a stand-alone server.

FAILOVERMODE is available only for independent deployments, and should be set in the server-level `essbase.cfg` file.

Syntax

```
FAILOVERMODE TRUE | FALSE
```

- TRUE—Essbase runs as a failover cluster managed by WebLogic Server.
- FALSE—Essbase runs as a stand-alone server. The default value is FALSE.

Description

When FAILOVERMODE is TRUE, WebLogic Server and Essbase Agent automatically manage nodes and make one of the nodes active for processing Essbase requests.

Example

```
FAILOVERMODE FALSE
```

FEDERATEDAVCALC

FEDERATEDAVCALC is an application level configuration setting, applicable only for Essbase federated partition cubes, which enables or disables calculation and data load to be performed in Autonomous Data Warehouse. This functionality is new starting with Essbase 21.5.

Syntax

```
FEDERATEDAVCALC TRUE|FALSE
```

- TRUE—Calculation and data load operations for federated partition cubes are enabled. TRUE is the default behavior for federated partition cubes.
- FALSE—Users can only query on federated partition cubes. Calculation and data load operations are not permitted through Essbase.

Description

FEDERATEDAVCALC TRUE is the default behavior for federated partition cubes, even if this configuration is not explicitly set.

When FEDERATEDAVCALC is (explicitly or implicitly) set to TRUE, it enables Essbase data loads and block storage calculations to be performed against data in Autonomous Data

Warehouse, whenever possible. Exceptions, where Essbase compensates, are listed in Calculate and Query Federated Cubes and in Restrictions for Federated Partitions.

When `FEDERATEDAVCALC` is set to `FALSE`, calculation and data load operations on federated partition cubes are not permitted through Essbase, but queries are. For example, Essbase block storage (BSO) calculation scripts and data load jobs may not be performed, but queries from Smart View can be made, and the query results reflect the data that is in Autonomous Data Warehouse. The logic for query processing is pushed to Autonomous Data Warehouse whenever possible, as SQL statements. Otherwise, Essbase compensates to process the query.

Only set `FEDERATEDAVCALC` to `FALSE` if you have a reason to disallow calculation scripts and data load to be performed through Essbase to Autonomous Data Warehouse.

Though this setting affects whether block-storage (BSO) calculation is enabled for federated partition cubes, it does not affect how Essbase processes aggregate storage (ASO) custom calculations and allocations for federated partition cubes. Essbase Server always processes the ASO calculations and pushes the results to Autonomous Data Warehouse.

Example

```
FEDERATEDAVCALC TRUE
```

FILEGOVPATH

The `FILEGOVPATH` configuration setting sets the default file path location where Essbase saves client-based data exports on the Essbase Server. By default, file governance is implicitly ON, and the *path* is equivalent to the cube directory.

Syntax

```
FILEGOVPATH [ path | OFF ]
```

- *path*—Specifies the directory on Essbase Server to save client data exports, if something other than the default (cube directory) is preferred. By default, exports to the server are saved to the cube directory, unless you specify a different *path* in this configuration.
- OFF—File governance is turned OFF. Exports to the server are saved either to `<Application Directory>/app`, or to the absolute path clients provide in their export operations.
- *no value*—If `FILEGOVPATH` is present in the configuration but has no value, this means that file governance is turned ON, and data exports to the server are always saved to the cube directory on the server. If a client data export specifies an alternate path, the specification is ignored.

See Environment Locations in the Essbase Platform for information about `<Application Directory>`, cube directory, and other directory locations in Essbase.

Description

This configuration setting sets the default file path location where Essbase should save client-based data exports on the server. This configuration is applicable whether you are using:

- MaxL export data with the **server** and **data_file** keywords
- the `DATAEXPORT` command for calculation scripts

- API calls for exporting data

Example 1

```
FILEGOVPATH OFF
```

MaxL client data exports are saved to <Application Directory>/app on the Essbase Server. For example, the following MaxL statement creates the export file as <Domain Root>/applications/essbase/app/exp_file.txt:

```
export database Sample.Basic data to server data_file 'exp_file.txt';
```

Example 2

```
FILEGOVPATH
```

MaxL client data exports to the server are saved to the appropriate cube directory on the Essbase Server. For example, the following MaxL statement creates the export file as <Domain Root>/applications/essbase/app/Sample/Basic/exp_file.txt:

```
export database Sample.Basic data to server data_file 'exp_file.txt';
```

Example 3

```
FILEGOVPATH /scratch/export/
```

MaxL client data exports to the server are saved to <path> on the Essbase Server. For example, the following MaxL statement creates the export file as /scratch/export/exp_file.txt:

```
export database Sample.Basic data to server data_file 'exp_file.txt';
```

Example 4

```
FILEGOVPATH /scratch/export/
```

MaxL client data exports to the server are saved to <path> on the Essbase Server. For example, the following MaxL statement creates the export file as /scratch/export/export_dir/Sample/Basic/exp_file.txt:

```
export database Sample.Basic data to server data_file '/export_dir/exp_file.txt';
```

See Also

export data (MaxL)

DATAEXPORT calculation command

FILELOCKINGMODE

The FILELOCKINGMODE configuration setting specifies how files are locked for Essbase deployed on Linux systems. File locking mode is not applicable to Windows deployments.

Syntax

```
FILELOCKINGMODE Advisory | Mandatory | None
```

- **Advisory**—Locks files. All applications that follow rules (including Essbase) will honor the locks.
- **Mandatory**—Locks files at the kernel level. This setting provides extra security to protect against malicious software. This is the default setting.
- **None**—No files are locked. This option is added for Failover mode where file access is not done by the operating system, but by an application acquiring a lease.

Notes

When **FAILOVERMODE** is set to TRUE, setting FILELOCKINGMODE to Advisory or Mandatory has no effect.

Example

```
FILELOCKINGMODE Mandatory
```

FORCEALLDENSECALCON2PASSACCOUNTS

Normally, a two-pass tagged member of a dense accounts dimension triggers a second calculation pass on all dense cells of the data block. If you set the FORCEALLDENSECALCON2PASSACCOUNTS configuration setting to false, Essbase prevents the second pass for all other than the cells for the member tagged as two-pass.

Syntax

```
FORCEALLDENSECALCON2PASSACCOUNTS TRUE | FALSE
```

- **TRUE**—When a two-pass member of a dense accounts dimension is calculated, the second calculation pass calculates all dense cells of the data block. This is the default.
- **FALSE**—In the same situation, the FALSE setting blocks the second calculation pass for all dense cells except those affiliated with the two-pass member.

Description

FORCEALLDENSECALCON2PASSACCOUNTS addresses the situation where a two-pass member of a dense accounts dimension links through @XREF to a two-pass member of a dense accounts dimension in another cube outline, and that two-pass member links back to the original outline. The additional calculations in the second calculation pass can result in an infinite loop. The FALSE parameter value blocks the additional calculations. If you are very cautious about data correctness, check calculation results.

Example

```
FORCEALLDENSECALCON2PASSACCOUNTS FALSE
```

FORCEGRIDEXPANSION

The Essbase configuration setting FORCEGRIDEXPANSION, when set to ON, forces the expansion of the grid when transparent partitions are queried, helping to ensure correct results when used with GRIDEXPANSION in certain conditions.

Syntax

```
FORCEGRIDEXPANSION [appname [dbname]] ON | OFF
```

- *appname*—Optional. If you specify an application name, the setting applies to all cubes within the named application. If you do not specify an application name, the setting applies to all applications and cubes on the Essbase Server.
- *dbname*—Optional. If you specify a database (cube) name and an application name, the setting applies only to the named cube. If you do not also specify an application name, the cube is ignored and the setting applies to all applications and databases on the Essbase Server.
- ON— Forces grid expansion for transparent partition queries.
- OFF— Does not force grid expansion for transparent partition queries. This is the default.

Description

If GRIDEXPANSION is set to ON, the grid is not expanded if all of the following conditions are met, and, thus, incorrect results are returned:

- The client queries the target cube of a transparent partition.
- The client query requests values from a dynamically calculated block.
- Cells requested from the dynamically calculated block reference dense, dynamically calculated members.
- Dense, dynamically calculated members depend on values from one or more source databases.

When both GRIDEXPANSION and FORCEGRIDEXPANSION are set to ON, the grid is expanded and the correct values for cells that contain data are displayed. Query performance, however, is slowed.

If GRIDEXPANSION is set to OFF, the FORCEGRIDEXPANSION setting is ignored.

See Also

[GRIDEXPANSION](#)

[GRIDEXPANSIONMESSAGES](#)

FORCESHUTDOWNINTERVAL

The FORCESHUTDOWNINTERVAL configuration setting enables you to set the interval at which Essbase checks to confirm that the Essbase application server process (ESSSVR) is running. The default interval is 600 seconds.

Failover support for Essbase is a function of the WebLogic Server interface on which the Essbase Java Agent runs. The agent and the Essbase Server continually register "heartbeats" with the application server, to confirm that it is still actively running. If a server process does not return a regular heartbeat, the agent assumes there is a problem, and terminates the server process.

FORCESHUTDOWNINTERVAL configuration applies to block storage and aggregate storage cubes.

Syntax

```
FORCESHUTDOWNINTERVAL n
```

Example

```
FORCESHUTDOWNINTERVAL 2000
```

Checks for a heartbeat every 2000 seconds.

GRIDEXPANSION

The GRIDEXPANSION configuration setting, when set to ON, improves performance of some queries on Essbase transparent partition cubes.

Syntax

```
GRIDEXPANSION [appname [dbname]] ON | OFF
```

- *appname*—Optional. If you specify an application name, the setting applies to all databases within the named application. If you do not specify an application name, the setting applies to all applications and databases on the Essbase Server.
- *dbname*—Optional. If you specify a database name and an application name, the setting applies only to the named database. If you do not also specify an application name, the database is ignored and the setting applies to all applications and databases on the Essbase Server.
- ON—This is the default value. Enables grid expansion.
- OFF— Suppresses grid expansion.

Description

GRIDEXPANSION improves performance of some queries. If all of the following conditions are met, however, client queries may receive incorrect results (such as most data values displaying as #MISSING, whether or not cells contain data):

- The client queries the target database of a transparent partition.

- The client query requests values from a dynamically calculated block.
- Cells requested from the dynamically calculated block reference dense, dynamically calculated members.
- Dense, dynamically calculated members depend on values from one or more source databases.

If client queries receive incorrect results, set `FORCEGRIDEXPANSION` to `ON`. (If `GRIDEXPANSION` is set to `OFF`, the `FORCEGRIDEXPANSION` setting is ignored.)

See Also

[FORCEGRIDEXPANSION](#)

[GRIDEXPANSIONMESSAGES](#)

GRIDEXPANSIONMESSAGES

The `GRIDEXPANSIONMESSAGES` configuration setting sets whether Essbase grid expansion-related messages are displayed to Smart View and other grid client users, and are written to the application log.

Syntax

```
GRIDEXPANSIONMESSAGES [appname [dbname]] ON | OFF
```

- *appname*—Optional. If you specify an application name, the setting applies to all databases within the named application. If you do not specify an application name, the setting applies to all applications and databases on the Essbase Server.
- *dbname*—Optional. If you specify a database name and an application name, the setting applies only to the named database. If you do not also specify an application name, the database is ignored and the setting applies to all applications and databases on the Essbase Server.
- `ON`—Allows grid-expansion-related messages.
- `OFF`—This is the default value. Suppresses grid-expansion-related messages.

Description

If a grid client user retrieves data from a partition, the following message may be displayed repeatedly and written to the application log:

```
Grid expansion enabled for this query
```

To prevent this message from appearing, set `GRIDEXPANSIONMESSAGES` to `OFF`.

Example

```
GRIDEXPANSIONMESSAGES OFF
```

See Also

[GRIDEXPANSION](#)

[FORCEGRIDEXPANSION](#)

GRIDSUPPRESSINVALID

The GRIDSUPPRESSINVALID configuration setting for Essbase sets whether invalid attribute combinations, which are represented on the grid by #invalid, are suppressed in Smart View.

An invalid attribute combination is the result of an intersection of a dimension member for which an attribute is not assigned or, if an attribute is assigned to the member, the attribute combination is not within the scope of the grid query or the assigned attribute is incorrect. Invalid attribute combinations are suppressed when the row contains all #invalid values. Valid combinations with #MISSING values are not suppressed.

GRIDSUPPRESSINVALID configuration applies to block storage and aggregate storage cubes.

Syntax

```
GRIDSUPPRESSINVALID [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. If you specify an application name, the setting applies to all cubes within the named application. If you do not specify an application name, the setting applies to all applications and cubes on the Essbase Server.
- *dbname*—Optional. If you specify a database (cube) name and an application name, the setting applies only to the named cube. If you do not also specify an application name, the setting applies to all applications and cubes on the Essbase Server.
- TRUE—Enables suppressing invalid attribute combinations on the grid.
- FALSE—Invalid attribute combinations are not suppressed on the grid. This is the default value.

Example

```
GRIDSUPPRESSINVALID Sample Basic TRUE
```

Suppresses #invalid values in the Sample.Basic cube.

See Also

Suppressing Invalid Attribute Combinations in the Grid.

HYBRIDBSOINCALCSCRIPT

The HYBRIDBSOINCALCSCRIPT configuration setting controls whether block storage (BSO) cubes in the Essbase application use hybrid mode in calculation scripts when stored members depend on dynamic members.

When set to FULL, the BSO calculation engine uses hybrid mode to calculate the results, and then stores them.

Hybrid mode means that wherever possible, data calculation executes with efficiency similar to that of aggregate storage (ASO) cubes.

HYBRIDBSOINCALCSCRIPT is not applicable to ASO cubes. It is applicable only for BSO cubes that have hybrid mode enabled for query (i.e., ASODYNAMICAGGINBSO is set to something other than NONE).

If you enable this setting, do not disable [ASODYNAMICAGGINBSO](#), which is on by default for BSO cubes (meaning hybrid mode is enabled for queries).

Syntax

```
HYBRIDBSOINCALCSCRIPT [appname [dbname]] FULL|NONE
```

- *appname*—Optional. Specifies the application for which hybrid mode is used for calculation scripts.
If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.
To enable the setting for a specific cube, you must specify an application and cube.
- *dbname*—Optional. Specifies the cube, in the application specified by *appname*, for which hybrid mode is used.
If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- FULL—BSO calculation scripts run in hybrid mode.
- NONE—Calculation scripts run in non-hybrid mode, block storage format. This is the default.



Note:

If your cube has a federated partition to Autonomous Data Warehouse, then this setting is FULL by default, and you cannot change it to NONE.

Notes

The following limitations apply to hybrid mode for calculation scripts. If encountered, Essbase defaults to block storage format for these kinds of calculation scripts.

- CALC DIM, CALC ALL, AGG, and any other assignment-free expressions that calculate a sub-tree, do not use hybrid mode.
Oracle recommends limiting your use of CALC DIM and AGG to dimensions wherein no stored members are dependent on dynamic members. To calculate upper-level stored members that depend on dynamic members, use assignment formulas with calculation functions.
- DATAEXPORT for dynamic members does not use hybrid mode.
- Intelligent calculation does not use hybrid mode.
- Do not use CREATENONMISSINGBLOCK or CREATEBLOCKONEQ in calculation scripts you want to run in hybrid mode.
- CALCPARALLEL is not supported in hybrid mode. For parallel calculation, use FIXPARALLEL.

Example

```
HYBRIDBSOINCALCSCRIPT FULL
```

See Also[ASODYNAMICAGGINBSO](#)

Adopt Hybrid Mode for Fast Analytic Processing

SET HYBRIDBSOINCALCSCRIPT

IDCS_REQ_LIMIT_PER_SEC

The IDCS_REQ_LIMIT_PER_SEC configuration setting manages the rate of requests Essbase sends to OCI Identity and Access Management (IAM) or Oracle Identity Cloud Service (IDCS) each second.

Syntax

```
IDCS_REQ_LIMIT_PER_SEC n
```

n—Specifies the number of requests per second. Default value is no limit.

Description

IDCS_REQ_LIMIT_PER_SEC configuration manages the rate of requests sent to IAM or IDCS from Essbase. When there are more requests than the configured value, they are not rejected; instead, they are handled the next second through a managed queue.

Essbase does not send any direct IAM or IDCS requests. They are sent indirectly through other user management APIs. The rate of requests is an approximate value and may not exactly match the number specified with this configuration setting.

IDCS_REQ_LIMIT_PER_SEC helps to avoid the error, "Operations Error Http status: 429 message: Too Many Requests."

Example

```
IDCS_REQ_LIMIT_PER_SEC 50
```

IDLETIMEMERGE

The IDLETIMEMERGE configuration setting enables the performing of slice merge operations while the server is idle. This allows incremental slices that were created with new data during Essbase aggregate storage data load to be merged soon after load, during server idle time.

Syntax

```
IDLETIMEMERGE TRUE | FALSE
```

The default IDLETIMEMERGE value is false.

Description

IDLETIMEMERGE specifies whether slice merge is enabled and performed during server idle time.

Use IDLETIMEMERGE to enable or disable continuous slice merge.

Example

```
IDLETIMEMERGE TRUE
```

Sets a continuous slice merge to be enabled during server idle time.

See Also

[SVRIDLETIME](#)

[ASOBUFFERCOMMITWAIT](#)

[ASOCACHECONCURRENTCONSUMINGTHREADS](#)

IGNORECONSTANTS

The IGNORECONSTANTS configuration setting controls whether #Missing values, when used as operands in formulas, should remain #Missing after the Essbase formula calculation.

Syntax

```
IGNORECONSTANTS TRUE | FALSE
```

- TRUE—#Missing values remain missing regardless of interaction with formula constants.
- FALSE— Default option. #Missing values can be changed by interaction with formula constants.

Description

If a #Missing data value is processed in a formula with a constant or other data-independent construct, the default behavior is that #Missing is treated like a data value. For example, if A is missing, A+5 returns 5.

If you set IGNORECONSTANTS to TRUE, #Missing is not treated like a data value. For example, if A is missing, A+5 returns #Missing.

Example

If the configuration is as follows:

```
IGNORECONSTANTS TRUE
```

then the result for X in the following formula is #Missing

```
IF (X)  
5;  
ELSE  
3  
ENDIF
```

See Also

[QUERYBOTTOMUP](#) configuration setting

@QUERYBOTTOMUP calculation function

IMPLIED_SHARE_ON_CREATE

By default, implied sharing is not enabled in Essbase 21c. You can use the `IMPLIED_SHARE_ON_CREATE` configuration setting to enable it if needed.

If implied sharing is enabled, parent members share the value of their single child members (or of their only consolidating child member), unless the parent member has a formula, or is set to Never Share.

Syntax

```
IMPLIED_SHARE_ON_CREATE [app_name] TRUE | FALSE
```

- *app_name*—Optional. If provided, the setting applies only to the specified application; otherwise, the setting applies to the Essbase Server.
Application-specific settings take precedence over server-level settings.
- `TRUE`—Parent is treated as an implied share when it has only one child or when it has only one child that consolidates to the parent.
- `FALSE`—Never use Implied Share. This is the default behavior.

About Implied Sharing

Shared members typically are used to calculate the same member across multiple parents; for example, to calculate a Diet Cola member in both the Cola and Diet parents.

The data values associated with a shared member come from another member with the same name, called the prototype member. The shared member stores a pointer to data contained in the prototype member, and the data is stored only once. You use the shared member property to define a shared data relationship explicitly.

Implied shared members act as shared, even if you have not explicitly set them as shared.

If your applications depend on the use of implied sharing, you can set this configuration to `TRUE`.

Migration Guidelines for Implied Sharing

`IMPLIED_SHARE_ON_CREATE` configuration setting defines the default implied sharing behavior for Essbase application, when it is first created in (or migrated to) Essbase 21c.

In Essbase 11g On-Premise, the implied share setting could be changed for an application using the `IMPLIED_SHARE` setting. In Essbase 21c and later, you can set implied share behavior only once, at application creation time.

If you migrate an application from Essbase 11g On-Premise to Essbase 21c, and `essbase.cfg` has a value set for the `IMPLIED_SHARE` configuration, that setting will be preserved in `IMPLIED_SHARE_ON_CREATE` in Essbase 21c. In other words, an Essbase 11g On-Premise application configured as `IMPLIED_SHARE TRUE` will migrate to Essbase 21c as `IMPLIED_SHARE_ON_CREATE TRUE`.

If you migrate an application from Oracle Analytics Cloud - Essbase or Essbase 19c to Essbase 21c, and it has an application-level value set for the `IMPLIED_SHARE` configuration, that setting will be preserved in `IMPLIED_SHARE_ON_CREATE` in Essbase 21c.

Changing the `IMPLIED_SHARE_ON_CREATE` setting does not affect any existing applications or cubes. You cannot toggle this setting for an existing application. If you change this setting at the server level `essbase.cfg`, then only newly created applications will be affected.

If you need to change your implied share settings for only one existing Essbase 21c application, you will have to recreate it, using the following workflow:

1. Recreate the application, but not the cube.
2. Add the `IMPLIED_SHARE_ON_CREATE` setting with your new chosen value of `TRUE` or `FALSE` to the application configuration.
3. Create the cube and rebuild your outline. Once the setting is applied to the cube, it is retained in the `.otl` file.
4. Reload the data.
5. Run aggregation or calculation scripts.

If you need to change your implied share settings for more than one application,

1. In the Essbase Server configuration file `essbase.cfg`, add `IMPLIED_SHARE_ON_CREATE` with your preferred value of `TRUE` or `FALSE`.
2. Save the file and restart Essbase Server.
3. Recreate each application (without the cube). If you need a different Implied Share setting for any application than what is defined globally in `essbase.cfg`, add the `IMPLIED_SHARE_ON_CREATE` setting with your preferred value of `TRUE` or `FALSE` to that application's configuration.
4. Rebuild the cubes from the source outlines.
5. Reload the data.
6. Run aggregation or calculation scripts.

Example

```
IMPLIED_SHARE_ON_CREATE Sample TRUE
```

Use Implied Share for application Sample.

See Also

Understanding Implied Sharing

INDEXCACHESIZE

The index cache is a buffer in memory that holds index pages for block storage databases. Essbase allocates the memory upon startup of the cube. The `INDEXCACHESIZE` configuration setting enables you to define the size of the index cache.

The value of the index cache size can be expressed in bytes, kilobytes, megabytes, or gigabytes. Terabytes must be expressed in gigabytes.

`INDEXCACHESIZE` configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
INDEXCACHESIZE n
```

n—An integer value expressed in bytes (B), kilobytes (K), megabytes (M), or gigabytes (G):

- Minimum value: 1 megabytes (1M)
- Maximum value: 256TB
- Default value: 100M

If a value is given without a B, K, M, or G qualifier, it is assumed the value is in bytes.

