

# CA Gen

## Client Server Encyclopedia Administration Guide

Release 8.5



This Documentation, which includes embedded help systems and electronically distributed materials, (hereinafter referred to as the "Documentation") is for your informational purposes only and is subject to change or withdrawal by CA at any time.

This Documentation may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Documentation is confidential and proprietary information of CA and may not be disclosed by you or used for any purpose other than as may be permitted in (i) a separate agreement between you and CA governing your use of the CA software to which the Documentation relates; or (ii) a separate confidentiality agreement between you and CA.

Notwithstanding the foregoing, if you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all CA copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is CA.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Copyright © 2013 CA. All rights reserved. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

## CA Technologies Product References

This document references the following CA Technologies products:

- CA Gen

## Contact CA Technologies

### Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At <http://ca.com/support>, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

### Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to [techpubs@ca.com](mailto:techpubs@ca.com).

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at <http://ca.com/docs>.



# Contents

---

## Chapter 1: Introduction 9

Identify Bottlenecks among System Resources .....	9
Optimize CPU Usage .....	10
Optimize Memory Usage .....	10
Optimize Disk Usage .....	11
Optimize Network Usage .....	12
Anticipate and Avoid Problems .....	12

## Chapter 2: Collecting Performance Statistics 15

Monitor the Encyclopedia Tables .....	15
Oracle .....	15
SQL Server .....	17
Monitor the Encyclopedia Profile .....	18
Monitor Operating System Resources .....	19
Monitor Encyclopedia Usage .....	19
Monitor Encyclopedia Contention .....	20

## Chapter 3: Capacity Planning 21

Become Familiar with Table Sizes .....	21
Host Versus CSE Table Comparison .....	21
Schema Tables .....	21
Small Tables for User Data .....	22
Variable-Sized Tables for User Data .....	23
Large Tables for User Data .....	23
Relative Size of Indexes to Tables .....	23
Estimate CSE Size .....	24
Split Files Across Multiple Physical Drives .....	25
Capacity Planning Checklist .....	26

## Chapter 4: Tuning the Client Server Encyclopedia 29

Client Server Encyclopedia Processes Affecting Performance .....	29
Resource Intensive Processes .....	29
Enable Parallel Processing .....	30
Backup and Recovery of the CSE Tables .....	30
Backup Options .....	30

---

Table Backup/Recovery with Disk Device Contents .....	30
Table Backup/Recovery Without Update Logs .....	30
Table Backup/Recovery with Update Logs .....	31
Guidelines for Scheduling Backups .....	31
Performing CSE Cleanup .....	31
Cleaning Up the CSE .....	32
Maintaining CSE Tables and Indexes .....	32
Cache and Threshold Settings .....	33
Object Cache .....	33
Override Checkout Threshold .....	34

## **Chapter 5: Managing the Client Server Encyclopedia Workload 35**

Ways to Improve Processes .....	35
Causes of Contention .....	35
DBMS Timeouts and Deadlocks .....	36
Encyclopedia Enqueue Waits .....	36
Avoiding Contention and Model Corruption.....	37
What Is Currently Running? .....	37
What Is Scheduled to Run?.....	40
Communicate What Is Going to Run .....	41
Database Object ID Partitioning.....	41
Peak and Non-Peak Processing Times .....	42
Peak Processing .....	42
Non-Peak Processing.....	42
Subject Matter Experts.....	44
Subsetting Expert .....	44
Migration Expert.....	45
Reports .....	45
Determine Objects Needing Migrating .....	46
Determine Potential Checkout Downgrades .....	46
Examine Listings .....	46
Project Planning and Coordination.....	46
Naming Standards.....	47
Project Models.....	48
Coordinating Shared Objects .....	48
Avoiding Mass Upload.....	48
Regeneration and Installation.....	49
Specific Encyclopedia Functions .....	49
Download.....	49
Upload.....	50
Override Checkout Status .....	51

---

Migration .....	52
Adoption .....	53
Adding a New Model .....	54
Model Delete .....	55