The qualifier can be in upper or lowercase and can be entered adjacent to the value (10M) or separated by a space (10 M).

Example

```
INDEXCACHESIZE 200M
```

Sets the index cache size of cubes to 200 megabytes.

INPLACEDATAWRITE

In-place data writing means that when updates occur, the data block can be written to the same location, as long as the compressed size of the data block fits in its original location on the disk.

In-place data writing can help reduce data fragmentation and lower the need for frequent restructuring of the Essbase cube. It also reduces the need for frequent index updates, resulting in improved performance.

The INPLACEDATAWRITE configuration setting enables or disables in-place data writing as an alternative to block updates requiring new disk locations.

Use this configuration only when [ORACLEHARDWAREACCELERATION](#) is true.

INPLACEDATAWRITE configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
INPLACEDATAWRITE [appname [dbname]] TRUE|FALSE
```

- *appname*—Optional. If provided, the setting applies only to the specified application; otherwise, the setting applies to all applications.
- *dbname*—Optional. Specifies the database (cube) in the application specified by *appname*. If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- TRUE—In-place data writing is enabled when ORACLEHARDWAREACCELERATION is true. This is the default.
- FALSE—In-place data writing is disabled.

Example

```
INPLACEDATAWRITE Sample Basic FALSE
```

See Also

[ORACLEHARDWAREACCELERATION](#)

[INPLACEDATAWRITEMARGINPERCENT](#)

INPLACEDATAWRITEMARGINPERCENT

In-place data writing can help reduce data fragmentation and lower the need for frequent restructuring of the Essbase cube. It also reduces the need for frequent index updates, resulting in improved performance.

The INPLACEDATAWRITEMARGINPERCENT configuration setting defines how much extra margin to allow in the data block for writebacks. This helps prevent any fragmentation that could be caused if writes use a new page location.

In-place data writing means that when updates occur, the data block can be written to the same location, as long as the compressed size of the data block fits in its original location on the disk.

Use this configuration only when [ORACLEHARDWAREACCELERATION](#) is true, and [in-place data writing](#) is enabled.

INPLACEDATAWRITEMARGINPERCENT configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
INPLACEDATAWRITEMARGINPERCENT n
```

n—A margin percentage of additional block space to allow for a writeback, so it can avoid allocating a new page location. The default is 20.

Example

```
INPLACEDATAWRITEMARGINPERCENT 25
```

See Also

[ORACLEHARDWAREACCELERATION](#)

[INPLACEDATAWRITE](#)

JAVAMAXSMARTLISTSPEROUTLINE

A text list (or smart list) is an object that stores text values for a cell. The JAVAMAXSMARTLISTSPEROUTLINE configuration setting sets a maximum number of smart lists (text lists) that can be referenced in a Java API Essbase outline.

Syntax

```
JAVAMAXSMARTLISTSPEROUTLINE appname n
```

- *appname*—The application to which the setting applies.
- *dbname*—Optional. Specifies the database (cube) in the application specified by *appname*. If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- *n*—A value specifying the maximum number of allowed text lists. The minimum value is 1024. The maximum and default value is 4294967295 (4 G). Prior to Release 11.1.2.2.100, the default was 65536.

Example

```
JAVAMAXSMARTLISTSPEROUTLINE Sample 16384
```

See Also

Working with Typed Measures

JVMMODULELOCATION

The JVMMODULELOCATION configuration setting specifies a Java Virtual Machine (JVM) library Essbase should use. This parameter is useful if you have more than one version of the JVM library installed, or if you do not have the JVM location set in LD_LIBRARY_PATH.

Syntax

```
JVMMODULELOCATION pathToJVM
```

pathToJVM—Specifies a fully-qualified path and file name of a Java Virtual Machine library to be used by Essbase.

Description

If you do not include JVMMODULELOCATION in the Essbase configuration, or if you include it with an incorrect path and file name, Essbase searches the LD_LIBRARY_PATH for a version of the JVM library, and uses the first version that it finds.

If you specify JVMMODULELOCATION with no path parameter, it disables JVM-dependent functions, including custom-defined macros and custom-defined functions in the Calculator module.

Example

UNIX example

```
JVMODULELOCATION /usr/java/jdk1.8.0_261/jre/lib/amd64/server/libjvm.so
```

Windows example

```
JVMODULELOCATION C:\Program Files\Java\jdk1.8.0_261\jre\bin\server
```

The path name cannot include spaces. In Essbase configuration, a parameter is not followed by a semicolon. Do not enclose the path parameter in quotation marks.

See Also

Java Requirement and Tips

LOCKTIMEOUT

The LOCKTIMEOUT configuration setting limits the number of seconds a Smart View or other Essbase grid client user can hold an exclusive lock.

LOCKTIMEOUT configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
LOCKTIMEOUT n
```

where *n* is the number of seconds. The default value is 3600 seconds (60 minutes).

Description

LOCKTIMEOUT specifies, in seconds, the maximum amount of time a Smart View or other grid client user can hold an exclusive lock on a block. This setting applies to all applications and cubes on the Essbase Server, and is meant to specify a default value for newly created or migrated applications. To override this default for any application, specify a value using the MaxL alter application statement.

Example

```
LOCKTIMEOUT 300
```

Commits locked data and releases the exclusive lock after the lock has been held for 300 seconds (five minutes).

LOGMESSAGELEVEL

The LOGMESSAGELEVEL configuration setting enables you to set the priority level of messages that Essbase writes to the application log. The message levels are ERROR, WARNING, INFO, DEBUG, and FATAL.

Syntax

```
LOGMESSAGELEVEL ERROR | WARNING | INFO | DEBUG | FATAL
```

where ERROR, WARNING, INFO, DEBUG, and FATAL are priority levels:

- ERROR—Only error messages are written to the application log. This is the default.
- WARNING—Warning and error messages are written to the application log.
- INFO—Info, warning, and error messages are written to the application log.
- DEBUG—Debug, info, warning, and error messages are written to the application log.
- FATAL—Only messages about application failure are written to the application log.

Example

```
LOGMESSAGELEVEL WARNING
```

Sets the log message level to WARNING. Only warning and error messages are written to the application log.

LONGCALCTIMETHRESHOLD

The LONGCALCTIMETHRESHOLD configuration setting enables you to track information about long-running, top-level command blocks in Essbase block storage calculation scripts. Specify the upper-limit calc execution time, in seconds, before which you want to enable tracking.

Syntax

```
LONGCALCTIMETHRESHOLD n
```

where *n* is the upper-limit execution time, in seconds, before which Essbase tracks the calculations and logs alert messages. The default is 0 seconds (calculation tracking is off).

Description

Use LONGCALCTIMETHRESHOLD to specify the upper-limit execution time, in seconds, that top-level calculation command blocks can run before being considered long running and thus subject to tracking. A top-level command block is a command block that is not enclosed in a FIX/ENDFIX block. Essbase sends alert messages if any top level command block runs longer than the specified time limit.

The logged messages include the following information.

- CALC_USER: ID of the user running the calculation script

- `CALC_SCRIPT_NAME`: Calculation script name
- `LINE_NUMBERS`: Range of script line numbers encompassed by the calculation command
- `Calc Command Text`: First 500 characters of the calculation command code
- `BLOCKS_CREATED`: Number of blocks created by the calculation command
- `BLOCKS_READ`: Number of blocks read into memory during execution of the calculation command
- `BLOCKS_WRITE`: Number of blocks updated by the calculation command
- `EXEC_TIME`: Time, in milliseconds, that the calculation command has been running
- Statistics about custom-defined functions, if any were executed.

Examples

```
LONGCALCTIMETHRESHOLD 60
```

Specifies message logging for calculations running 60 seconds or longer.

```
LONGCALCTIMETHRESHOLD .001
```

Turns on diagnostics for most calculation commands (the threshold is very small, but not zero).

The following request-pending message indicates that user *essuser* executed calculation script *esscalcxl.csc* on cube *Sample.Basic*. The execution time of the command block from line 9 - 15 was about 75 seconds. There were 562500, 562500, and 1125000 blocks read, written, and created, respectively. The text of the command block is printed inside {} braces.

```
[2019-11-27T17:43:51.783-07:00] [Sample] [NOTIFICATION:16] [CAL-912] [CAL]
[ecid: 1445992953893,0]
[tid: 1107577152] [Basic/essuser] [CALC_USER: essuser] [CALC_SCRIPT_NAME:
esscalcxl.csc] [LINE_NUMBERS: 9 - 15]
[EXEC_TIME: 75420] [BLOCKS_READ: 562500] [BLOCKS_WRITE: 562500]
[BLOCKS_CREATED: 1125000]
Calc Command Text [ FIX ("Conversion Rate","No Period","Plan_A","No CO","No
Agreement","No Supplier",@RELATIVE
("From Group1",0),@RELATIVE("ToCO Group1",0),"No From Currency","No To
Currency") "All Product"
(@CREATEBLOCK("No Product");)ENDFIX]
```

LONGQUERYTIMETHRESHOLD

The `LONGQUERYTIMETHRESHOLD` configuration setting enables you to capture statistics on long-running Essbase grid and MDX queries. Specify the lowest query-time length, in seconds, for which you want to capture statistical information.

When you enable `LONGQUERYTIMETHRESHOLD`, Essbase:

- briefly summarizes query performance in the Oracle Diagnostic Log (ODL) for each query running longer than the specified time value
- sets a threshold/filter on what is logged for query tracing (when the `TRACE_REPORT` setting is enabled)

Syntax

```
LONGQUERYTIMETHRESHOLD n
```

where *n* is a number of seconds. If set to 0, then this setting is off (this is the default). If set to -1, this setting is always on.

Example

```
LONGQUERYTIMETHRESHOLD 60
```

Specifies statistics collection for queries running 60 seconds or longer.

Configuring Ongoing Query Tracing

Use the configuration settings TRACE_REPORT and LONGQUERYTIMETHRESHOLD together to manage the query tracing information that should be collected for the cube in the application's Oracle Diagnostic Log (ODL), and/or printed to `trace_report.log`.

You can enable TRACE_REPORT for tracking general query performance, but Oracle recommends filtering out all but the longest running queries by also setting LONGQUERYTIMETHRESHOLD.

Ideally,

- Configure Essbase to only log information for the top 5 percent of longest-running queries. If you don't know this value, 60 seconds may be a good place to start.
- Regularly check that log files do not grow too large.

Configuration Type	TRACE_REPORT value	LONGQUERYTIMETHRESHOLD value	Effect on Logs
ODL only, with threshold	Set to 0 (or not set at all)	Set to a value <i>n</i> (seconds) that is greater than 0	Performance metrics on all queries running longer than <i>n</i> seconds are printed to the application's ODL log.
ODL and trace, with threshold	Set to -1	Set to a value <i>n</i> (seconds) that is greater than 0	Performance metrics on all queries running longer than <i>n</i> seconds are printed to the application ODL log. Information on queries running longer than <i>n</i> seconds is printed to <code>trace_report.log</code> .

Configuration Type	TRACE_REPORT value	LONGQUERYTIMETHR ESHOLD value	Effect on Logs
Trace without threshold - <i>not recommended!</i>	Set to -1	Set to 0 (or not set at all)	Information on all queries, regardless of running time, is printed to <code>trace_report.log</code> . This configuration is not recommended, because <code>trace_report.log</code> can become too large if not filtered using a threshold, and query performance is affected if the threshold is too low.

The `trace_report` log is located in `<Domain Home>/servers/<Essbase-Managed-Server-Name>/logs/essbase/essbase/app/<application-name>/<cube-name>/trace_report.log`.

The application ODL log is located in `<Domain Home>/servers/<Essbase-Managed-Server-Name>/logs/essbase/essbase/app/<application-name>/<application-name>_ODL.log`.

For details about log file locations, refer to Environment Locations in the Essbase Platform.

When to Use Different Query Tracing Options

The QUERYTRACE configuration setting is designed for debugging a single query, in case any problems are observed. It prints the trace log to the cube directory as `query_trace.txt`, and the file is cleared by default before each query execution. QUERYTRACE should not be left enabled in the application configuration for long term use, as it may affect performance.

The TRACE_REPORT configuration setting provides fewer details than QUERYTRACE, but is able to trace multiple, concurrent queries. TRACE_REPORT is a good option for gathering information on many concurrent queries running over a period of time.

Enabling query logging by setting the QUERYLOG parameter in `dbname.cfg` (in the cube directory) is designed for tracking user query patterns for a cube, in XML format.

See Also

[TRACE_REPORT](#)

[QUERYTRACETHRESHOLD](#)

[QUERYTRACE](#)

LONGREQTIMETHRESHOLD

The LONGREQTIMETHRESHOLD configuration setting enables you to log informational messages for long-running requests sent to the Essbase application. Specify the upper-limit request execution time, in seconds, before which you want to enable tracking.

Syntax

```
LONGREQTIMETHRESHOLD n
```

where *n* is the upper-limit execution time, in seconds, before which Essbase tracks the requests and logs alert messages. Default: 0 seconds (request tracking is turned off). Minimum: 60 seconds (if using request tracking).

Description

The logged messages include the following information.

- **REQ_ID**: a unique number used to identify a request
- **REQ_INFO**: a string describing the type of request (for example, Calculation, Data-load)
- **REQ_STATE**: a string describing the current state of the request when this message is logged (for example, **started**, **in_progress**, or **finished**)
- **EXEC_TIME**: time, in milliseconds, that the request has been running
- **LEVEL0_BLOCKS**: number of level 0 blocks at the time the message is logged. Essbase only logs this information for requests that could potentially change the cube.
- **TOTAL_BLOCKS**: number of total blocks at the time the message is logged. Essbase only logs this information for requests that could potentially change the cube.

Example

```
LONGREQTIMETHRESHOLD 600
```

Specifies message logging for requests running 600 seconds or longer.

The following request-pending message in the log shows that request [REQ_ID: 109] has been running for 650232 milliseconds. Therefore, it is considered a long-running request.

```
[2020-01-18T17:31:03.667-08:00] [Sample] [NOTIFICATION:16] [REQ-451] [REQ]
[ecid: 1484789397502,0] [tid: 9664]
[DBNAME: Basic] [REQ_ID: 109] [REQ_INFO: Calculate] [REQ_STATE: in_progress]
[EXEC_TIME: 650232]
```

LROONSHAREDMBR

The LROONSHAREDMBR configuration setting specifies whether Essbase shared members have Linked Reporting Objects that are unique from those of their prototype members.

LROONSHAREDMBR configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
LROONSHAREDMBR TRUE | FALSE
```

- **TRUE**—LROs related to regular members are unique, and not shared by shared members. This is the default.

- FALSE—Shared members have the same LROs as their prototype members.

Description

A Linked Reporting Object (LRO) is an external file, cell note, or URL that you link to a cell in a cube. Users can then retrieve the object from their queries in Smart View or other grid clients.

With an LROONSHAREDMBR setting of TRUE, Essbase makes shared member LROs unique from the LROs of prototype members.

For example, assume the LROONSHAREDMBR option is FALSE. If you link an LRO to the data cell related to Diet Colas (100-20) under the parent member Colas (100), the corresponding data cell for Diet Colas (100-20) under the parent member Diet shares the same LRO.

Example

```
LROONSHAREDMBR FALSE
```

MAXDATE

Specify the MAXDATE configuration setting to define the upper limit for date measures you can export and import. This configuration is applicable only to Essbase cubes that have typed measures enabled and have date measures defined.

Syntax

```
MAXDATE [appname [dbname]] dateConst
```

appname—The application for which you want to specify max date.

dbname—The database (cube) for which you want to specify a max date.

dateConst—A constant in format of YYYY-MM-DD representing the latest possible date recognized by Essbase. The default value is 3000-12-31. The highest allowed value is the same as the default, on Windows, or 9999-12-31 on Linux.

Description

There are several cases where the value of a date measure can become invalid:

- If multiple dates are loaded into the same cell with the Add load option. See Adding to and Subtracting from Existing Values.
- If a non-base member of a hierarchy summarizes values of its date-type children.
- If a data set imported into an Essbase database includes incorrect date values, or date values not supported by the operating system.

To avoid encountering invalid date measures, Oracle recommends that once you choose a value for *dateConst* (or accept the default), you do not make changes to it. Changing the constant value could render old, exported data unusable.

Notes

- If any date is less than 1970-01-01 or greater than *dateConst*, it is considered out of range.
- Data loads will fail if a date measure value exceeds the MAXDATE *dateConst* value with an appropriate message. For example, one of the messages you may see is the following:

Date measure value [#Txt:2040-05-15] is out of range. Please make sure the date value is valid. [3] records Completed

- During data export, date values that are out of range will be exported as "#Txt:OutOfRange". The exported data file with "#Txt:OutOfRange" strings can be reimported to Essbase if you want to load out of range values to the date measures.
- The MAXDATE format does not have any correlation with the date format of the outline. Regardless of the date format of the outline, the format of *dateConst* should always be specified in the format of YYYY-MM-DD.
- The value of *dateConst* is printed to the application log. If you specify it incorrectly, a warning is logged, and it is set to its default value.
- Dates that have a stored internal value resulting in year with more than 4 digits are displayed as out of range and exported as out of range.

Example

```
MAXDATE 3000-12-31
```

See Also

Working with Typed Measures

MAXERRORMBRVERIFYREPORT

The MAXERRORMBRVERIFYREPORT configuration setting enables you to set the maximum number of members on which Essbase should report errors during outline verification. Setting a limit helps avoid performance overhead when a large number of members may cause outline verification errors.

Syntax

```
MAXERRORMBRVERIFYREPORT n
```

n—Specifies the number of members on which to report outline validation errors. The default is 500 members.

Example

```
MAXERRORMBRVERIFYREPORT 25
```

MAXFORMULACACHESIZE

The MAXFORMULACACHESIZE configuration setting enables you to set the maximum size of the formula cache Essbase should make available for calculating members with formulas. You can use this setting to limit how much memory may be used by a single query.

MAXFORMULACACHESIZE configuration applies to aggregate storage (ASO) databases and block storage (BSO) databases in hybrid mode.

Syntax

```
MAXFORMULACACHESIZE [appname [dbname]] n
```

- *appname*—Optional. To set the cache size maximum for a specific application, specify the application name.
- *dbname*—Optional. To set the cache size maximum for a specific database (cube), specify the cube name. If *dbname* is specified, *appname* must also be specified.
- *n*—An integer that specifies the number of kilobytes (KB) to set as the maximum cache size to be made available for calculating members with formulas. The default is 1024 KB for aggregate storage cubes, and 102400 KB for block storage cubes in hybrid mode.

Description

If the amount of cache Essbase sets aside for calculating outline members is insufficient, Essbase switches its formula cache mechanism to allocate only existing and temporary cache values. If this occurs and query tracing is enabled, the following message is logged in `query_trace.txt`:

```
Max formula cache size overflow. Calculation cache will use map structure for
values. If query performance is insufficient try to extend
MAXFORMULACACHESIZE over this limit: n
```

The following guidelines can help you determine what value to use for *n*:

1. Identify which queried dimensions are represented by dynamic members.
2. Multiply the sizes of those dimensions to get a number of members.
3. Multiply the number of members by 8 to get the recommended *n* value (not more than 4G).

For example, the default formula cache size of 1024 allows $1024/8=128$ members to be in the cache.

Notes

- This setting is only relevant if your query references at least one dynamic member with a formula, or if your MDX query has a calculated member in the WITH section.
- This cache is allocated per calculation thread. Concurrent MDX requests can be allocated multiple cache objects, each with a maximum size specified in `MAXFORMULACACHESIZE`.
- The entire specified amount is not used unless needed.
- The memory is released after the query completes.

Example

```
MAXFORMULACACHESIZE 2048
```

Sets the formula cache size maximum to 2048 KB for every application and database.

See Also

[QUERYTRACE](#)

[TRACE_REPORT](#)

MAXLOGINS

The MAXLOGINS configuration setting enables you to limit the number of user sessions that can be connected to the Essbase Server at any one time, including multiple instances of the same user.

Syntax

```
MAXLOGINS n
```

n—Any integer from 1000 to 1048575 is valid. The default value is 10000.

Description

You may wish to adjust the value of MAXLOGINS to match computer resources, or to more closely manage concurrent ports and user sessions. A concurrent port is used for each unique combination of client machine, Essbase Server and login name. For example, the same user with five open Excel worksheets connected to the same Essbase Server use one port, but five sessions.

Notes

- Increasing the value of MAXLOGINS increases memory use approximately 6 bytes per user session.
- If the setting is less than the minimum value, 1000, the value is assumed to be 1000.

Example

```
MAXLOGINS 50000
```

Increases the maximum number of simultaneous logins possible, from the default of 10000 to 50000.

MAXNUMBEROFACTIVEDB

The MAXNUMBEROFACTIVEDB configuration setting specifies the maximum number of active Essbase databases that can be accessed concurrently. If the maximum number of active databases is exceeded, the next database does not start.

Syntax

```
MAXNUMBEROFACTIVEDB n
```

n—Specifies the maximum number of databases that can be accessed concurrently.

A value of 0 means that there is no maximum limit. The default value is 0.

Example

```
MAXNUMBEROFACTIVEDB 10
```

Specifies that 10 databases can be active.

MAX_REQUEST_GRID_SIZE

Sometimes you need to limit the size of Essbase query grids flowing from a block storage (BSO) replicated partition source cube to an aggregate storage (ASO) replicated partition target cube.

You can use the Essbase configuration setting `MAX_REQUEST_GRID_SIZE` to limit the request grid size (the number of cells requested from a source cube by a user on an ASO target cube). Limiting the size of the request grid, which can be millions of cells, improves response time.

If you find that you must set a small request grid size, you should look into improving the design of the application.

Syntax

```
MAX_REQUEST_GRID_SIZE [appname [dbname]] n
```

- *appname*—Optional. Specifies the application for which the request grid size is to be set.
If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application.
To enable the setting for a specific database, you must specify an application and database.
If you do not specify an application, you cannot specify a database, and the setting applies to all applications and databases on Essbase Server.
- *dbname*—Optional. Specifies the database, in the application specified by *appname*, for which the request grid size is to be set.
If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored, and logging diagnostic messages is enabled for all applications and databases on Essbase Server.
- *n*—Specifies the size of the request grid to be returned from the data source.
The default value is 10 million (10000000) cells.
The maximum value is limited by the unsigned int value of 4294967295.

You must restart Essbase Server to initialize any change to the configuration file.

Example

```
MAX_REQUEST_GRID_SIZE ASOSamp 5000000
```

Limits the request grid to 5 million cells for all databases associated with the ASOSamp application.

See Also

[MAX_RESPONSE_GRID_SIZEg](#)

MAX_RESPONSE_GRID_SIZE

Sometimes you need to limit the size of Essbase query grids flowing from a block storage (BSO) replicated partition source cube to an aggregate storage (ASO) replicated partition target cube. You can use MAX_REQUEST_GRID_SIZE to limit the total request grid size.

The MAX_RESPONSE_GRID_SIZE configuration setting enables you to also limit the maximum response grid size that can be sent to the ASO target in one operation. Essbase splits the request grid into slices of data and sends multiple, smaller response grids to the target.

For example, if MAX_REQUEST_GRID_SIZE is set to one billion (1000000000) cells and MAX_RESPONSE_GRID_SIZE is set to one million (1000000) cells: $1000000000 / 1000000 = 1000$. Essbase sends response grids to the ASO target in multiple slices of one thousand (1000) cells.

The amount of memory required to temporarily hold the response grid in the data target is proportional to the size of the request grid (MAX_REQUEST_GRID_SIZE). In the case of a huge request grid with millions of cells, the amount of memory required for the response grid to be sent in one operation could pose problems (for example, the system could reach memory boundaries or fail to allocate enough memory).

Syntax

```
MAX_RESPONSE_GRID_SIZE [appname [dbname]] n
```

- *appname*—Optional. Specifies the application for which the response grid size is to be set. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application. To enable the setting for a specific database, you must specify an application and database. If you do not specify an application, you cannot specify a database, and the setting applies to all applications and databases on the Essbase Server.
- *dbname*—Optional. Specifies the database, in the application specified by *appname*, for which the response grid size is to be set. If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored, and logging diagnostic messages is enabled for all applications and databases on the Essbase Server.
- *n*—Specifies the size of the slice of the response grid to be sent to the data target. The default value is one million (1000000) cells, which requires 8 MB of memory.

You must restart Essbase Server to initialize any change to the configuration.

Example

```
MAX_RESPONSE_GRID_SIZE ASOSamp 500000
```

Limits the response grid to a half-million cells (which requires 4 MB of memory) for all databases associated with the ASOSamp application.

See Also[MAX_REQUEST_GRID_SIZE](#)

MAX_SIZE_PER_FETCH

The `MAX_SIZE_PER_FETCH` configuration setting enables you to set the maximum size of the grid after expansion. When using this setting, `GRIDEXPANSION` must be set to `ON`. For Essbase cubes that are the target of a transparent partition, Oracle recommends a smaller maximum grid size, to retain the advantages of grid expansion.

`MAX_SIZE_PER_FETCH` configuration applies only to non-hybrid mode, block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
MAX_SIZE_PER_FETCH [appname [dbname]] n
```

- *appname*—Optional. Specifies the application for which to set the limit. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application. To enable the setting for a specific cube, you must specify an application and cube.
- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which to set the limit. If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- *n*—The maximum number of cells in the grid after grid expansion. The default is 102400000.

Description

`MAX_SIZE_PER_FETCH` specifies the maximum size of the grid after grid expansion.

If, after grid expansion, the size of the grid is greater than the maximum size specified, grid expansion will not occur for the query, which might result in a slight degradation of performance.

Example

```
MAX_SIZE_PER_FETCH Sample Basic 75000000
```

Limits the grid to 75 million cells after grid expansion for each query to the Basic cube associated with the Sample application.

See Also[GRIDEXPANSION](#)

MDXINSERTBUFFERAGGMETHOD

When you perform an MDX Insert, Essbase creates an output buffer in memory, which accumulates with values until the Insert is completed.

The MDXINSERTBUFFERAGGMETHOD configuration setting enables you to control resolution of conflicts in the buffer. The method you use can impact the results of the Insert. By default, duplicate cell values are summed together, but you can configure Essbase to use the value of the cell that was loaded last.

Syntax

```
MDXINSERTBUFFERAGGMETHOD ADD | LAST
```

- **LAST**—If, during an MDX Insert operation, a value needs to be written to an output buffer location that already contains a value, the latest value overwrites the older value.
- **ADD**—If, during an MDX Insert operation, a value needs to be written to an output buffer location that already contains a value, the latest value is summed with the older value. This is the default behavior.

Description

During execution of an MDX Insert query, an output buffer is created in memory which accumulates with values until the query is completed. MDXINSERTBUFFERAGGMETHOD enables you to define the method with which values are aggregated in the output buffer. The method that you use can have an effect on the data results of the MDX Insert operation.

Assume that in an MDX Insert query, two source tuples are mapped to a single target tuple, as shown:

```
INSERT
([Payroll], [Jan]) TO ([Revised Payroll], [Jan])
([Payroll], [Feb]) TO ([Revised Payroll], [Jan])
...
```

Assume that the value of ([Payroll], [Jan]) is 100, and the value of ([Payroll], [Feb]) is 200.

Using the default buffer aggregation behavior (ADD),

```
MDXINSERTBUFFERAGGMETHOD ADD
```

1. The value for ([Payroll], [Jan]) is written to the output buffer for ([Revised Payroll], [Jan]), making its value 100.
2. The value for ([Payroll], [Feb]) is added to the same output buffer for ([Revised Payroll], [Jan]), increasing its value to 300.

If you change the buffer aggregation behavior to LAST,

```
MDXINSERTBUFFERAGGMETHOD LAST
```

1. The value for ([Payroll], [Jan]) is written to the output buffer for ([Revised Payroll], [Jan]), making its value 100.
2. The value for ([Payroll], [Feb]) is written to the same output buffer for ([Revised Payroll], [Jan]), overwriting the previous value, and changing it to 200.

See Also

MDX Insert

MDXINSERTREQUESTTIMEOUT

The MDXINSERTREQUESTTIMEOUT configuration setting enables you to set a timeout for MDX Insert requests to the Essbase database.

Syntax

```
MDXINSERTREQUESTTIMEOUT n
```

where *n* is the number of seconds the MDX Insert request is permitted to run before timing out. The default is -1, meaning there is no timeout.

Example

```
MDXINSERTREQUESTTIMEOUT 240
```

Sets the timeout for MDX Insert requests at four minutes.

See Also

MDX Insert

MDXLIMITFORMULARECURSION

You can use the MDXLIMITFORMULARECURSION configuration setting to remove Essbase Server limitations on the number of execution levels allowed for MDX calculated members or formulas.

When set to false, MDXLIMITFORMULARECURSION does not prevent Essbase Server from exceeding 128 MDX formula execution levels. You can use this if you know that a recursive execution in a formula/calculated member will eventually terminate, and you want Essbase to continue processing it.

Syntax

```
MDXLIMITFORMULARECURSION [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the application for which to set or remove the limit. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application. To enable the setting for a specific cube, you must specify an application and cube.
- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which to set the limit. If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- TRUE—Imposes a limit of 128 on the number of MDX formula execution levels. The default setting is TRUE.
- FALSE—Imposes no limit on the number of MDX formula execution levels.

Description

MDX calculated member or formula execution may be recursive (for example, a formula can refer to itself, or a calculated member can refer to itself). By default, Essbase limits the number of MDX formula execution levels, because formulas with excessive execution levels may lead to stack overflow errors and crash the server. However, setting MDXLIMITFORMULARECURSION to FALSE prevents Essbase from imposing the limitation. You can use this setting when you know that a recursive execution in a formula/calculated member will eventually terminate, and you wish to have a recursion depth greater than 128.

If an MDX formula reaches 128 execution levels and MDXLIMITFORMULARECURSION is not set, or is set to TRUE, Essbase stops processing that formula and writes error messages in the application log. If a formula reaches 128 execution levels and MDXLIMITFORMULARECURSION is set to FALSE, Essbase continues processing that formula.

Caution:

Before setting MDXLIMITFORMULARECURSION to FALSE, be sure that the MDX formulas in the outline are not infinitely recursive; for example, be sure that formulas do not depend on each other.

MDXQRYGOVCOUNT

The MDXQRYGOVCOUNT configuration setting enables you to control how often Essbase checks for conditions that would warrant termination of an MDX query.

Use MDXQRYGOVCOUNT to reduce or increase the default number of MDX query checks. Reducing the number of checks (by setting *n* higher) improves performance. The counter starts at *n* and decrements until the counter reaches zero: at that time Essbase performs a check.

Syntax

```
MDXQRYGOVCOUNT [appname [dbname]] n
```

- *appname*—Optional. Specifies the application for which to apply the checking counter. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application. To enable the setting for a specific database, you must specify an application and database.
- *dbname*—Optional. Specifies the database, in the application specified by *appname*, for which to apply the checking counter. If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- *n*—Integer specifying the counter (number of check conditions) that Essbase checks for conditions that warrant query termination. You must specify this parameter or Essbase ignores the MDXQRYGOVCOUNT setting. If do not specify *appname* or *dbname*, the counter applies to the entire Essbase Server. The default value is 1000. The minimum value is 100, and the maximum value is 5000.

 **Note:**

You can use the **Esc** key to cancel any query running from the MaxL client.

Example

```
MDXQRYGOVCOUNT 1500
```

See Also

[QRYGOVEXECTIME](#)

[QRYGOVEXECLBK](#)

MEMORYMAPPEDDATA

When ORACLEHARDWAREACCELERATION is enabled, the MEMORYMAPPEDDATA configuration setting enables performance improvements for Essbase block storage (BSO) databases.

Areas in which performance is improved include data load, serial and parallel calculation, export, and restructuring, especially for remote storage (NFS). Setting MEMORYMAPPEDDATA to TRUE improves in-memory computing of Essbase block storage databases by using memory mapped I/O for page files.

 **Note:**

Virtual memory usage will be equal to the size of the sum of page files in the database.

Syntax

```
MEMORYMAPPEDDATA [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the application for which to enable memory mapped data. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application. To enable the setting for a specific database, you must specify an application and cube.
- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which to enable memory mapped data. If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- TRUE—Enables memory mapped data.
- FALSE—Disables memory mapped data. This is the default.

Example

```
MEMORYMAPPEDDATA Sample Basic TRUE
```

See Also[ORACLEHARDWAREACCELERATION](#)

METADATABASEDAGGVIEWSBUILD

The METADATABASEDAGGVIEWSBUILD configuration setting enables you to select whether Essbase automatically maintains default views for aggregate storage (ASO) cubes, and to select the view selection algorithm Essbase should use.

In releases prior to Essbase 19c, Essbase created default aggregate views by using internal analysis based on data sampling. Starting in Essbase 19c, default view selection can be based on metadata analysis. In some cases, this algorithm is better for query performance.

Note: In Essbase 21c or later, METADATABASEDAGGVIEWSBUILD replaces [DEFAULTVIEWBUILD](#).

Syntax

```
METADATABASEDAGGVIEWSBUILD appname OFF | ON | AUTO
```

- *appname*—Application name. The configuration applies to all cubes within the named application. If unspecified, the configuration applies to all aggregate storage cubes on the Essbase Server.
- OFF—This is the default. Creation and maintenance of default aggregate views are not automated. Instead, you select, build, and maintain the aggregate views yourself. For its view selection algorithm, Essbase uses internal analysis based on data sampling.
- ON—Creation of default aggregate views is not automated. For its view selection algorithm, Essbase uses database metadata analysis.
- AUTO—Essbase performs automatic default aggregate view selection, build, and maintenance, meaning that default aggregate view selection and materialization are recreated automatically in response to the following qualifying cube events:
 - data is updated (loaded, calculated, allocated, deleted, or submitted from Smart View)
 - there is a metadata change that would cause an update of data indexes, requiring a redesign or rebuilding of aggregate views

The AUTO setting is not supported if your cube uses custom calculations and allocations.

For its view selection algorithm, Essbase uses metadata analysis.

Use [DEFAULTVIEWBUILDSIZE](#) to limit the growth of cubes as a result of automatic views maintenance.

Example

```
METADATABASEDAGGVIEWSBUILD AUTO
```

See Also

[DEFAULTVIEWBUILDSIZE](#) to limit the growth of cubes as a result of automatic views maintenance

Generate Aggregate Views Automatically

Optimization for Aggregate View Selection

MULTIPLEBITMAPMEMCHECK

The MULTIPLEBITMAPMEMCHECK configuration setting enforces the size limit for the amount of memory that is used for the calculator cache, when Essbase selects the multiple bitmap cache option.

MULTIPLEBITMAPMEMCHECK configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
MULTIPLEBITMAPMEMCHECK TRUE | FALSE
```

- TRUE—The calculator cache size limit is enforced in multiple bitmap mode.
- FALSE—The calculator cache size limit is not enforced in multiple bitmap mode. This is the default.

Description

If MULTIPLEBITMAPMEMCHECK is set to TRUE, then any time the memory limit is exceeded for the calculator cache in multiple bitmap mode, Essbase switches to single bitmap mode and enforces the size limit that you selected.

If the MULTIPLEBITMAPMEMCHECK setting is not present or has any other value than TRUE, then the limit is not strictly enforced, and your server process may grow too large.

Example

```
MULTIPLEBITMAPMEMCHECK TRUE
```

See Also

[CALCCACHE](#)

[PARCALCMULTIPLEBITMAPMEMOPT](#)

NETBINDRETRYDELAY

The NETBINDRETRYDELAY configuration setting specifies the amount of time, in milliseconds, before the Essbase application server retries after a network bind failure. Do not change NETBINDRETRYDELAY unless advised by Oracle Support.

Syntax

```
NETBINDRETRYDELAY n
```

n—An integer value, expressed in milliseconds. The default value is 10 seconds. The minimum value is 0.

Description

NETBINDRETRYDELAY configuration is applicable to networking when the ESSNET protocol is in use.

⚠ WARNING:

Oracle does not recommend changing this setting unless advised by Oracle Support, as it could influence other network protocols in use.

Example

```
NETBINDRETRYDELAY 5
```

Causes the application server network to retry on a bind failure after 5 milliseconds.

See Also

[NETDELAY](#)

[NETRETRYCOUNT](#)

[NETTCPCONNECTRETRYCOUNT](#)

NETDELAY

The NETDELAY configuration setting specifies the network request delay time in milliseconds. This is the amount of time an unsuccessful operation waits before Essbase retries the operation. Do not change NETDELAY unless advised by Oracle Support.

Syntax

```
NETDELAY n
```

n—Number of milliseconds to wait before Essbase retries the operation. Must be an integer value of 100 or higher. The default value is 200 milliseconds.

Description

NETDELAY configuration is applicable to networking when the ESSNET protocol is in use.

⚠ WARNING:

Oracle does not recommend changing this setting unless advised by Oracle Support, as it could influence other network protocols in use.

Example

```
NETDELAY 500
```

See Also

[NETBINDRETRYDELAY](#)

[NETRETRYCOUNT](#)

[NETTCPCONNECTRETRYCOUNT](#)

NETRETRYCOUNT

The NETRETRYCOUNT configuration setting specifies the number of times Essbase is allowed to retry a network connection before failing and reporting an error. Do not change NETRETRYCOUNT unless advised by Oracle Support.

Syntax

```
NETRETRYCOUNT n
```

n—Number of retries. Must be an integer value of 300 or higher. The default value is 500 retries.

Description

NETRETRYCOUNT configuration is applicable to networking when the ESSNET protocol is in use.

 **WARNING:**

Oracle does not recommend changing this setting unless advised by Oracle Support, as it could influence other network protocols in use.

Example

```
NETRETRYCOUNT 400
```

See Also

[NETBINDRETRYDELAY](#)

[NETDELAY](#)

[NETTCPCONNECTRETRYCOUNT](#)

NETSSLHANDSHAKETIMEOUT

The NETSSLHANDSHAKETIMEOUT configuration setting specifies the maximum time that Essbase clients can wait for a response to a secure session request before timing out. Do not change NETSSLHANDSHAKETIMEOUT unless advised by Oracle Support.

Syntax

```
NETSSLHANDSHAKETIMEOUT n
```

n—The number of milliseconds to wait. Must be a positive integer. The default is 10000 milliseconds (10 seconds). The minimum value is 100 milliseconds. Values less than the minimum are ignored.

Description

NETSSLHANDSHAKETIMEOUT configuration is applicable to networking when the ESSNET protocol is in use.

WARNING:

Oracle does not recommend changing this setting unless advised by Oracle Support, as it could influence other network protocols in use.

Notes

The secure handshake may timeout due to network congestion, or because the connection modes at either end are mismatched (for example, a client in clear mode tries to connect to the secure port of Essbase Agent by mistake).

Example

```
NETSSLHANDSHAKETIMEOUT 20000
```

The TLS/SSL handshake fails after 20,000 milliseconds if Essbase Agent does not respond to the secure session request.

See Also

[AGENTSECUREPORT](#)

[CLIENTPREFERREDMODE](#)

[ENABLECLEARMODE](#)

[ENABLESECUREMODE](#)

[SSLCIPHERSUITES](#)

[WALLETPATH](#)

For information on implementing secure mode (TLS), see *About Securing Your Communication and Network*.

NETTCPCONNECTRETRYCOUNT

The NETTCPCONNECTRETRYCOUNT configuration setting specifies the number of times an Essbase client can attempt to connect to a TCP/IP network before failing and reporting an error. Do not change NETTCPCONNECTRETRYCOUNT unless advised by Oracle Support.

Syntax

```
NETTCPCONNECTRETRYCOUNT n
```

n—Number of connection attempts. Must be an integer value between 2 and 1000000. The default value is 3.

Description

NETTCPCONNECTRETRYCOUNT configuration is applicable to networking when the ESSNET protocol is in use.

WARNING:

Oracle does not recommend changing this setting unless advised by Oracle Support, as it could influence other network protocols in use.

Notes

Some causes of connection failures are network congestion, server inaccessibility, and network interruption.

Example

```
NETTCPCONNECTRETRYCOUNT 100
```

See Also

[NETRETRYCOUNT](#)

[NETDELAY](#)

[NETBINDRETRYDELAY](#)

NUMBLOCKSTOEXTEND

The NUMBLOCKSTOEXTEND configuration setting enables you to control extension of the data file in Essbase block storage (BSO) databases, to accommodate block updates that require additional disk space.

Syntax

```
NUMBLOCKSTOEXTEND [appname [dbname]] n
```

The product of *n* and the currently requested block size is the number of bytes by which the data file is extended.

The default value is 2,048.

Description

When the Essbase BSO kernel updates a block, it writes to a new disk location. The kernel searches free space to find a new disk location to use. If there is not enough free space to service the current request, the data file is extended.

**Note:**

Upon first upgrading to this release, there is an increase in the amount of disk space pre-allocated for page files unless you set NUMBLOCKSTOEXTEND to 1.

Example

```
NUMBLOCKSTOEXTEND Sample Basic 2240
```

NUMERICPRECISION

The NUMERICPRECISION configuration setting for Essbase enables you to set the number of precision digits used by Report Writer for numerical comparison.

Syntax

```
NUMERICPRECISION n
```

n—The number of precision digits to be considered in the numerical comparison. Acceptable values for *n* are -1 through 15. A value of -1 indicates a full comparison. The default value is 4.

Description

This setting defines the number of precision digits used by Report Writer for numerical comparison.

The numeric comparison function subtracts one value from the other, and compares the absolute value of the result with 10^{-n} . If 10^{-n} is greater than the absolute value of the subtraction result, the numbers are equal.

A value of -1 indicates a full comparison.

Example

Compare the values 3.289999 and 3.290000 with a numeric precision of 2:

```
NUMERICPRECISION 2
```

Is $3.289999 == 3.290000$ given a numeric precision of 2?

$| 3.289999 - 3.290000 | = 0.000001$ (the absolute value)

$10^{-2} = 0.01$

$0.01 > 0.000001$, so the numbers are equal.

ODBCERRORLOGOFF

The ODBCERRORLOGOFF configuration setting enables you to log ODBC driver error messages. If you enable it, Essbase writes ODBC messages to the application log file.

Syntax

```
ODBCERRORLOGOFF TRUE | FALSE
```

- TRUE—Logging of ODBC driver errors is disabled.
- FALSE—Logging of ODBC driver errors is enabled. This is the default.

Example

```
ODBCERRORLOGOFF TRUE
```

ORACLEHARDWAREACCELERATION

The ORACLEHARDWAREACCELERATION configuration setting enables optimizations for Essbase Server deployments on the Oracle Exalytics In-Memory machine.

Essbase enhancements take advantage of Exalytics In-Memory machine CPUs, memory, and other aspects of the machine hardware that allows Essbase to deliver improved performance and scalability.

Caution:

This setting must only be used when Essbase is deployed on the Exalytics In-Memory machine. ORACLEHARDWAREACCELERATION is not supported and should never be set to TRUE on deployments of Essbase on non-Exalytics In-Memory machines.

Syntax

```
ORACLEHARDWAREACCELERATION TRUE | FALSE
```

- TRUE—Essbase uses specific Oracle Exalytics In-Memory machine optimizations. Set this setting to TRUE only if Essbase is deployed on an Exalytics In-Memory machine.
- FALSE—This is the default.

Example

```
ORACLEHARDWAREACCELERATION TRUE
```

See Also

[Oracle Exalytics In-Memory Machine Configuration Settings](#)

OUTLINECHANGELOG

The OUTLINECHANGELOG configuration setting enables you to log the history of Essbase outline modifications. The revision history provides information you can use to roll back changes if needed.

Syntax

```
OUTLINECHANGELOG TRUE | FALSE
```

- TRUE—Essbase logs outline changes into the file *database_name.olg*.
- FALSE—Essbase does not log outline changes. This is the default.

Description

If OUTLINECHANGELOG is set to TRUE, Essbase logs all outline changes into the file *database_name.olg*.

Each database (cube) contains a separate outline change log file in the cube directory.

The data format of the outline change log is:

- Date and time of outline modification
- Name of the user who made the change
- Type of change the user made
- Details describing the type of change made

Notes

- During a restructure, Essbase holds outline change information in memory until all updates have been made to the outline change log. Turning on the outline change log might affect your restructure performance, particularly after dimension builds of several hundred or more members.
- To set the size of the outline change log, use the [OUTLINECHANGELOGFILESIZE](#) parameter.

Example

```
OUTLINECHANGELOG TRUE
```

See Also

[OUTLINECHANGELOGFILESIZE](#)

OUTLINECHANGELOGFILESIZE

If outline change logging is enabled, the OUTLINECHANGELOGFILESIZE configuration setting enables you to set the maximum file size of the Essbase outline change log.

Syntax

```
OUTLINECHANGELOGFILESIZE n
```

n—The number of bytes to allocate for the outline change log. The default is 64,000 bytes. The minimum is 8,092 bytes. The maximum is 2 megabytes.

Description

If `OUTLINECHANGELOG` is enabled, `OUTLINECHANGELOGFILESIZE` sets the maximum file size of the outline change log in bytes. When the outline change log reaches the maximum file size, Essbase copies the contents of the file to a separate backup file with the same name as the outline change log file (`database_name.olg`), but with an `.olb` extension.

Notes

- The outline change log is disabled by default. To enable it, use the [OUTLINECHANGELOG](#) parameter.
- The outline change log file is located in the database (cube) directory of the Essbase Server installation. It is named in the format `cube_name.olg`.
- The default, minimum, and maximum file sizes for the backup file are the same as the file sizes specified for the outline change log file.
- Each time the outline change log file reaches its maximum file size, Essbase clears the outline change log and replaces the backup file with a backup of the current outline change log.

Example

```
OUTLINECHANGELOGFILESIZE 8092
```

See Also

[OUTLINECHANGELOG](#)

PARCALCMULTIPLEBITMAPMEMOPT

The `PARCALCMULTIPLEBITMAPMEMOPT` configuration setting enables you to optimize memory use during parallel calculation, when Essbase is using multiple bitmap mode for the calculator cache.

`PARCALCMULTIPLEBITMAPMEMOPT` configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
PARCALCMULTIPLEBITMAPMEMOPT TRUE | FALSE
```

- `TRUE`—Memory usage is optimized when using multiple bitmap mode during `CALCPARALLEL` parallel calculation.
- `FALSE`—Memory usage is not optimized. This is the default.

Description

If the setting is present and its value is `TRUE`, then Essbase optimizes memory usage when using parallel calculation in calculator cache multiple bitmap mode. This setting can be used together with, or separately from, [MULTIPLEBITMAPMEMCHECK](#).

Example

```
PARCALCMULTIPLEBITMAPMEMOPT TRUE
```

See Also[CALCCACHE](#)[CALCPARALLEL](#)[MULTIPLEBITMAPMEMCHECK](#)

PIPEBUFFERSIZE

The PIPEBUFFERSIZE configuration setting enables you to decrease size of the buffer used for communication between the Essbase grid extractor and Report Writer on the network.

Syntax

```
PIPEBUFFERSIZE n
```

n—An integer value from 2,048 to 65,534, expressed in bytes. The default is 65000 (about 65K).

Example

```
PIPEBUFFERSIZE 20000
```

For the application Sample, defines a 20-kilobyte buffer to store pipes.

PREFIXID

The PREFIXID configuration setting enables you to change the prefix in automatically generated Essbase member IDs to a prefix of your choice.

Syntax

```
PREFIXID [<appname> [<dbname>]] <prefix>
```

Description

A member ID is a permanent, unique identifier for a member, separate from its name. All members in a cube's outline have a member ID. When Essbase auto generates member IDs, the IDs follow an incremental naming pattern: `id__0`, `id__1`, `id__2`, etc.

In rare cases, it may be necessary to change the prefix to ensure that it is unique. If the auto generated member ID is already in use by another member name or alias, you can change it, using PREFIXID.

Example

```
PREFIXID Sample Basic memberid
```

See Also

Member IDs

QRYGOVEXECBLK

The QRYGOVEXECBLK configuration setting enables you to implement a size-based query governor for block storage (BSO) databases. Use it if you want to limit the number of blocks that a query can retrieve before the Essbase Server terminates the query.

QRYGOVEXECBLK configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
QRYGOVEXECBLK [appname [dbname]] n
```

- *appname*—Optional. Applies the query block limit to the application specified. If you specify *appname*, you must also specify a value for *n*, or Essbase Server ignores QRYGOVEXECBLK. If you do not specify an application, you cannot specify a cube, and the query block limit applies to all applications and cubes on the server. If you specify a value for *appname* and do not specify a value for *dbname*, the query time limit applies to all cubes in the specified application.
- *dbname*—Optional. Must be used with *appname* and *n*, or Essbase Server ignores QRYGOVEXECBLK. If you specify *dbname*, *appname*, and *n*, the query block limit is applied only to the specified cube.
- *n*—The number of blocks that Essbase Server allows a query to access before the query is terminated. You must specify this parameter or the server ignores QRYGOVEXECBLK. The default is unlimited (no limits are imposed on block storage queries).

Description

When a query exceeds the QRYGOVEXECBLK block limit and is terminated, Essbase writes an error message to the application log.

Restarting Essbase Server after adding or changing this setting activates the new setting values.

Use QRYGOVEXECBLK to prevent these types of queries:

- A long-running query against a cube that accesses attributes at a high level, forcing many dynamic calculations to occur.
- A query that uses the zoom-in "Drill to bottom" option in a large dimension.
- A query that uses the zoom-in "Drill to all levels" option in a large dimension.

Use QRYGOVEXECBLK, for example, if you have users who try to retrieve so much data in a single query that their query appears to hang for minutes at a time. A query launched against the cube involving attribute dimensions, for example, may be larger than the user realizes.

Notes

- If you use an invalid value (such as a negative number, a letter, a word, or a special character) for *n*, Essbase Server ignores QRYGOVEXECBLK.

- Query governor settings are ignored during data load and calculation. You can leave query governor settings in the configuration whether you are performing these operations or querying against the data.

Example

QRYGOVEXECBLK Sample Basic 3

Sets three blocks as the maximum number of blocks that a query to Sample Basic can access before being terminated. A block is created for each unique combination of sparse dimension members. If a user issues a query that accesses four unique combinations of sparse dimensions, Essbase Server terminates the query and writes a message to the application log.

QRYGOVEXECBLK 5

Sets five blocks as the maximum number of blocks that a query can access before being terminated. The query time limit applies to all applications and cubes on Essbase Server .

See Also

[QRYGOVEXECTIME](#)

QRYGOVEXECTIME

The QRYGOVEXECTIME configuration setting enables you to implement a time-based query governor. Use it if you want to limit the number seconds that a query can run before the Essbase Server terminates the query.

Syntax

```
QRYGOVEXECTIME [appname [dbname]] n
```

- *appname*—Optional. Applies the query time limit to the application specified. If you do not specify an application, then you cannot specify a cube, and the query time limit applies to all applications and cubes on Essbase Server. If you specify a value for *appname* and do not specify a value for *dbname*, the query time limit applies to all cubes in the specified application.
- *dbname*—Optional. Must be used with *appname* and *n*, or Essbase Server ignores QRYGOVEXECTIME. If you specify *dbname*, *appname*, and *n*, the query time limit is applied only to the specified cube.
- *n*—Integer specifying the number of seconds that Essbase Server allows a query to run before the query is terminated. By default, there is no limit.

Description

QRYGOVEXECTIME specifies the maximum amount of time that a query can run before Essbase Server terminates it.

When a query exceeds the time limit and is terminated, an error message is written to the application log.

Restarting Essbase Server after adding or changing this setting activates the new setting values.

Use QRYGOVEXEETIME to prevent these types of queries:

- A long-running query against a cube that accesses attributes at a high level, forcing many dynamic calculations to occur.
- A query that uses the "Drill to bottom" option in a large dimension.
- A query that uses the "Drill to all levels" option in a large dimension.

Use QRYGOVEXEETIME, for example, if you have users who try to retrieve so much data in a single query that their query appears to hang for minutes at a time.

Notes

- Because the query time setting is evaluated in 10 second increments, the query may actually run nine seconds longer than specified before being terminated.
- If you use an invalid value (such as a negative number, a letter, a word, or a special character) for *n*, the server ignores QRYGOVEXEETIME.
- Query governor settings are ignored during data load and calculation. You can leave query governor settings in the configuration whether you are performing these operations or querying against the data.

Example

```
QRYGOVEXEETIME Sample Basic 20
```

Sets 20 seconds as the maximum time that a query can run before being terminated. In this example the restriction applies only to the Basic cube in the Sample application.

```
QRYGOVEXEETIME 45
```

Sets 45 seconds as the maximum time that a query can run before being terminated. The query time limit applies to all applications and cubes on the server.

See Also

[QRYGOVEXECBLK](#)

QUERYBOTTOMUP

Use the QUERYBOTTOMUP configuration setting to activate bottom-up query processing for Essbase formula calculation. In hybrid mode, this optimizes query times by identifying the required intersections for calculation, making the query time proportional to input data size.

Syntax

```
QUERYBOTTOMUP TRUE | FALSE
```

- **TRUE**—Query execution occurs in bottom-up mode, to resolve dependency analysis quickly in cases where the formula cache has a relatively small input data set.
- **FALSE**— Default option. Query execution is top-down.

Description

To optimize cubes in the application for hybrid mode for faster formula execution, set QUERYBOTTOMUP to TRUE.

This usage is recommended for hybrid mode performance tuning if both of these are true:

- you experience query performance problems
- formulas are complex and contain many cross-dimensional operators or IF/ELSE statements

Complex formulas can cause the formula cache to grow large while also being sparse (having a relatively small input data set). Using QUERYBOTTOMUP TRUE forces query execution to occur in bottom-up mode, to resolve dependency analysis more efficiently in cases where the formula cache is sparse.

Example

```
QUERYBOTTOMUP TRUE
```

See Also

@QUERYBOTTOMUP calculation function

QUERYRESULTLIMIT

The QUERYRESULTLIMIT configuration setting enables you to change the maximum number of result cells that an Essbase MDX query or grid client query can retrieve. Use it to limit the result volume of queries.

Syntax

```
QUERYRESULTLIMIT [appname [dbname]] n
```

appname—Optional. Applies the query result limit to the application specified. If you specify *appname*, you must also specify a value for *n*, or Essbase ignores QUERYRESULTLIMIT. If you do not specify an application, you cannot specify a database, and the query result limit applies to all applications and databases on the server. If you specify a value for *appname* and

do not specify a value for *dbname*, the query time limit applies to all databases in the specified application.

dbname—Optional. Must be used with *appname* and *n*, or the server ignores QUERYRESULTLIMIT. If you specify *dbname*, *appname*, and *n*, the query result limit is applied only to the specified database.

n—An integer value between 0 and 2^{31} specifies the number of query result cells that the server allows a query to return.

The default value is 1000000 (1M).

Description

When a query exceeds the QUERYRESULTLIMIT, Essbase terminates the query and returns an error message.

Use this setting to limit the result volume of queries, and prevent a query from freezing when a very large number of result cells are returned.

QUERYRESULTLIMIT configuration applies to aggregate storage (ASO) databases and block storage (BSO) databases, including those that are in hybrid mode.

Examples

```
QUERYRESULTLIMIT Sample Basic 100000
```

Sets 100,000 cells as the maximum number of results cells returned in a query to the Basic database for the Sample application.

```
QUERYRESULTLIMIT 150000
```

Sets 150,000 cells as the maximum number of cells that a query can return before being terminated. The query result limit applies to all applications and databases on the Essbase Server that corresponds to this configuration.

QUERYTRACE

Use QUERYTRACE to debug performance for a single query. Query tracing helps you monitor Essbase query performance metrics. The level of information you want to keep is reported in the application log and/or a separate tracing log, `query_trace.txt`.

QUERYTRACE configuration applies to block storage databases (including those in hybrid mode), but does not apply to aggregate storage (ASO) databases.

Description

This setting enables query tracing for calculation flows. The query tracing output file includes:

- The input query
- An expanded query odometer
- General information about query calculation units
- A list of formulas and aggregations

- An ordered list of all output cells that are calculated or aggregated during the query, according to solve order

The query tracing output file, `query_trace.txt`, is written to the cube directory.

 **Note:**

Query tracking and query tracing are different.

Query *tracking* enables you to capture user retrieval statistics against an aggregate storage cube, so that Essbase can make view-based optimizations to improve the performance of aggregations. It is on by default. Related MaxL statements include:

```
import query_tracking
export query_tracking
alter database enable query_tracking
query database appname.dbname get cube_size_info
```

Query *tracing* helps you monitor Essbase query performance metrics for block storage cubes (including hybrid mode). It is off by default. If you enable it, Essbase logs metrics in a trace report. Related configuration parameters: TRACE_REPORT, QUERYTRACE, QUERYTRACETHRESHOLD, LONGQUERYTIMETHRESHOLD.

Syntax

```
QUERYTRACE n
```

Where *n* should be set to -1, to enable query tracing. The default value is 0 (query tracing is off).

When to Use Different Query Tracing Options

The QUERYTRACE configuration setting is designed for debugging a single query, in case any problems are observed. It prints the trace log to the cube directory as `query_trace.txt`, and the file is cleared by default before each query execution. QUERYTRACE should not be left enabled in the application configuration for long term use, as it may affect performance.

The TRACE_REPORT configuration setting provides fewer details than QUERYTRACE, but is able to trace multiple, concurrent queries. TRACE_REPORT is a good option for gathering information on many concurrent queries running over a period of time.

Enabling query logging by setting the QUERYLOG parameter in `dbname.cfg` (in the cube directory) is designed for tracking user query patterns for a cube, in XML format.

Example

```
QUERYTRACE -1
```

Sets a tracing query to be run that includes all tracing features listed in Description.
--

See Also[QUERYTRACETHRESHOLD](#)[TRACE_REPORT](#)[LONGQUERYTIMETHRESHOLD](#)

QUERYTRACETHRESHOLD

The QUERYTRACETHRESHOLD configuration setting enables you to set the maximum number of cells (or tuples) that a QUERYTRACE query will log for an Essbase MDX query or other grid client query. Use this setting to limit the number of cells printed for each formula or aggregation number.

Syntax

```
QUERYTRACETHRESHOLD n
```

Where *n* is an integer value between 0 and unlimited, specifying the maximum number of cells to display for each formula calculation path. The default value is 100.

Notes

QUERYTRACETHRESHOLD configuration applies only to block storage (BSO) databases, including those in hybrid mode. It does not apply to aggregate storage (ASO) databases.

Example

```
QUERYTHRESHOLD 50
```

Sets 50 as the maximum number of cells to be displayed for a query that traces the solve order or calculation order.

See Also[QUERYTRACE](#)

REJECTEDRECORDSLIMIT

The REJECTEDRECORDSLIMIT configuration setting enables you to log rejected data load records in the Essbase cube directory.

Syntax

```
REJECTEDRECORDSLIMIT n
```

n—The number of rejected data load records to include in the rejected records log. The default is 0 (logging of rejected records is turned off). The maximum is 65,000 records.

Description

This server-level configuration sets the maximum number of rejected data load records Essbase should log, before stopping.

If this configuration does not exist in `essbase.cfg`, logging rejected data load records is turned off. To keep track of details about rejected data load records, you can turn it on by adding this configuration to `essbase.cfg` and specifying a limit for *n*.

If rejected records are logged, the log entry looks like the following:

```
\\ Record rejected during data load:  
"100-10", "New York", "Feb", "Actual", "645", "258", "90", "51", "1", "#Missing", "619"
```

The following criteria define which data load records are rejected and logged:

- Rejected due to the selection or rejection criteria set in the rule file.
- Rejected because there is no data field in the record.
- Rejected because all data fields are set to be ignored.

Notes

- This configuration setting can be set in `essbase.cfg`, and applies to all applications on the server. Setting this at application level has no effect.
- Essbase logs rejected data load records in `RejectedRecords_SEQID_TimeStamp.log` in the cube directory.

See Environment Locations in the Essbase Platform for information about directory locations in Essbase.

- Data load error logs are handled differently from rejected record logs. Data load errors are logged separately, at the end of the data load. See [DATAERRORLIMIT](#).

Example

```
REJECTEDRECORDSLIMIT 25
```

RENEGADELOG

The `RENEGADELOG` configuration setting enables you to log data values loaded to renegade members in an Essbase aggregate storage (ASO) database.

`RENEGADELOG` configuration applies only to ASO databases, and does not apply to block storage (BSO) databases.

By default, Essbase does not create a log file to track data loaded to renegade members. If `RENEGADELOG` is set to true, Essbase creates a log file in the Essbase logs directory. The log file name is `renDataLoad_<filename>_<timestamp>.log` for non-SQL data loads and `renDataLoad_SQL_<timestamp>.log` for SQL-based data loads.

The log file records the data value loaded to the renegade member. If more than one member in a given data load is missing for a dimension with renegade members enabled, the log file lists only one value. Information on the remaining missing data values is provided in comments.

Syntax

```
RENEGADELOG [appname [dbname]] TRUE | FALSE
```

- *appname*—Application name. Optional parameter for applying the TRUE or FALSE setting to one or all cubes within the application. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application. If you do not specify an application, you cannot specify a cube, and the setting applies to all applications and cubes on the Essbase Server.
- *dbname*—Database (cube) name. Optional parameter for applying the TRUE or FALSE setting to the specified cube within the specified application. If you do not specify a value for *dbname*, the setting applies to all cubes within the specified application. If *appname* is not specified, you cannot specify *dbname*.
- TRUE—Creates a log file to track data loaded to renegade members.
- FALSE—No log file is created. This is the default value.

Example

```
RENEGADELOG TRUE
```

See Also

[RENEGADELOGLIMIT](#)

Renegade Members in Aggregate Storage Data Loads

RENEGADELOGLIMIT

The RENEGADELOGLIMIT configuration setting enables you to specify how many records per data load can be written to the renegade log, when data values are loaded to renegade members in an Essbase aggregate storage (ASO) database.

Syntax

```
RENEGADELOGLIMIT n
```

n—An integer specifying the number of records that can be written to the renegade log. The default value is 100.

RENEGADELOGLIMIT configuration applies only to ASO databases, and does not apply to block storage (BSO) databases.

Example

```
RENEGADELOGLIMIT 350
```

See Also

[RENEGADELOG](#)

Renegade Members in Aggregate Storage Data Loads

REPLAYSECURITYOPTION

The REPLAYSECURITYOPTION configuration setting specifies the user security settings that are used when replaying logged Essbase transactions.

Syntax

```
REPLAYSECURITYOPTION n
```

n—An integer that specifies the user security setting. Valid values are as follows:

- 1—(Default) Specifies the security settings of the user who originally performed the transaction. If that user no longer exists or that user's username was changed, the replay operation will fail.

Oracle does not recommend renaming another user with the name of the original user, as the security settings of the renamed user might not match those of the original user and the transaction might be played with the incorrect security settings.

- 2—Specifies the security settings of the administrator performing the replay operation.
- 3—Specifies the security settings of the user who originally performed the transaction. If that user no longer exists or that user's username was changed, the security settings of the administrator performing the replay operation are used.

You must restart Essbase Server to initialize any change to the configuration.

See Also

Alter Database MaxL statement (with **replay transactions**)

[TRANSACTIONLOGLOCATION](#) configuration setting

[TRANSACTIONLOGDATALOADARCHIVE](#) configuration setting

REPLICATIONASSUMEIDENTICALOUTLINE

The REPLICATIONASSUMEIDENTICALOUTLINE configuration setting optimizes the replication of a partitioned, aggregate storage (ASO) cube when the ASO cube is the target and a block storage cube is the source, and the two Essbase outlines are identical.

REPLICATIONASSUMEIDENTICALOUTLINE configuration affects only the target ASO application, and does not apply to block storage replication.

REPLICATIONASSUMEIDENTICALOUTLINE can be enabled at the server, application, or cube level. You can also use the **alter database** MaxL statement with the **replication_assume_identical_outline** grammar to enable replication optimization at the cube level only.

Syntax

```
REPLICATIONASSUMEIDENTICALOUTLINE [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the application to be enabled for replication optimization. If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.

To enable the setting for a specific cube, you must specify an application and cube.

If you do not specify an application, you cannot specify a cube, and the setting applies to all applications and cubes on Essbase Server.

- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, to be enabled for replication optimization.

If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored, and replication optimization is enabled for all applications and cubes on Essbase Server.

- `TRUE`—Optimize replication, assuming outlines are identical.
- `FALSE`—Do not optimize replication with the assumption of identical outlines. This is the default.

You must restart Essbase Server to initialize changes to the configuration.

Example

```
REPLICATIONASSUMEIDENTICALOUTLINE AsoSamp.Sample TRUE
```

Optimizes the replication of the ASOsamp.Sample database, when it is the target of a replicated partition and its outline is identical to the outline of the source block storage cube.

See Also

Alter Database (Aggregate Storage) MaxL statement

RESTRUCTURETHREADS

The RESTRUCTURETHREADS configuration setting lets you enable parallel restructuring for an Essbase cube, and to specify the number of threads to use.

RESTRUCTURETHREADS configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
RESTRUCTURETHREADS [ appname [ dbname ] ] n
```

- *appname*—Application name. Optional parameter for enabling parallel restructuring for one or all cubes in an application.
- *dbname*—Database (cube) name. Optional parameter for enabling parallel restructuring for an individual cube. This parameter must be used in combination with *appname*.
- *n*—Number of threads to use in parallel restructuring. The default is 1.

Notes

- Use the value `xxxxx` to indicate "all" for any application or cube argument. For example:

```
RESTRUCTURETHREADS xxxxx Basic 2
```

enables parallel restructuring for any application with a Basic cube.
- Settings for nonexistent applications or cubes are ignored.

- Oracle recommends setting an application's RESTRUCTURETHREADS to 2 for most systems (with a maximum of 4), or 4 if the application runs on Exalytics (maximum of 8). Check your calculation and restructure performance after making any changes.

Examples

RESTRUCTURETHREADS Sample 2

Specifies two threads and applies to all databases in the Sample application

RESTRUCTURETHREADS Sample Basic 4

Specifies four threads and applies to the Basic database in the Sample application

See Also

Parallel Restructuring

[WORKERTHREADS](#)

RTDEPCALCOPTIMIZE

The RTDEPCALCOPTIMIZE configuration enables you to make the Essbase @CURRMBRRANGE calculation function behave as runtime dependent. This is useful when @CURGEN or @CURLEV are passed as an argument.

Syntax

```
RTDEPCALCOPTIMIZE [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. If you specify an application name, the setting applies to all databases within the named application. If you do not specify an application name, the setting applies to all applications and databases on the Essbase Server.
- *dbname*—Optional. If you specify a database name and an application name, the setting applies only to the named database. If you do not also specify an application name, the database is ignored and the setting applies to all applications and databases on the Essbase Server.
- TRUE — @CURRMBRRANGE behaves as a non runtime dependent formula. This could result in incorrect calculation results if the @CURGEN or @CURLEV functions are used as arguments to @CURRMBRRANGE, because Essbase would fail to generate the correct dependency list to compute @CURRMBRRANGE. This is the default.
- FALSE — @CURRMBRRANGE behaves as runtime dependent formula, but only when @CURGEN or @CURLEV are passed as an argument to @CURRMBRRANGE. Calculations involving @CURRMBRRANGE may run slowly, as computation of runtime dependent formulas requires more memory.

Example

```
RTDEPCALCOPTIMIZE FALSE
```

SERVERPORTBEGIN

The SERVERPORTBEGIN configuration setting specifies the first port number that Essbase tries to use for its first application process (ESSSVR).

Syntax

```
SERVERPORTBEGIN n
```

n—Specifies the port number that Essbase tries to use for its first application process. This port number should not be in use by any other process. The default value is 32768.

Description

SERVERPORTBEGIN specifies the first port that Essbase tries to use for the first application process it tries to start.

When you configure Essbase using the configuration tool, you select the range of ports to reserve for Essbase application servers. The SERVERPORTBEGIN and SERVERPORTEND configuration settings are automatically added to the Essbase configuration file, with the values corresponding to your selected range.

Notes

SERVERPORTBEGIN and SERVERPORTEND cannot have the same value.

Example

```
AGENTPORT 1478  
SERVERPORTBEGIN 32470  
SERVERPORTEND 32600  
PORTINC 5
```

This example produces these results:

- AGENTPORT sets the port that the Agent will use at 1478.
- SERVERPORTBEGIN sets the value that the first application process will try to use for a port at 32470.
- SERVERPORTEND sets the highest port number value the server can use.
- PORTINC controls the increment value used for each port. In this example, if the first application process was able to use port number 32470, then the next process would use 32475.

See Also

[AGENTPORT](#)

[SERVERPORTEND](#)

SERVERPORTEND

The SERVERPORTEND configuration setting specifies the highest value that Essbase tries to use for a port when it starts an application process (ESSSVR). If the value is unavailable, the application process fails.

Syntax

```
SERVERPORTEND n
```

n—The highest value for a port number that Essbase tries to use for a application process. If the port is unavailable, the application process fails. This port number should not be in use by any other process. The default value is 33768.

Description

SERVERPORTEND specifies the highest port number that Essbase uses when trying to start an application process.

When you configure Essbase using the configuration tool, you select the range of ports to reserve for Essbase application servers. The SERVERPORTBEGIN and SERVERPORTEND configuration settings are automatically added to the Essbase configuration file, with the values corresponding to your selected range.

Notes

SERVERPORTBEGIN and SERVERPORTEND cannot have the same value.

Example

```
AGENTPORT 1478
SERVERPORTBEGIN 32470
SERVERPORTEND 32600
PORTINC 5
```

This example produces these results:

- AGENTPORT sets the port that the additional Agent will use at 1478.
- SERVERPORTBEGIN sets the value that the first application process will try to use for a port at 32470.
- SERVERPORTEND sets the highest port number value this installation can use.
- PORTINC controls the increment value used for each port. In this example, if the first server process was able to use port number 32470, then the next process would use 32475.

See Also

[AGENTPORT](#)

[SERVERPORTBEGIN](#)

SERVERTHREADS

The `SERVERTHREADS` configuration setting overrides the default value of the number of threads that the Essbase application process (ESSSVR) can spawn. Application threads are used in calculations, client requires, administrative activities, etc.

When a transaction is requested, the application process (ESSSVR) assigns a thread to the transaction and releases the thread when the transaction is completed.

Syntax

```
SERVERTHREADS [appname] n
```

- *appname*—Optional. Specifies an application; the `SERVERTHREADS` setting applies to all databases within the named application.

If you do not specify an application, the setting applies to all applications and databases on the Essbase instance.

- *n*—The number of threads that the application process (ESSSVR) can spawn. You can specify a number between 20 to 1024, inclusive. The default value is 60.

If you specify a value that is less than the minimum, Essbase interprets the value as 20. If you specify a value greater than 1024, it is interpreted as 1024.

Notes

- While the actual maximum value you can set is 1024, the maximum number of threads an operating system can handle might be much lower. Before specifying a value greater than the default value, check with your system administrator, as higher values can significantly consume system resources.
- If the computer on which Essbase Server runs freezes while running multiple reports simultaneously, increase the value of `SERVERTHREADS` by one for each report you run.
- Each application thread may create child threads for tasks such as parallel calculation, parallel data load or export, and parallel restructuring. If the total number of running threads is too high, threads may lose efficiency in contending for server resources.
- When running a parallel calculation that includes the `@XREF` calculation function, the application associated with the database returns a timeout error if the number of threads specified for the `CALCPARALLEL` configuration setting is higher than the number of threads specified by the `SERVERTHREADS` configuration setting. For example, the default value of `SERVERTHREADS` is 20. If you set `CALCPARALLEL` to 25, an application timeout error is generated.

Example

```
SERVERTHREADS 25
```

Allows all applications on Essbase Server to spawn up to 25 threads.

```
SERVERTHREADS Sample 100
```

Allows the Sample application on Essbase Server to spawn up to 100 threads.

SPLITARCHIVEFILE

When backing up an Essbase cube to an archive file, the SPLITARCHIVEFILE configuration setting enables you to split the archive file into multiple files. This can be useful if you cannot use large files, or the file-transfer tools that you use cannot handle large files.

Syntax

```
SPLITARCHIVEFILE TRUE | FALSE
```

- **TRUE**— Creates multiple archive files.
- **FALSE**— Creates a single archive file. This is the default.

You must restart Essbase Server to initialize any change to the configuration file.

Description

The first (or main) archive file that Essbase creates uses the file name that you specify (for example, `samplebasic.arc`). When the main archive file reaches the 2 GB limit, Essbase creates another archive file. In naming the other archive files, Essbase increments the main archive file name with "_x", where x is an integer (starting with 1). Using the `samplebasic.arc` example, if three archive files are created when backing up the Sample Basic cube, the file names would be:

```
samplebasic.arc  
samplebasic_1.arc  
samplebasic_2.arc
```

All archive files are created in the directory that you specified when specifying the file name and location of the main archive file.

If you use the default, single-file configuration, Oracle recommends saving archive files to a file system that supports large files. For Windows, the file system must be formatted as NTFS. For UNIX, large file support must be enabled (for example, use the **ulimit** command to specify a specific file size based on the size of the cube or set **ulimit** to unlimited). See your operating system documentation.

Note:

When restoring a cube in which the archive file is split into multiple files, Essbase looks for multiple archive files, even if, after the backup, you subsequently set SPLITARCHIVEFILE to FALSE for that cube. Also, Essbase expects all of a cube's archive files (main and split) to be in the same directory.

See Also

Alter Database MaxL (see **archive**) operations)

SQLFETCHERRORPOPUP

You can use the SQLFETCHERRORPOPUP configuration setting if you want Essbase to generate SQL import errors when fetching data from a SQL database during a data load or a dimension build.

Syntax

```
SQLFETCHERRORPOPUP TRUE | FALSE
```

- TRUE—SQL imports generate error messages.
- FALSE—SQL imports do not generate error messages. This is the default.

Example

```
SQLFETCHERRORPOPUP TRUE  
SQLFETCHERRORPOPUP FALSE
```

SSANCESTORONTOP

The SSANCESTORONTOP configuration setting enables Smart View users to specify that ancestors be positioned at the top in Essbase grid client operations.

Syntax

```
SSANCESTORONTOP [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the application to which the configuration applies.
- *dbname*—Optional. Specifies the cube to which the configuration applies.
- TRUE—Smart View users can specify the ancestor position for hierarchies in ad hoc grids.
- FALSE—Smart View users cannot specify the ancestor position. This is the default.

Description

If this configuration property is set to TRUE, Smart View users can specify ancestor position for hierarchies in ad hoc grids. By default, this parameter is not enabled, and the ancestor is positioned at the bottom.

Example

```
SSANCESTORONTOP Sample TRUE
```

See Also

[Specifying Ancestor Position in Ad Hoc Grids](#) in *Working with Oracle Smart View for Office* (available on the Books tab)

SSAUDIT

Use the SSAUDIT configuration setting to have Essbase log successfully completed grid update transactions, appending to existing logs after archiving. The resulting logs can be used as a source of input data upon recovery after archive operations or other server interruptions.

SSAUDIT configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
SSAUDIT appname [ dbname [ log_path ] ]
```

- *appname*—Application name.
- *dbname*—Optional. Database (cube) name.
- *log_path*—Optional. Full directory path where you want the information stored. Do not specify a *log_path* value unless you have also provided a value for *dbname*.

Default behavior:

- If SSAUDIT (or SSAUDITR) is specified, grid update logging is not enabled.
- If SSAUDIT (or SSAUDITR) is specified with no arguments, Essbase activates grid update logging for all cubes in all applications on the Essbase Server, and puts the log in the default directory: *<Application Directory>/app/appname/dbname*.

See Environment Locations in the Essbase Platform for information about directory locations in Essbase.

Use the value `xxxxx` to indicate "all" for any argument.

You can issue up to ten (total) SSAUDIT statements per application.

Notes

- SSAUDIT creates two logs for each cube: *dbname.atx*, which stores the update transaction records that can be used as the input source for data load, and *dbname.alg*, which stores history records from every update transaction, including user name, time stamp, and number of updated rows.
- SSAUDIT is not available when using Free-Form reporting in Smart View.
- If you have duplicate cube names in different applications, do not store their error logs in the same directory. If you do, the log for one cube will be replaced by the log for any subsequent cube with the same name.
- Essbase ensures that if you enable grid update logging, updates do not take place without getting logged. If Essbase cannot write to the update logs for any reason, Essbase fails the update transaction and issues an error message.
- SSAUDIT may slow grid client data-update operations.

Example

```
SSAudit xxxxx xxxxx c:\sslog
```


Enables logging for all applications and cubes, storing the log in the path `c:\sslog`. This example assumes that you do not have duplicate cube names (see Notes).

The following is an example of the contents of an .ATX log file for Sample Basic:

```
"New York" "Massachusetts" "Florida" "Connecticut" "New Hampshire" "East"
"Actual" "100-20" "Sales" "Jan" #Mi #Mi 200. 100. 200. 200.
"Actual" "100-20" "Sales" "Feb" #Mi #Mi 206. 100. 200. 206.
"Actual" "100-20" "Sales" "Mar" #Mi #Mi 214. 100. 200. 214.
"Actual" "100-20" "Sales" "Apr" #Mi #Mi 267. 100. 200. 267.
"Actual" "100-20" "Sales" "May" #Mi #Mi 273. 100. 200. 273.

"New York"
"Actual" "100-20" "Sales" "Jan" 8888.
"Actual" "100-20" "COGS" "Jan" 8888.
"Actual" "100-20" "Marketing" "Jan" 8888.
"Actual" "100-20" "Payroll" "Jan" 8888.
"Actual" "100-20" "Misc" "Jan" 8888.
"Actual" "100-20" "Opening Inventory" "Jan" 8888.
"Actual" "100-20" "Additions" "Jan" 8888.
"Actual" "100-20" "Ending Inventory" "Jan" 8888.
"Actual" "100-20" "Inventory" "Jan" 8888.
"Actual" "100-30" "Sales" "Jan" 8888.

"New York"
"Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"
"Actual" "100-20" "Sales" 8888. 9999. #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi
"Actual" "100-20" "COGS" 8888. 9999. #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi
"Actual" "100-20" "Marketing" 8888. 9999. #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi
#Mi
"Actual" "100-20" "Payroll" 8888. 9999. #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi
#Mi
"Actual" "100-20" "Misc" 8888. 9999. #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi

"New York"
"Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"
"Actual" "200-20" "COGS" 7777. 6666. #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi #Mi
```

The following is an example of the contents of an .alg log file for Sample Basic. The ALG information describes the updated data records that are logged in the .atx file.

```
[Thu May 24 17:29:07 2012]
Create Spreadsheet Update Log
[Thu May 24 17:37:46 2012]
Log Updates From User [ admin ] Starting At Row [ 1 ] For A Total Of [ 7 ]
Rows
[Thu May 24 17:42:29 2012]
Log Updates From User [ admin ] Starting At Row [ 8 ] For A Total Of [ 12 ]
Rows
[Thu May 24 17:45:31 2012]
Log Updates From User [ admin ] Starting At Row [ 20 ] For A Total Of [ 8 ]
Rows
[Thu May 24 17:47:14 2012]
```

Log Updates From User [admin] Starting At Row [28] For A Total Of [4]
Rows

See Also

Alter Database begin | end archive (MaxL)

[SSAUDITR](#)

SSAUDITR

Use the SSAUDITR configuration setting to have Essbase log successfully completed grid update transactions, clearing the logs at the end of the archiving process. The logs can be used as a source of input data upon recovery after archive operations or other server interruptions.

SSAUDITR configuration applies only to block storage (BSO) databases, and does not apply to aggregate storage (ASO) databases.

Syntax

```
SSAUDITR appname [dbname [log_path]]
```

- *appname*—Application name.
- *dbname*—Optional. Database (cube) name.
- *log_path*— Optional. Full directory path where you want the information stored. Do not specify a *log_path* value unless you have also provided a value for *dbname*.

Default behavior:

- If SSAUDITR (or SSAUDIT) is not specified, grid update logging is not enabled.
- If SSAUDITR (or SSAUDIT) is specified with no arguments, Essbase activates grid update logging for all cubes in all applications on the Essbase Server, and puts the log in the default directory: *<Application Directory>/app/appname/dbname*.

See Environment Locations in the Essbase Platform for information about directory locations in Essbase.

Use the value `xxxxx` to indicate "all" for any argument.

You can issue up to ten (total) SSAUDITR and/or [SSAUDIT](#) statements per application.

Notes

- SSAUDITR creates two logs for each cube: *dbname.atx*, which stores the update transaction records that can be used as the input source for data load, and *dbname.alg*, which stores history records from every update transaction, including user name, time stamp, and number of updated rows.
- Essbase ensures that if you enable grid update logging, updates do not take place without getting logged. If Essbase cannot write to the update logs for any reason, the update transaction fails and an error message is issued.
- SSAUDITR may slow grid client data-update operations.
- The update log file will not be cleared if the cube is shut down during archive mode. The cube is expected to remain running while in archive mode.

Example

```
SSAuditR demo
```

Enables logging with refresh (clear) for all cubes belonging to the Demo application. The log is stored in the default directory.

See Also

[SSAUDIT](#), which does not clear the logs after archive.

Alter Database begin | end archive (MaxL)

SSBULKGRIDPROCESSING

Enable the SSBULKGRIDPROCESSING configuration setting if you want Essbase to optimize asymmetric Grid API queries for XOLAP. The optimization is applicable only if SSOPTIMIZEDGRIDPROCESSING is false.

Syntax

```
SSBULKGRIDPROCESSING [appname [dbname]] ON | OFF
```

- *appname*—Optional. Specifies the application for which bulk grid processing is to be set.
If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.
To enable the setting for a specific cube, you must specify an application and cube.
If you do not specify an application, you cannot specify a cube, and the setting applies to all applications and cubes on Essbase Server.
- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which bulk grid processing is to be set.
If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- ON—Essbase optimizes asymmetric Grid API queries for XOLAP.
- OFF—Essbase does not optimize asymmetric Grid API queries for XOLAP.
The default value is OFF.

For changes to the configuration to take effect, you must restart Essbase Server.

Example

```
SSBULKGRIDPROCESSING ON
```

Enables bulk grid processing for grid client operations on all applications and cubes on Essbase Server.

See Also

[SSOPTIMIZEDGRIDPROCESSING](#)

SSINVALIDTEXTDETECTION

Enable the SSINVALIDTEXTDETECTION configuration setting if you want Essbase to generate an error when a grid client user enters invalid text data into a cell, which could possibly cause the user to misinterpret the data in the grid.

Syntax

```
SSINVALIDTEXTDETECTION TRUE | FALSE
```

- **TRUE**—An error message is displayed citing the invalid text and location, and saying to remove the text and retry.
- **FALSE**—Default value. No error message is displayed. The text that was entered is ignored.

Examples

```
SSINVALIDTEXTDETECTION TRUE  
SSINVALIDTEXTDETECTION FALSE
```

SSLCIPHERSUITES

The SSLCIPHERSUITES configuration setting defines one or more cipher suites to use for negotiating the security settings for a network connection when Essbase is in secure mode (TLS/SSL protocol).

Syntax

```
SSLCIPHERSUITES ciphersuite_1[,ciphersuite_2,...,ciphersuite_6]
```

At least one cipher suite is required. A comma-delimited list of cipher suites, in order by preference, is supported. The first cipher suite in the list has the highest priority.

Description

You can change the default cipher suite by changing the order. The following is the default order.

```
TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA,  
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,  
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,  
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,  
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,  
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA,  
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,  
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,  
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,  
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,  
SSL_RSA_WITH_3DES_EDE_CBC_SHA,  
TLS_RSA_WITH_AES_128_CBC_SHA,  
TLS_RSA_WITH_AES_128_CBC_SHA256,  
TLS_RSA_WITH_AES_128_GCM_SHA256,
```

```
TLS_RSA_WITH_AES_256_CBC_SHA,  
TLS_RSA_WITH_AES_256_CBC_SHA256,  
TLS_RSA_WITH_AES_256_GCM_SHA384
```

**Note:**

For the highest level of security, reverse the order in which these cipher suites are listed.

Example

```
SSLCIPHERSUITES SSL_RSA_WITH_3DES_EDE_CBC_MD5,SSL_RSA_WITH_DES_CBC_SHA
```

See Also[AGENTSECUREPORT](#)[CLIENTPREFERREDMODE](#)[ENABLECLEARMODE](#)[ENABLESECUREMODE](#)[NETSSLHANDSHAKETIMEOUT](#)[WALLETPATH](#)

SSLOGUNKNOWN

The SSLOGUNKNOWN configuration setting enables you to control whether Essbase logs error messages for unknown member names during grid operations. You can get a list of every unknown member name, or suppress messages of this type.

Syntax

```
SSLOGUNKNOWN TRUE | FALSE
```

- **TRUE**—Essbase displays and logs an error message for each unknown member name that it encounters during a grid operation. The default is TRUE.
- **FALSE**—Essbase does not display error messages when it encounters an unknown member name nor does it log an error for each unknown member it encounters during a grid operation.

Notes

SSLOGUNKNOWN creates an entry in the application log, *application_name.log*, in the application directory.

See Environment Locations in the Essbase Platform for information about directory locations in Essbase.

Example

```
SSLOGUNKNOWN TRUE
```

Essbase generates and logs an error message each time it encounters any number of unknown member names during a grid operation.

See Also

[TIMINGMESSAGES](#)

SSMEMBERIDPROCESSING

The SSMEMBERIDPROCESSING configuration setting enables you to have Smart View's member display options include the ability to display Essbase Member IDs (using the **Member Identifier** option).

Options

<ul style="list-style-type: none"> Member Options Data Options Advanced Formatting Cell Styles Extensions 	<p>Change member and dimension options on the grid.</p> <p>General ⓘ</p> <p>Zoom In Level Next Level ▾</p> <p>Member Name Display Member Identifier ▾</p> <p>Indentation Subitems ▾</p> <p>Ancestor Position Bottom ▾</p>
--	--

An internal member ID is associated with each member for all outlines.

If you want to display member IDs in Smart View, enable this setting at the [application level](#) to make that option available in Smart View member options.

Shared members have unique member IDs, distinct from their prototype members.

Syntax

```
SSMEMBERIDPROCESSING [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the application for which member ID display is enabled.
 - If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application.
 - To enable the setting for a specific database, you must specify an application and database.
 - If you do not specify an application, you cannot specify a database, and the setting applies to all applications and databases on Essbase Server.
- *dbname*—Optional. Specifies the database, in the application specified by *appname*, for which member ID display is enabled.
 - If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.

- TRUE—Smart View options enable display of members using member IDs.
- FALSE—Smart View options do not enable display of member IDs. This is the default.

Description

For Smart View queries on duplicate member name outlines, displaying Member IDs can help Smart View maintain report validity for all members, even when members in the outline are moved or renamed.

Notes

If you opt to track members using qualified member names instead of member IDs, Smart View reports may become invalid if members in the outline are moved or renamed.

Example

```
SSMEMBERIDPROCESSING Sample TRUE
```

SSOPTIMIZEDGRIDPROCESSING

The SSOPTIMIZEDGRIDPROCESSING configuration setting enables you to change whether optimized grid processing, which cuts the input grid into symmetric grids to create fewer symmetric queries, is enabled for Essbase grid client operations. The optimization is enabled by default.

Syntax

```
SSOPTIMIZEDGRIDPROCESSING [appname [dbname]] TRUE | FALSE
```

- *appname*—Optional. Specifies the application for which optimized grid processing is to be set.
If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all databases in the specified application.
To change the setting for a specific database, you must specify an application and database.
If you do not specify an application, you cannot specify a database, and the setting applies to all applications and databases on Essbase Server.
- *dbname*—Optional. Specifies the database, in the application specified by *appname*, for which optimized grid processing is to be set.
If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.
- TRUE—Enables optimized grid processing for grid client operations.
The default value is TRUE.
- FALSE—Disables optimized grid processing for grid client operations.

For changes to the configuration file to take effect, you must restart Essbase Server.

Example

```
SSOPTIMIZEDGRIDPROCESSING FALSE
```

Turns off optimized processing for grid client operations on all applications and databases on Essbase Server.

SSPROCROWLIMIT

The SSPROCROWLIMIT configuration setting controls the maximum number of rows Essbase processes on a Smart View or other grid client request.

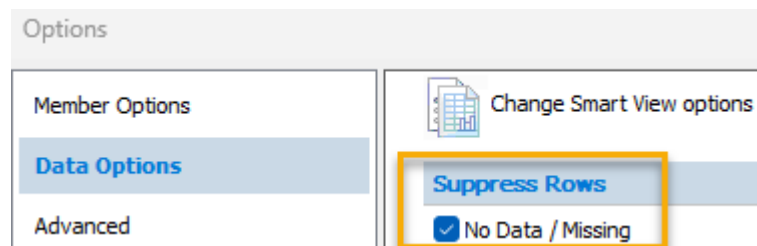
Syntax

```
SSPROCROWLIMIT n
```

n—An integer value of 16,384 or higher. The default value is 250,000.

Description

SSPROCROWLIMIT is in effect only when the option to suppress #Missing rows is selected.



Essbase counts the rows before suppression; that is, #Missing rows and rows containing zero values are included.

When Smart View users zoom in on one or more members, Essbase must process a larger grid containing selected members expanded to the zoom-in level set in the options. When the option to suppress #Missing Rows is set, Essbase returns only rows with at least one column containing a non-missing value. SSPROCROWLIMIT defines the maximum size (number of rows) of the larger grid that Essbase needs to process. This setting prevents excessive memory usage for a single grid operation.

When the option to suppress #Missing rows is not selected, the limit is 64000.

Notes

- SSPROCROWLIMIT applies to unprocessed rows; that is, it is the number of rows Essbase accepts before processing. Row processing eliminates missing rows. After processing, the number of rows that the client can retrieve depends on grid-client-defined limits.
- If SSPROCROWLIMIT is exceeded, Essbase issues an error message and stops processing the request.
- This setting is not used in the Smart View Free form mode.
- Oracle does not recommend using a limit higher than 500,000.

Example

```
SSPROCROWLIMIT 300000
```


SUPNA

This Essbase configuration setting enables you to have Smart View or another grid client interface suppress the display of cells for which a user has no access (#NoAccess), in addition to suppressing rows with no data (#Missing).

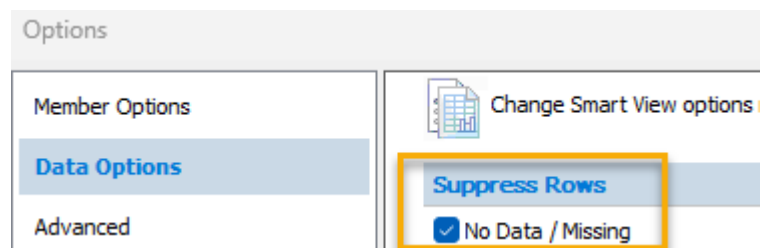
Syntax

SUPNA ON | OFF

- ON—The option to suppress #Missing rows also suppresses the display of cells for which a user has no access.
- OFF—The option to suppress #Missing rows does not suppress the display of cells for which a user has no access. This is the default.

Description

SUPNA is in effect only when the Smart View option to suppress #Missing rows is selected.



You can use SUPNA configuration to have Essbase also suppress the display of #NoAccess cells.

Example

SUPNA OFF

For all databases on the server, Essbase does not suppress cells for which a user has no access. These cells appear in the grid as #NoAccess. Rows of missing data are suppressed.

SVRIDLETIME

The SVRIDLETIME configuration setting specifies the number of minutes an Essbase application can be idle before it is shut down.

If an application is idle for the time period set, it will stop running. When any new request is made (for example, Smart View query, Outline, or CLI activity), the application starts automatically.

Syntax

SVRIDLETIME [*appname*] *n*

where n is the number of minutes of idle time permitted before shutdown. The default value is 120 minutes. The minimum value is 1 minute. The maximum value is 20160 minutes (two weeks). To disable automatic shutdown, set n to 0.

Notes

- By default, applications are set to shut down after 2 hours of idle time.
- The SVRIDLETIME value should always be set to a few more minutes than the idle user session logout interval, which you can set using the MaxL alter system **set session_idle_limit** grammar.

Example

```
SVRIDLETIME Sample 15
```

TARGETASOOPT

You can enable the TARGETASOOPT configuration setting for Essbase to speed up large queries when an aggregate storage (ASO) cube is the source of a transparent partition.

Use TARGETASOOPT to potentially optimize large queries (from Smart View or other grid clients, MDX, or Report Writer) to an aggregate storage database across a transparent partition when the source outline and target outline are identical in the partition region definition area.

Syntax

```
TARGETASOOPT [appname] TRUE | FALSE
```

- *appname*—Optional. Application name. If you specify a value for *appname*, the setting applies to all databases in the specified application. If you do not specify an application, the setting applies to all applications and databases on the Essbase Server.
- FALSE—The default. Optimization is not enabled, even if queries match the required criteria (see [Description](#)).
- TRUE—Optimization is enabled for queries that match the required criteria (see [Description](#)).

When TARGETASOOPT is TRUE, Essbase determines if the partitioned region's outlines are identical on the source and target, and if so, optimizes the query format.

You must restart Essbase Server to initialize any change to the configuration file.

Description

TARGETASOOPT enables an alternate (compact) format for sending a query (from Smart View or other grid clients, MDX, or Report Writer) to an ASO source database. It may speed up large queries between databases that match the following criteria:

- Two Essbase databases are transparently partitioned (for example, to enable write-back for an ASO database)
- The source of the transparent partition is an ASO database
- The partitioned area definitions in the source and target are identical.
- The source and target outlines are identical.

Notes

If at query time the source and target outlines have been modified after the last validation, even if the partition region definition outlines are still identical, TARGETASOOPT is disabled for the query. To enable TARGETASOOPT for the query, you must revalidate the partitions.

Example

```
TARGETASOOPT TRUE
```

See Also

[TARGETTIMESERIESOPT](#)

TARGETTIMESERIESOPT

The TARGETTIMESERIESOPT configuration setting enables you to optimize queries across transparent partitions for Essbase outlines that have a time dimension with Dynamic Time Series members. Use TARGETTIMESERIESOPT only if the time dimensions on the source and target partitions are identical.

If TARGETTIMESERIESOPT is enabled, queries with Dynamic Time Series members will be faster. However, the time dimensions on the source and target partitions are not the same, this setting may produce incorrect results. Restart Essbase to enable this setting to take effect for the Dynamic Time Series members that have been enabled at run time.

Syntax

```
TARGETTIMESERIESOPT TRUE | FALSE
```

- TRUE—Enables query optimization across transparent partitions for outlines that have a time dimension with Dynamic Time Series members.
- FALSE—Query optimization is not enabled. This is the default.

Example

```
TARGETTIMESERIESOPT TRUE
```

See Also

[TARGETASOOPT](#)

TIMINGMESSAGES

The TIMINGMESSAGES configuration setting enables you to control whether Essbase logs the duration of each grid and report query in the application log.

Syntax

```
TIMINGMESSAGES TRUE | FALSE
```

- TRUE—Essbase logs these items:

- The duration of all grid and report queries in the application log.
- Timestamps of the queries' execution.
- Messages about dynamic calculator cache usage for each data retrieval.

The default setting is TRUE.

- FALSE—Essbase does not log the query timing information.

Description

TIMINGMESSAGES controls whether Essbase logs the duration of each grid and report query in the application log. Setting TIMINGMESSAGES to FALSE disables the logging of query durations in the application log. If the timing of queries is disabled, Essbase does not have to communicate with the operating system to get query start and finish times. As a result, query execution times may be improved in environments with many concurrent users. Disabling this parameter also decreases the size of the application log.

Example

```
TIMINGMESSAGES TRUE
```

```
Causes Essbase to time and log the duration of queries in the
application log. For example: [Thu Nov 26 14:55:32 2020]Local/
Sample/Basic/admin/Info(1020055) Spreadsheet Extractor
Elapsed Time : [0.078] seconds.
```

```
TIMINGMESSAGES FALSE
```

```
Disables the logging of query durations.
```

See Also

[SSLOGUNKNOWN](#)

TRACE_REPORT

Use TRACE_REPORT configuration setting to debug performance for multiple, concurrent queries. Query tracing helps you monitor Essbase query performance metrics. The level of information you want to keep is reported in the application log and/or a separate tracing log, `trace_report.log`.

Description

TRACE_REPORT enables query tracing for grid retrievals, MDX queries, and transparent partition sources. The query tracing output file includes information on:

- The query type
- The time of the query
- User name and IP address
- Report text

Syntax

```
TRACE_REPORT n
```

where *n* should be set to -1, to enable query tracing. The default value is 0 (query tracing is off).



Note:

Oracle Support may set a different non-zero integer value to reduce the items logged to `trace_report.log`.

Example

```
TRACE_REPORT -1
```

Configuring Ongoing Query Tracing

Use the configuration settings `TRACE_REPORT` and `LONGQUERYTIMETHRESHOLD` together to manage the query tracing information that should be collected for the cube in the application's Oracle Diagnostic Log (ODL), and/or printed to `trace_report.log`.

You can enable `TRACE_REPORT` for tracking general query performance, but Oracle recommends filtering out all but the longest running queries by also setting `LONGQUERYTIMETHRESHOLD`.

Ideally,

- Configure Essbase to only log information for the top 5 percent of longest-running queries. If you don't know this value, 60 seconds may be a good place to start.
- Regularly check that log files do not grow too large.

Configuration Type	TRACE_REPORT value	LONGQUERYTIMETHRESHOLD value	Effect on Logs
ODL only, with threshold	Set to 0 (or not set at all)	Set to a value <i>n</i> (seconds) that is greater than 0	Performance metrics on all queries running longer than <i>n</i> seconds are printed to the application's ODL log.
ODL and trace, with threshold	Set to -1	Set to a value <i>n</i> (seconds) that is greater than 0	Performance metrics on all queries running longer than <i>n</i> seconds are printed to the application ODL log. Information on queries running longer than <i>n</i> seconds is printed to <code>trace_report.log</code> .

Configuration Type	TRACE_REPORT value	LONGQUERYTIMETHR ESHOLD value	Effect on Logs
Trace without threshold - <i>not recommended!</i>	Set to -1	Set to 0 (or not set at all)	Information on all queries, regardless of running time, is printed to <code>trace_report.log</code> . This configuration is not recommended, because <code>trace_report.log</code> can become too large if not filtered using a threshold, and query performance is affected if the threshold is too low.

The `trace_report` log is located in `<Domain Home>/servers/<Essbase-Managed-Server-Name>/logs/essbase/essbase/app/<application-name>/<cube-name>/trace_report.log`.

The application ODL log is located in `<Domain Home>/servers/<Essbase-Managed-Server-Name>/logs/essbase/essbase/app/<application-name>/<application-name>_ODL.log`.

For details about log file locations, refer to Environment Locations in the Essbase Platform.

When to Use Different Query Tracing Options

The `QUERYTRACE` configuration setting is designed for debugging a single query, in case any problems are observed. It prints the trace log to the cube directory as `query_trace.txt`, and the file is cleared by default before each query execution. `QUERYTRACE` should not be left enabled in the application configuration for long term use, as it may affect performance.

The `TRACE_REPORT` configuration setting provides fewer details than `QUERYTRACE`, but is able to trace multiple, concurrent queries. `TRACE_REPORT` is a good option for gathering information on many concurrent queries running over a period of time.

Enabling query logging by setting the `QUERYLOG` parameter in `dbname.cfg` (in the cube directory) is designed for tracking user query patterns for a cube, in XML format.

Notes

`TRACE_REPORT` applies to block storage (BSO) databases, including those in hybrid mode. `TRACE_REPORT` does not apply to aggregate storage (ASO) databases.

See Also

[LONGQUERYTIMETHRESHOLD](#)

[QUERYTRACETHRESHOLD](#)

[QUERYTRACE](#)

TRANSACTIONLOGDATALOADARCHIVE

The TRANSACTIONLOGDATALOADARCHIVE configuration setting specifies the type of data to archive when logging transactions. By default, Essbase archives only data load and rules files for client data loads.

During transaction logging, Essbase creates archive copies of data load and rules files in the following directory:

```
<Application Directory>/app/appname/dbname/Replay
```

See Environment Locations in the Essbase Platform for information about directory locations in Essbase.

These files are then used during the replay of a logged transaction.

To enable transaction logging and replay, use the [TRANSACTIONLOGLOCATION](#) configuration setting.

Transaction logging and replay facilitates recovery of an Essbase block storage cube. Transaction logging and replay does not apply to aggregate storage cubes.

Syntax

```
TRANSACTIONLOGDATALOADARCHIVE [appname [dbname]] [OPTION]
```

- *appname*—Optional. Specifies the application for which to archive the data and rules associated with logged transactions.

If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.

To enable the setting for a specific cube, you must specify an application and cube.

If you do not specify an application, you cannot specify a cube. If you do not specify an application and cube, the setting is global and applies to all cubes on Essbase Server.

- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which to archive the data and rules associated with logged transactions.

If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.

- *OPTION*—Valid values are as follows:
 - CLIENT: (Default) Archives data load and rules files for client data loads. This is the default.
 - SERVER: Archives data load and rules files on the server and SQL-server data loads.

⚠ Caution:

Server data loads are replayed using the data load and rules files that are archived on the server in the `Replay` directory. Do not rename these files. Also, if the contents of the data load and rules files are changed before the replay operation, the modified data is used during replay. Therefore, the data in the recovered cube will not be the same as the original data.

- `SERVER_CLIENT`: Archives server and client data.
- `NONE`: No data is archived.

If you select `NONE` and use client data, Essbase cannot replay the data load. In this case, to recover transactions, you must manually load the client data before you replay the remaining transactions.

If you use server or SQL data, and the data and rules files are not archived in the `Replay` directory (for example, you did not use the `SERVER` or `SERVER_CLIENT` option), Essbase replays the data that is currently in the data source, which may or may not be the data that was originally loaded.

You must restart Essbase Server to initialize any change to the configuration.

Example

```
TRANSACTIONLOGDATALOADARCHIVE SERVER_CLIENT
```

Archives server and client data for all cubes on Essbase Server.

See Also

[TRANSACTIONLOGLOCATION](#) configuration setting

Query Database MaxL statement (with **list transactions**)

Alter Database MaxL statement (with **replay transactions**)

TRANSACTIONLOGLOCATION

The `TRANSACTIONLOGLOCATION` configuration setting specifies whether to enable write transaction logging, and specifies an existing directory on Essbase Server for the transaction log store.

Transaction logging and replay facilitates recovery of an Essbase block storage cube. Transaction logging and replay does not apply to aggregate storage cubes.

Syntax

```
TRANSACTIONLOGLOCATION [appname [dbname]] LOGLOCATION NATIVE ENABLE | DISABLE
```

- *appname*—Optional. Specifies the application for which transaction logging and replay is to be enabled.

If you specify a value for *appname* and do not specify a value for *dbname*, the setting applies to all cubes in the specified application.

To enable the setting for a specific cube, you must specify an application and cube.

If you do not specify an application, you cannot specify a cube. If you do not specify an application and cube, the setting is global and applies to all databases on Essbase Server.

- *dbname*—Optional. Specifies the database (cube), in the application specified by *appname*, for which transaction logging and replay is to be enabled.

If you specify a value for *dbname* but do not specify a value for *appname*, your specification is ignored.

- *LOGLOCATION*—Specifies the directory in which the transaction log store is written.

Oracle recommends specifying multiple log locations.

See Environment Locations in the Essbase Platform for information about directory locations in Essbase.

On Windows, you can specify the location using Uniform Naming Convention (UNC) syntax, which is `\\ComputerName\SharedFolder\Resource`. Use UNC syntax only if `<Application Directory>` is also specified in UNC. Mixed path types are not supported.

- *NATIVE*—A reserved field. Do not change this value.
- *ENABLE* | *DISABLE*—Specifies whether to enable or disable transaction logging.

You must restart Essbase Server to initialize any change to the configuration.

Description

You can use multiple `TRANSACTIONLOGLOCATION` statements to enable transaction logging at a more global level and, at the same time, disable logging at a more granular level. In the configuration, the more global enabling statement must precede the more granular disabling statement for the override to take effect.



Note:

If transaction logging is enabled for an application or database that you later rename or copy, you must enable logging for the renamed or copied application or database and you must use the same path that is specified in the `TRANSACTIONLOGLOCATION` setting.

Example

```
TRANSACTIONLOGLOCATION Sample C:\logs\trlogs NATIVE ENABLE
```

Enables transaction logging for all databases associated with the Sample application and writes the log store to the `trlogs` directory.

```
TRANSACTIONLOGLOCATION logs/trlogs NATIVE ENABLE
```

```
TRANSACTIONLOGLOCATION Sample logs/trlogs NATIVE DISABLE
```

The first statement enables transaction logging for all applications and their associated cubes on Essbase Server; the second statement

disables transaction logging for all cubes associated with a specific application (Sample).

```
TRANSACTIONLOGLOCATION Sample logs/trlogs/Sample NATIVE ENABLE
TRANSACTIONLOGLOCATION Sample Basic logs/trlogs/Sample NATIVE DISABLE
```

The first statement enables transaction logging at the application level (Sample); the second statement disables transaction logging for a specific cube (Basic) in the application.

```
TRANSACTIONLOGLOCATION Sample Basic \\machine-name\shared\loglocation NATIVE
ENABLE
```

Enables transaction logging for Sample Basic and writes the log store to the specified shared location.

See Also

[TRANSACTIONLOGDATALOADARCHIVE](#) configuration setting

Query Database MaxL statement (with **list transactions**)

Alter Database MaxL statement (with **replay transactions**)

TRIGMAXMEMSIZE

The TRIGMAXMEMSIZE configuration setting specifies the maximum amount of memory that Essbase can allocate to the triggers feature.

Syntax

```
TRIGMAXMEMSIZE [appname [dbname]] memsize
```

- *appname*—Optional. Sets the available memory cache for all cubes in the specified application.
- *dbname*—Optional. Sets the available memory cache for the specified database (cube). If you specify a cube, you must specify the application that contains it.
- *memsize*—Available memory cache size (in bytes). Default: 4096 bytes. Minimum: 4096 bytes. Maximum: 8388608 bytes (8MB). Setting *memsize* to zero (0), or a negative value, disables all triggers.

Description

TRIGMAXMEMSIZE specifies the maximum amount of memory available to the Essbase triggers feature. The triggers feature lets you efficiently monitor data changes in a cube. If data breaks the rules that you have specified, Essbase logs the information in a file or sends an e-mail alert.

Notes

You must specify the memory in bytes. If you specify a size greater than the maximum of 8388608 bytes, Essbase automatically sets the size to 8388608 bytes.

Example

```
TRIGMAXMEMSIZE 12288
```

Sets the maximum memory cache for the triggers feature to 12288 bytes (12K). The setting applies to all applications and cubes on the Essbase Server.

See Also

create trigger (MaxL statement)
display trigger (MaxL statement)
alter trigger (MaxL statement)
drop trigger (MaxL statement)

UPDATECALC

The UPDATECALC configuration setting enables you to set whether Essbase's Intelligent Calculation feature is on or off. By default, Intelligent Calculation is on for block storage (BSO) cubes, meaning that Essbase calculates only updated blocks and their dependent parents.

This setting does not apply to aggregate storage cubes.

Syntax

```
UPDATECALC TRUE | FALSE
```

- TRUE—Intelligent Calculation is turned on. Essbase calculates only updated blocks and their dependent parents. This is the default.
- FALSE—Intelligent Calculation is turned off. Essbase calculates all data blocks, regardless of whether they have been updated.

Description

UPDATECALC specifies whether Intelligent Calculation is turned on or off by default.

If required during a calculation, you can override this default setting and turn Intelligent Calculation on and off using the SET UPDATECALC command in a calculation script.

Using Intelligent Calculation, Essbase calculates only updated data blocks and their dependent parents. Therefore, the calculation is very efficient.

Example

```
UPDATECALC TRUE
```

See Also

SET CLEARUPDATESTATUS (calculation command)
SET UPDATECALC (calculation command)

WALLETPATH

On Essbase independent deployments, the WALLETPATH configuration setting enables you to specify the path to the Oracle Wallet to enable secure (TLS/SSL) communication for application server and client.

Syntax

```
WALLETPATH path
```

Where *path* is a fully-qualified path that contains less than 1,024 characters.

The default *path* is `<Essbase Config Path>/walletssl`.

To ensure that the configuration is consistent across all nodes of the WebLogic cluster on which Essbase is deployed, set the wallet path to a directory in `<Essbase Config Path>`.

See Environment Locations in the Essbase Platform for information about `<Essbase Config Path>` and other directory locations in Essbase.

On Windows, you can specify the path using Uniform Naming Convention (UNC) syntax, if environment variables are also specified in UNC (mixed path types are not supported). The UNC syntax is `\\ComputerName\SharedFolder\Resource`.

Description

To set up Oracle Wallet, you need the Oracle public key infrastructure (PKI) command line tool, `orapki`. You use the `orapki` utility to manage public key infrastructure elements such as wallets and certificate revocation lists. You must use the version of `orapki` that is in the following directory:

```
<ORACLE_HOME>/oracle_common/bin
```

Notes

For information on implementing secure mode (TLS), see [About Securing Your Communication and Network](#).

Examples

```
WALLETPATH /scratch/oracle/user_projects/domains/essbase_domain/config/  
fmwconfig/essconfig/essbase/walletssl
```

```
WALLETPATH \  
\c\oracle\user_projects\domains\essbase_domain\config\fmwconfig\essconfig\essb  
ase\walletssl
```

WORKERTHREADS

The WORKERTHREADS configuration setting enables you to increase or decrease the number of threads available within the thread pool for parallel Essbase operations.

Syntax

```
WORKERTHREADS [appname] value
```

- *appname* (optional)—Application name. If you do not specify an application, the specified number of threads applies to all applications.
- *value*—The number of threads to make available in the thread pool. The minimum value is 5. The maximum value is 2048. The default value, if the WORKERTHREADS setting is not present, is half of the SERVERTHREADS value.

Description

Historically, Essbase dynamically created threads for parallel operations such as parallel calculation, parallel data load, and parallel restructure. However, beginning in Release 11.1.2.4.000, the following parallel operations do not dynamically create threads, but instead use a set number of threads from a pre-created pool of threads:

- Parallel calculation, with CALCPARALLEL or FIXPARALLEL
- Parallel data load, for aggregate storage and block storage cubes
- Parallel export, for block storage cubes
- Parallel restructuring

Guidelines for Threaded Operations

Oracle recommends the following settings, depending on your system architecture.

Table 2-1 Guidelines for Threaded Operations

Parameter	64-bit (16 cores)	64-bit (32 cores)	64-bit (16 cores) and Two Cubes	Oracle Exalytics In-Memory Machine, 40 core, X2-4	Oracle Exalytics In-Memory Machine, SPARC 128 core	Oracle Exalytics In-Memory Machine, 60 core, X4-4
SERVERTHRE ADS	100 ¹	100 ¹	100 ¹	100 ¹	200 ¹	120 ¹
WORKERTHRE ADS	50	50	50	50	100	60
AGENTTHREA DS	30 ¹	30 ¹	30 ¹	30 ¹	30 ¹	30 ¹
DLTHREADSP REPAIR	2	2	2	2 ²	2 ²	2 ²
DLTHREADSW RITE	2 ³	2 ³	2 ³	2 ³	2 ³	2 ³
EXPORTTHRE ADS ⁴	8	8	8	40	40	40
RESTRUCTUR ETHREADS	4	4	4	8	8	8

Table 2-1 (Cont.) Guidelines for Threaded Operations

Parameter	64-bit (16 cores)	64-bit (32 cores)	64-bit (16 cores) and Two Cubes	Oracle Exalytics In-Memory Machine, 40 core, X2-4	Oracle Exalytics In-Memory Machine, SPARC 128 core	Oracle Exalytics In-Memory Machine, 60 core, X4-4
CALCPARALLEL maximum ⁵	8	8	8	32 ⁶	32	32 ⁶
FIXPARALLEL maximum ⁷	8	8	8	32 ⁶	32	32 ⁶

¹ May need adjustment based on partitioning and concurrency

² 24 on aggregate storage

³ 1 if using .txt load

⁴ In addition to setting EXPORTTHREADS, you must specify multiple data files in the data export

⁵ SET CALCPARALLEL must be set in the calculation script

⁶ 16 if three or more cubes are running concurrent calculations

⁷ FIXPARALLEL must be set in the calculation script

Notes

- WORKERTHREADS is a configuration setting to manipulate the number of threads available in the thread pool. Whether this availability threshold is set explicitly in the configuration, or left to its default value of `SERVERTHREADS/2`, if the effective setting is less than what is demanded by a requested parallel operation, Essbase implicitly lowers the parallelism of the requested operation so that it fits within the parameters of the thread pool.

For example, consider loading data:

The concept of a *pipeline* is relevant to Essbase data loads. A pipeline is a series of data processing elements in memory that may be executed serially or in parallel. An Essbase data load operation uses a pipeline consisting of 5 stages. Therefore, all data load operations need a minimum of 5 threads.

If you use `DLTHREADSPREPARE` or `DLTHREADSWRITE`, it increases the minimum number of data load threads needed. For example, if you set `DLTHREADSPREPARE` to 4 and `DLTHREADSWRITE` to 2, you need 9 threads (the minimal 5, plus the increase of 4).

If the number of threads requested for all pipelines is 9, as in the case above, but `WORKERTHREADS` is set to only 8, then Essbase implicitly sets `DLTHREADSPREPARE` and `DLTHREADSWRITE` to 1, and the data load runs with a total thread requirement of 5 (the minimum and default).

- Similarly, if you run a parallel calculation, a parallel export, or a parallel restructure, these operations execute with parallelism not exceeding the number of available threads.

For example:

- If `WORKERTHREADS` is set to 8, and you attempt `CALCPARALLEL 16`, the parallel calculation runs as `CALCPARALLEL 8`.
- If `WORKERTHREADS` is set to 5, and you attempt `EXPORTTHREADS 12`, the parallel export runs as `EXPORTTHREADS 5`.
- If `WORKERTHREADS` is set to 16, and you attempt `RESTRUCTURETHREADS 32`, the parallel restructure runs as `RESTRUCTURETHREADS 16`.

- If you run concurrent parallel operations on the same application, the total parallelism is limited by the number of threads in the thread pool. For example, if you set `WORKERTHREADS` to 16, and run two concurrent requests of `CALCPARALLEL 16`, the two calculation requests share the 16 threads. The calculations will run using up to 16 threads, but because they share the thread pool of 16, they are running at close to half the capacity that was requested.
- If you have multiple cubes within an application, Oracle recommends setting `WORKERTHREADS` slightly higher than the parallel threads you need for one cube. For example, if `WORKERTHREADS` is set to 16 and the number of CPUs on the system is 16, you should set the `CALCPARALLEL` value to less than 16.
- If cubes on different servers reference each other using `@XREF` or `@XWRITE`, ensure that the source cube of the cross reference has a higher number of `SERVERTHREADS` than of parallel calculation threads (`CALCPARALLEL` or `FIXPARALLEL`).
- With thread pooling for parallel operations, increased resources are needed to start an Essbase application. If you encounter the error `Unable to Request Server Thread` during startup, adjust the operating system settings that control the number of threads per process and the total number of threads in the system (for example, on Linux, tune `nproc` using `ulimit`).

Example

`WORKERTHREADS` Sample 32

See Also

[CALCPARALLEL](#)

[DLTHREADSPREPARE](#)

[DLTHREADSWRITE](#)

[EXPORTTHREADS](#)

[FIXPARALLEL...ENDFIXPARALLEL](#) (calculation command block)

[RESTRUCTURETHREADS](#)

[SERVERTHREADS](#)

[Sample Configurations for Threaded Operations](#)

Configuration Settings Categorical List

This section lists the Oracle Essbase configuration settings, grouped categorically. Some may appear in more than one category.

- [Backup and Recovery Configuration Settings](#)
- [Calculation Configuration Settings](#)
- [Data Import and Export Configuration Settings](#)
- [Logging and Error Handling Configuration Settings](#)
- [Memory Management Configuration Settings](#)
- [Miscellaneous Configuration Settings](#)
- [Oracle Exalytics In-Memory Machine Configuration Settings](#)

- [Partitioning Configuration Settings](#)
- [Ports and Connections Configuration Settings](#)
- [Query Management Configuration Settings](#)

Backup and Recovery Configuration Settings

The following configuration settings are related to Essbase backup and audit functionality.

- [AUDITTRAIL](#)
- [REPLAYSECURITYOPTION](#)
- [SSAUDIT](#)
- [SSAUDITR](#)
- [TRANSACTIONLOGDATALOADARCHIVE](#)
- [TRANSACTIONLOGLOCATION](#)

Calculation Configuration Settings

The following configuration settings are related to Essbase calculation functionality.

- [AGGRESSIVEBLKOPTIMIZATION](#)
- [CALCCACHE](#)
- [CALCCACHEHIGH](#)
- [CALCCACHEDEFAULT](#)
- [CALCCACHELOW](#)
- [CALCLIMITFORMULARECURSION](#)
- [CALCMODE](#)
- [CALCNOTICE](#)
- [CALCOPTFRMLBOTTOMUP](#)
- [CALCPARALLEL](#)
- [CALCREUSEDYNCALCBLOCKS](#)
- [CALCTASKDIMS](#)
- [CALCTRACE](#)
- [CCTRACK](#)
- [CUSTOMCALCANDALLOCTHRUINSERT](#)
- [DYNCALCCACHEBLKRELEASE](#)
- [DYNCALCCACHEBLKTIMEOUT](#)
- [DYNCALCCACHECOMPRBLKBUFFSIZE](#)
- [DYNCALCCACHEMAXSIZE](#)
- [DYNCALCCACHEONLY](#)
- [DYNCALCCACHEWAITFORBLK](#)
- [ENABLERTSVLOGGING](#)

- EXCLUSIVECALC
- FORCEALLDENSECALCON2PASSACCOUNTS
- HYBRIDBSOINCALCSCRIPT
- IGNORECONSTANTS
- LONGCALCTIMETHRESHOLD
- MULTIPLEBITMAPMEMCHECK
- PARCALCMULTIPLEBITMAPMEMOPT
- RTDEPCALCOPTIMIZE
- UPDATECALC

Data Import and Export Configuration Settings

The following configuration settings are related to Essbase data load (import) and data export functionality.

- AUTOMERGE
- AUTOMERGEMAXSLICENUMBER
- DIMBUILDERERRORLIMIT
- DIMBUILDSTATSINTERVAL
- DLSINGLETHREADPERSTAGE
- DLTHREADSPREPARE
- DLTHREADSWRITE
- EXPORTTHREADS
- IDLETIMEMERGE
- MDXINSERTBUFFERAGGMETHOD
- MDXINSERTREQUESTTIMEOUT
- ODBCERRORLOGOFF
- RENEGADELOG
- RENEGADELOGLIMIT
- REJECTEDRECORDSLIMIT
- SQLFETCHERRORPOPUP

Failover Configuration Settings

The following configuration settings are related to Essbase failover / high availability functionality.

- AGENTLEASEEXPIRATIONTIME
- AGENTLEASERENEWALTIME
- FAILOVERMODE

Logging and Error Handling Configuration Settings

The following configuration settings are related to Essbase logging, tracking, and error handling functionality.

- CALCTRACE
- CRASHDUMP
- CRASHDUMPLOCATION
- DATAERRORLIMIT
- DELIMITEDMSG
- DELIMITER
- DIMBUILDERERRORLIMIT
- ENABLERTSVLOGGING
- EXCEPTIONLOGOVERWRITE
- GRIDEXPANSIONMESSAGES
- LOGMESSAGELEVEL
- LONGCALCTIMETHRESHOLD
- LONGQUERYTIMETHRESHOLD
- LONGREQTIMETHRESHOLD
- MAXERRORMBRVERIFYREPORT
- ODBCERRORLOGOFF
- OUTLINECHANGELOG
- OUTLINECHANGELOGFILESIZE
- QUERYTRACE
- REJECTEDRECORDSLIMIT
- RENEGADELOG
- RENEGADELOGLIMIT
- SQLFETCHERRORPOPUP

Memory Management Configuration Settings

The following configuration settings are related to Essbase memory management functionality.

- ASOLOADBUFFERWAIT
- DATACACHESIZE
- DYNCALCCACHEMAXSIZE
- ESTIMATEDHASHSIZE
- INDEXCACHESIZE
- MAXFORMULACACHESIZE
- MULTIPLEBITMAPMEMCHECK

- [NUMBLOCKSTOEXTEND](#)
- [PARCALCMULTIPLEBITMAPMEMOPT](#)
- [SSOPTIMIZEDGRIDPROCESSING](#)
- [SSPROCROWLIMIT](#)
- [TRIGMAXMEMSIZE](#)

Miscellaneous Configuration Settings

The following configuration settings are related to various Essbase optimizations.

- [MAXNUMBEROFACTIVEDB](#)
- [NUMBLOCKSTOEXTEND](#)
- [RESTRUCTURETHREADS](#)
- [TARGETTIMESERIESOPT](#)

Oracle Exalytics In-Memory Machine Configuration Settings

The following configuration settings are related to optimizing Essbase on Oracle Exalytics In-Memory Machine.

- [INPLACEDATAWRITE](#)
- [INPLACEDATAWRITEMARGINPERCENT](#)
- [MEMORYMAPPEDDATA](#)
- [ORACLEHARDWAREACCELERATION](#)

Partitioning Configuration Settings

The following Essbase configuration settings are related to working with transparent and replicated partitions.

- [DISABLEREPLMISSINGDATA](#)
- [ENABLE_DIAG_TRANSPARENT_PARTITION](#)
- [MAX_REQUEST_GRID_SIZE](#)
- [MAX_RESPONSE_GRID_SIZE](#)
- [REPLICATIONASSUMEIDENTICALOUTLINE](#)

Ports and Connections Configuration Settings

The following configuration settings are related to Essbase ports and connections functionality.

- [AGENTPORT](#)
- [AGENTSECUREPORT](#)
- [AGENTTHREADS](#)
- [CLIENTPREFERREDMODE](#)
- [ENABLECLEARMODE](#)
- [ENABLESECUREMODE](#)

- FORCESHUTDOWNINTERVAL
- MAXLOGINS
- SERVERPORTBEGIN
- SERVERPORTEND
- SERVERTHREADS
- SVRIDLETIME

Query Management Configuration Settings

The following configuration settings are related to Essbase query and grid functionality.

- ASODYNAMICAGGINBSO
- ASODYNAMICAGGINBSOFOLDERPATH
- GRIDEXPANSION
- GRIDEXPANSIONMESSAGES
- GRIDSUPPRESSINVALID
- HYBRIDBSOINCALCSCRIPT
- IGNORECONSTANTS
- LONGQUERYTIMETHRESHOLD
- LONGREQTIMETHRESHOLD
- MAXFORMULACACHESIZE
- MDXQRYGOVCOUNT
- QRYGOVEXECBLK
- QRYGOVEXECTIME
- QUERYRESULTLIMIT
- QUERYTRACE
- QUERYTRACETHRESHOLD
- SSANCESTORONTOP
- SSOPTIMIZEDGRIDPROCESSING
- SSPROCROWLIMIT
- SUPNA
- TARGETASOOPT
- TRACE_REPORT

See also [Query Logging Configuration](#), which you can enable by means of a separate configuration file.

Aggregate Storage and Block Storage Settings Comparison

Some Essbase configuration settings are applicable only to block storage (BSO) databases, some are applicable only to aggregate storage (ASO) databases, and some are applicable to both.

- [Block Storage and Aggregate Storage Configuration Settings](#)
- [Aggregate Storage Configuration Settings](#)
- [Block Storage Configuration Settings](#)

Block Storage and Aggregate Storage Configuration Settings

The following Essbase configuration settings apply to both aggregate storage (ASO) databases and block storage (BSO) databases.

- [AGENTTHREADS](#)
- [DLSINGLETHREADPERSTAGE](#)
- [DLTHREADSPREPARE](#)
- [GRIDEXPANSION](#)
- [GRIDEXPANSIONMESSAGES](#)
- [GRIDSUPPRESSINVALID](#)
- [MAXLOGINS](#)
- [MAXNUMBEROFACTIVEDB](#)
- [QRYGOVEXECTIME](#)
- [SERVERTHREADS](#)
- [SSANCESTORONTOP](#)
- [SSMEMBERIDPROCESSING](#)
- [SSOPTIMIZEDGRIDPROCESSING](#)
- [SSPROCROWLIMIT](#)
- [SUPNA](#)
- [SVRIDLETIME](#)
- [TARGETASOOPT](#)
- [TARGETTIMESERIESOPT](#)

Aggregate Storage Configuration Settings

The following Essbase configuration settings apply only to aggregate storage (ASO) databases, and do not apply to block storage (BSO) databases.

- [ASOBUFFERCOMMITWAIT](#)
- [ASOCACHECONCURRENTCONSUMINGTHREADS](#)
- [ASODEFAULTCACHESIZE](#)
- [ASODYNHIERASAGG](#)
- [AUTOMERGE](#)
- [AUTOMERGEMAXSLICENUMBER](#)
- [CUSTOMCALCANDALLOCTHRUINSERT](#)
- [DEFAULTVIEWBUILD](#)
- [DEFAULTVIEWBUILDSIZE](#)

- IDLETIMEMERGE
- MAX_REQUEST_GRID_SIZE
- MAX_RESPONSE_GRID_SIZE
- METADATABASEDAGGVIEWSBUILD
- TARGETASOOPT

Block Storage Configuration Settings

The following Essbase configuration settings apply only to block storage (BSO) databases, and do not apply to aggregate storage (ASO) databases.

- ASODYNAMICAGGINBSO
- ASODYNAMICAGGINBSOFOLDERPATH
- AUDITTRAIL
- CALCCACHE
- CALCCACHEHIGH
- CALCCACHEDEFAULT
- CALCCACHELOW
- CALCLIMITFORMULARECURSION
- CALCMODE
- CALCTRACE
- DATACACHESIZE
- DISABLEREPLMISSINGDATA
- DLTHREADSWRITE
- DYNCALCCACHEMAXSIZE
- ENABLERTSVLOGGING
- FORCEALLDENSECALCON2PASSACCOUNTS
- HYBRIDBSOINCALCSCRIPT
- INDEXCACHESIZE
- QRYGOVEXECBLK
- RESTRUCTURETHREADS

Sample Configurations for Threaded Operations

The following configuration samples accompany [Table 2-1](#). There is a sample for each system architecture.

64-bit (16 cores)

```
WORKERTHREADS 50
SERVERTHREADS 100
AGENTTHREADS 30
DLTHREADSPREPARE 2
```

```
DLTHREADSWRITE 2  
EXPORTTHREADS 8  
RESTRUCTURETHREADS 2
```

64-bit (32 cores)

```
WORKERTHREADS 50  
SERVERTHREADS 100  
AGENTTHREADS 30  
DLTHREADSPREPARE 2  
DLTHREADSWRITE 2  
EXPORTTHREADS 8  
RESTRUCTURETHREADS 2
```

64-bit (16 cores) and Two Databases

```
WORKERTHREADS 50  
SERVERTHREADS 100  
AGENTTHREADS 30  
DLTHREADSPREPARE 2  
DLTHREADSWRITE 2  
EXPORTTHREADS 8  
RESTRUCTURETHREADS 4
```

Oracle Exalytics In-Memory Machine, 40 core, X2-4

Block storage:

```
WORKERTHREADS 50  
SERVERTHREADS 100  
AGENTTHREADS 30  
DLTHREADSPREPARE 2  
DLTHREADSWRITE 2  
EXPORTTHREADS 40  
RESTRUCTURETHREADS 8
```

Aggregate storage:

```
WORKERTHREADS 50  
SERVERTHREADS 100  
AGENTTHREADS 30  
DLTHREADSPREPARE 24  
DLTHREADSWRITE 2  
EXPORTTHREADS 40  
RESTRUCTURETHREADS 8
```

Oracle Exalytics In-Memory Machine, SPARC 128 core

Block storage:

```
WORKERTHREADS 100  
SERVERTHREADS 200  
AGENTTHREADS 30  
DLTHREADSPREPARE 2  
DLTHREADSWRITE 2  
EXPORTTHREADS 40  
RESTRUCTURETHREADS 8
```

Aggregate storage:

```
WORKERTHREADS 100  
SERVERTHREADS 200  
AGENTTHREADS 30  
DLTHREADSPREPARE 24  
DLTHREADSWRITE 2  
EXPORTTHREADS 40  
RESTRUCTURETHREADS 8
```

Oracle Exalytics In-Memory Machine, 60 core, X4-4

Block storage:

```
WORKERTHREADS 60  
SERVERTHREADS 120  
AGENTTHREADS 30  
DLTHREADSPREPARE 2  
DLTHREADSWRITE 2  
EXPORTTHREADS 40  
RESTRUCTURETHREADS 8
```

Aggregate storage:

```
WORKERTHREADS 60  
SERVERTHREADS 120  
AGENTTHREADS 30  
DLTHREADSPREPARE 24  
DLTHREADSWRITE 2  
EXPORTTHREADS 40  
RESTRUCTURETHREADS 8
```


3

Provider Services Configuration

The Provider Services configuration properties are available in the Essbase web interface to help you manage network timeout parameters.

You can set the values for Provider Services configurations in the Console. See [Set Provider Services Configuration Properties](#).

essbase.jobs.maxCount

The Provider Services configuration property `essbase.jobs.maxCount` enables you to set the maximum number of parallel jobs that Essbase can run.

Syntax

```
essbase.jobs.maxCount n
```

n—Specifies the number of active jobs that Essbase can run in parallel. The default is 10.

Description

Jobs are operations you can run from the Essbase web interface. Jobs include loading data, building dimensions, exporting cubes, running MaxL scripts, running calculations, and clearing data. Jobs are asynchronous, meaning they are run in the background as a unique thread. Each job has a unique id.

This property is case-sensitive, so be sure to specify its name just as shown in the example.

Example

```
essbase.jobs.maxCount 15
```

olap.server.netRetryCount

The Essbase Provider Services configuration property `olap.server.netRetryCount` enables you to set number of times an API can retry a unsuccessful network operation before failing and reporting an error. If `olap.server.netSocketTryInfinite` is set to true, then `olap.server.netRetryCount` has no effect.

Syntax

```
olap.server.netRetryCount n
```

n—An integer value. The default is 600 retries. Every five tries gives a timeout of one second. The default value, 600, gives a timeout of two minutes.

olap.server.netSocketTimeOut

The Provider Services configuration property `olap.server.netSocketTimeOut` enables you to set the maximum time that an Essbase network operation (read/write) can be blocked before the operation I/O times out.

Syntax

```
olap.server.netSocketTimeOut n
```

n—Specifies the number of milliseconds an Essbase network operation can be blocked before the operation I/O times out.



Note:

This value affects overall request timeout behavior and should not be changed arbitrarily.

Example

```
olap.server.netSocketTimeOut 200
```

olap.server.netSocketTryInfinite

The Essbase Provider Services configuration property `olap.server.netSocketTryInfinite` enables you to set whether the client will keep trying infinitely on a network operation. If set to **true**, then `olap.server.netRetryCount` has no effect. The default is **false**.

Syntax

```
olap.server.netSocketTryInfinite true|false
```

system.cluster.monitorInterval

The Provider Services configuration property `system.cluster.monitorInterval` enables you to set the interval at which Essbase polls the server nodes in active-active (read only) clusters to determine if the nodes are down or active.

Syntax

```
system.cluster.monitorInterval n
```

n—The interval, in seconds, at which Essbase polls the nodes in active-active (read only) clusters to see if they are down or active. The default interval is 30s.

Example

```
system.cluster.monitorInterval 30
```

See Also

Access Multiple Essbase Servers From Smart View

4

Query Logging Configuration

Query logging provides a way for Essbase administrators to track query patterns of the cube. The query log file tracks queries performed against the cube from Smart View, Report Writer, or Grid-API clients.

Query logging can track generation or level numbers of members belonging to specific generations or levels. Query logging also offers the flexibility to exclude logging of certain dimensions and members belonging to certain generations or levels. Because the query log file output is an XML document, you can import the log file to any XML-enabled tool to view the log.

For details about the query log file structure, refer to `querylog.dtd` on the Essbase Server, in the `<Essbase Path>/bin` directory.



Note:

If you do not know where `<Essbase Path>` is in your environment, refer to Environment Locations in the Essbase Platform.

Query logging is available for block storage (BSO) and aggregate storage (ASO) cubes.

To enable query logging, create a query log file and add to the file the settings that control how query logging is performed.

You must create a query log file for each cube that requires query logging. If the query log file is missing or the QUERYLOG setting is off, query logging is disabled.

- [Enable Query Logging](#)
- [Query Log Settings File Syntax](#)
- [Query Logging Sample File](#)
- [Query Logging Sample Output](#)

Enable Query Logging

To enable query logging for an Essbase cube, create a query logging settings file in the cube directory, configure it, and restart the cube. Log entries will be written to a query log `.qlg` file, which you can save and view as xml.

The following steps explain how to create a query log settings file. To see a sample query log file, see [Query Logging Sample File](#).

To enable query logging:

1. In the cube directory, create a query log settings file.

The settings file must be named `dbname.cfg`, where `dbname` matches the name of the cube. For example, the query log settings file for Sample Basic is `basic.cfg`. As Essbase

21c applications are Unicode-mode, you must use a UTF-8-enabled text editor, and save as UTF-8 with BOM.

 **Note:**

The *cube directory* means `<Application Directory>/app/appname/dbname`. If you do not know where `<Application Directory>` is in your environment,

- Refer to Environment Locations in the Essbase Platform if you use an independent Essbase deployment.
- If you use an Essbase deployment on Oracle Cloud Infrastructure Marketplace, then `<Application Directory>` is `/u01/data/essbase/app`.

2. In the settings file, specify required and optional elements, using the syntax from the section [Query Logging Syntax](#):
 - The dimension for which you want to log queries (QUERYLOG [*dimension_name*]).
 - **Optional:** The setting to log generation or level numbers for members of specified generations or levels in a dimension (QUERYLOG GENERATION *generation-range* or QUERYLOG LEVEL *level-range*).
 - **Optional:** The setting to exclude logging of members from specified generations or levels in a dimension (QUERYLOG NONE GENERATION *generation-range* or QUERYLOG NONE LEVEL *level-range*).
 - **Optional:** The location where the query log file is created (QUERYLOG LOGPATH *path-expression*).
 - **Optional:** The format of the log file output (QUERYLOG LOGFORMAT CLUSTER | TUPLE).
 - **Optional:** The size of the log file (QUERYLOG LOGFILESIZE *n*).
 - **Optional:** The size of all log files (QUERYLOG TOTALLOGFILESIZE *n*).
 - A setting to enable or disable query logging the next time the application starts (QUERYLOG ON | OFF).
3. Restart the cube to accept the settings.

 **Note:**

Restart after creating a file or changing any entries in a file.

4. After query logging is enabled, review the log entries in the query log file, `dbname.qlg`.
For example, you can view the output of the log file to analyze how many times a certain member has been queried. Use a UTF-8-enabled editor to view the log files.

Query Log Settings File Syntax

To configure query logging for an Essbase cube, configure the query logging settings file using QUERYLOG parameters to specify which dimensions and members you want to track, and how you want the information logged.

The query log settings filename must be of the form *dbname.cfg*, where *dbname* represents the name of the database (cube). The *dbname.cfg* file must be located in the cube directory. Configure the *dbname.cfg* file using the following syntax:

```
QUERYLOG [dimension_name]
QUERYLOG NONE GENERATION generation-range
QUERYLOG NONE LEVEL level-range
QUERYLOG GENERATION generation-range
QUERYLOG LEVEL level-range
QUERYLOG LOGPATH path-expression
QUERYLOG LOGFORMAT CLUSTER | TUPLE
QUERYLOG LOGFILESIZE n
QUERYLOG TOTALLOGFILESIZE n
QUERYLOG ON | OFF
```

Note:

The *cube directory* means *<Application Directory>/app/appname/dbname*. If you do not know where *<Application Directory>* is in your environment,

- Refer to Environment Locations in the Essbase Platform if you use an independent Essbase deployment.
- If you use an Essbase deployment on Oracle Cloud Infrastructure Marketplace, then *<Application Directory>* is */u01/data/essbase/app*.

Table 4-1 QUERYLOG Parameters

QUERYLOG Parameter	Description
<i>[dimension_name]</i>	Identifies the dimension name to be tracked. The brackets around the dimension name are required. QUERYLOG <i>[dimension_name]</i> logs all members of a dimension. For example, QUERYLOG [Product] tracks all members of the Product dimension. Each dimension must be specified in a separate QUERYLOG <i>[dimension_name]</i> setting.
NONE GENERATION <i>generation-range</i>	Prevents tracking of members from the specified generation range. For example, QUERYLOG NONE GENERATION 2 excludes tracking of all members from generation 2 of the named dimension.
NONE LEVEL <i>level-range</i>	Prevents tracking of members from the specified level range. For example, QUERYLOG NONE LEVEL 0-2 excludes tracking of all members of levels 0, 1, and 2 of the named dimension.

Table 4-1 (Cont.) QUERYLOG Parameters

QUERYLOG Parameter	Description
GENERATION generation-range	Tracks members of the specified generation range by generation number, rather than by member name. For example, QUERYLOG GENERATION 5-7 logs members of generations 5, 6, and 7 of the named dimension by their generation number in the log file.
LEVEL level-range	Tracks members of the specified level range by level number, rather than by member name. For example, QUERYLOG LEVEL -3 logs members of levels 0, 1, 2, and 3 of the named dimension by their level number in the log file.
LOGPATH <i>path-expression</i>	Specifies the location of the output log file. The log file name is <i>dbname00001.qlg</i> ; for example, <i>basic00001.qlg</i> . Examples of the log path are QUERYLOG LOGPATH <i>/usr/local/Essbaselogs/</i> and QUERYLOG LOGPATH <i>d:\Essbaselogs\querylogs\</i> . You must include a backslash \ (for Windows directories) or forward slash / (for UNIX directories) at the end of the path expression; otherwise, the query log file is not created. By default, the location for the log output file is the cube directory. If the LOGPATH <i>path-expression</i> setting is missing, the default is used. Essbase writes log information to the query log file after an application stops running.
LOGFORMAT CLUSTER TUPLE	Specifies the format of the log output. CLUSTER and TUPLE provide the same log information, but display the information differently. CLUSTER provides information on how many members of a dimension were queried and lists queried members within their respective dimensions. TUPLE lists each queried member combination. By default, CLUSTER is the log format. Because the TUPLE format lists each member combination queried, TUPLE may have a greater impact on query performance than CLUSTER. See Sample Cluster Output for an example of a query log in cluster format. See Sample Tuple Output for an example of a query log in tuple format.
LOGFILESIZE <i>n</i>	Specifies the maximum size of an individual query log file in megabytes (MB). The minimum value is 1 MB. The maximum value is 2048 MB (2 GB). If the LOGFILESIZE setting is missing, then, by default, the query log file size is 1 MB. If an initial query log file size exceeds the specification, log information is added to a new query log file. Each time a new file is created, the filename is incremented by one.

Table 4-1 (Cont.) QUERYLOG Parameters

QUERYLOG Parameter	Description
TOTALLOGFILESIZE <i>n</i>	Specifies the maximum size of all query log files combined in megabytes (MB). The minimum value is 512 MB (1/2 GB). The maximum value is 4095 MB. If the TOTALLOGFILESIZE setting is missing, then, by default, the total query log file size is 1024 MB (1 GB). Query log files are created until the file size total exceeds the specified maximum. When the maximum is exceeded, a message is displayed and query logging automatically turns off.
ON OFF	Specifies whether the query logging feature is turned on or off. All query log settings are ignored if this setting is OFF or missing. By default, the setting is OFF.

Generation-range and *level-range* values are represented in one of the following ways:

Table 4-2 Generation and Level Range Specifications

Generation-Range or Level-Range Value	Description
<i>x</i>	A specific generation or level number. For example, QUERYLOG NONE GENERATION 2 excludes generation 2 from query logging.
<i>x-y</i>	All generations or levels inclusive of number <i>x</i> through number <i>y</i> . For example, QUERYLOG GENERATION 1-3 or QUERYLOG LEVEL 1-3 includes generation or level numbers 1, 2, and 3.
<i>-x</i>	For <i>generation-range</i> , all generations within the range 1 through <i>x</i> . For <i>level-range</i> , all levels within the range 0 through <i>x</i> . For example, QUERYLOG GENERATION -2 includes generations 1 and 2. QUERYLOG LEVEL -3 includes levels 0, 1, 2, and 3.
<i>x-</i>	For <i>generation-range</i> , all generations within the range from number <i>x</i> through the highest generation. For <i>level-range</i> , all levels within the range from number <i>x</i> through the highest level. For example, QUERYLOG Level 1- includes levels 1, 2, 3 and so on up to the highest level.

Notes

- When query logging is enabled, queries to the cube may be slower. Performance depends on how many members are being tracked and the size of the query.
- If the settings file name does not match the name of the cube, or is located in a place other than the cube directory, Essbase ignores query logging.
- If, in the settings, QUERYLOG ON is missing or if QUERYLOG OFF is set, then query logging is off.
- If generation and level settings cause contradictions in the settings file, the following precedence rules apply:
 - generation numbers (highest priority)

- level numbers
- member names (lowest priority)

For example, if a member belongs to both level 1 and generation 2 and the settings `QUERYLOG GENERATION 2` and `QUERYLOG NONE LEVEL 1` are in the settings file, the generation setting takes precedence, and members of generation 2 are logged by generation number.

Tips

- To view query log output easily, change the file extension `.qlg` to `.xml`, and use a browser to view the XML.
- If Essbase is not producing a query log `.qlg` file as expected, view the `dbname.log` file in the cube directory to search for query log messages.

Query Logging Sample File



Note:

indicates a comment that describes a line of the settings file. Comments are not necessary to include in the actual query log settings file.

```
# Log the Product dimension
QUERYLOG [Product]
# Log the Market dimension
QUERYLOG [Market]
# Log members of generation 2 of Market by generation number
QUERYLOG GENERATION 2
# Display log output in cluster format
QUERYLOG LOGFORMAT CLUSTER
# Create log file in C:\QUERYLOG\
QUERYLOG LOGPATH C:\QUERYLOG\
# Start a new log file after an individual log file size reaches 2 MB
QUERYLOG LOGFILESIZE 2
# Turn off query logging after the total size of all log files reaches 1024
MB (1 GB)
QUERYLOG TOTALLOGFILESIZE 1024
# Enable query logging
QUERYLOG ON
```

Query Logging Sample Output

The following Essbase Sample Query Log Output segment shows an example of how log settings look in a log file.

In the example, the log settings show that all members of Product are logged and that members of generation 2 of Market are logged by generation number. The log format is cluster and the log path is C:\QUERYLOG\.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <root>
  - <session>
    <bootuptime>Wed Jul 23 15:27:26 2002</bootuptime>
  - <logsettings>
    - <dimensions>
      - <logdim name="Product">
      - <logdim name="Market">
        <spec>GENERATION 2</spec>
      </logdim>
    </dimensions>
  - <othersettings>
    <logformat>cluster</logformat>
    <logpath>C:\QUERYLOG\</logpath>
  </othersettings>
</logsettings>
```

Description

A query is a unit of retrieval from the user perspective. The way a user may perceive a query is different than how the server analyzes and executes a query. Even if a user performs a single retrieval, in order for the server to efficiently execute the logical query, the server splits the query into a number of subqueries to execute. Therefore, a single retrieval from the user perspective may actually consist of several subqueries from the server perspective. These subqueries are reflected in the query log.

Sample Cluster Output

The following segment shows an example of how queries are logged in cluster format. The username is listed along with the query execution date and the start time of the query. Each cluster contains two dimension entries. The first cluster shows that members 100 and 200 of the Product dimension were queried. The second cluster shows that member 300 of Product and Generation 2 of Market were queried. The elapsed time to perform the query is also provided.

```
<query>
  <user>User1</user>
  <time>Tue Aug 13 12:29:49 2002</time>
  <subquery>
    <cluster size="2">
      <dim size="2">
        <member>100</member>
        <member>200</member>
      </dim>
```

```
<dim size="1">
  <member>Market</member>
</dim>
</cluster>
</subquery>
<subquery>
<cluster size="2">
  <dim size="1">
    <member>300</member>
  </dim>
  <dim size="2">
    <member>Market</member>
    <generation>2</generation>
  </dim>
</cluster>
</subquery>
<elapsedtime>0.016 seconds</elapsedtime>
</query>
```

Sample Tuple Output

The following segment shows an example of how queries are logged in tuple format. The username is listed along with the query execution date and the start time of the query. Note that each member of Product is displayed with Market. Each possible member combination is displayed for a given query. The elapsed time to perform the query is also provided.

```
<query>
  <user>User1</user>
  <time>Tue Aug 13 12:28:14 2002</time>
  <subquery>
    <tuples>
      <tuple>
        <member>100</member>
        <member>Market</member>
      </tuple>
    </tuples>
  </subquery>
  <subquery>
    <tuples>
      <tuple>
        <member>200</member>
        <member>Market</member>
      </tuple>
    </tuples>
  </subquery>
  <elapsedtime>0.02 seconds</elapsedtime>
</query>
```