## Chapter 6: Monitoring and Tuning for Specific Platforms 57

Oracle.....	57
Routine Database Cleanup .....	57
NEXT and PCTINCREASE Reduction.....	58
Database Defragmentation .....	58
CSE Database Partitioned Across Multiple Drives .....	60
DDL Customization.....	60
SQL Server .....	61
Procedure Cache .....	61

## Chapter 7: Database Administration for Specific Platforms 63

SQL Server/Windows Routine Procedures .....	63
Cleaning Up Your Database.....	63
Updating Database Statistics .....	64
Backing Up Your Database.....	66
Backing Up Transaction Logs.....	67
Adjusting Database Size.....	68
Truncating Transaction Logs .....	69
Truncating Transaction Log Guidelines .....	69
Stopping SQL Server.....	70
Starting SQL Server .....	70
SQL Server/Windows Database Customization.....	71
Adjusting Storage Parameters .....	71
Distributing Data Files Across Disks .....	71
Accessing Public Interface Views .....	72
Setting Options to Improve Performance.....	72
Common SQL Server System Procedures .....	72
Oracle/Windows Routine Procedures .....	73
Cleaning Up Your Database.....	73
Backing Up Your Database.....	74
Adjusting Database Size.....	74
Stopping the Database.....	75
Starting the Database.....	75
Oracle/Windows Database Customization Procedures .....	76
Adjusting the Rollback Segments and Rollback Tablespace .....	76
Adjusting the init Encyclopedia database name.ora Oracle Windows .....	77

---

Adjusting the Number and Size of Log Files .....	77
Adjusting Storage Parameters .....	77
Distributing Data Files Across Disks .....	77
Accessing Public Interface Views .....	78
UNIX/Oracle Routine Procedures .....	79
Cleaning Up Your Database .....	79
Backing Up Your Database .....	80
Adjusting Database Size .....	80
Stopping the Database .....	81
Starting the Database .....	81
UNIX/Oracle Database Customization Procedures .....	82
Adjusting the Rollback Segments and Rollback Tablespace .....	82
Adjusting the init<Encyclopedia database name>.ora .....	83
Adjusting the Number and Size of Log Files .....	83
Adjusting Storage Parameters .....	84
Distributing Data Files Across Disks .....	84
Accessing Public Interface Views .....	85

## Chapter 8: Server Administration 87

What Happens If a Server Stops? .....	87
Working with the <iefmd>.ini File .....	88
What Is the <iefmd>.ini File? .....	88
Defining an Exclusive Message Dispatcher for the Coordination Server .....	90
Changing the DIRGROUP and ENCYGROUP Parameters .....	90
Assigning Port Numbers to Message Dispatchers .....	90
Starting Multiple Instances of Encyclopedia Utilities .....	91
Capturing Server Utility Errors During Unexpected Shutdowns .....	95
Determining Status of a Server .....	95
Setting the CSE LogLevel .....	100
LogLocation .....	101
NbrLogFiles .....	101
LimitEachLog .....	101
LogLevel .....	102

## Chapter 9: Troubleshooting 105

Common Performance Questions .....	105
------------------------------------	-----

## Index 109



# Chapter 1: Introduction

---

This section contains the following topics:

[Identify Bottlenecks among System Resources](#) (see page 9)

[Optimize CPU Usage](#) (see page 10)

[Optimize Memory Usage](#) (see page 10)

[Optimize Disk Usage](#) (see page 11)

[Optimize Network Usage](#) (see page 12)

[Anticipate and Avoid Problems](#) (see page 12)

## Identify Bottlenecks among System Resources

The goal of adjusting system resource usage is to satisfy the user throughput and response times required in your business. You can optimize system performance in a client/server (C/S) environment by overcoming potential bottlenecks of the four components of your server machine environment shown in the following table:

Component	Bottleneck
CPU	More requests are made than can be satisfied within a given timeframe
Memory	It is insufficient to accommodate processing requirements, resulting in memory paging and swapping
Disk I/O	The number of requested read/write operations exceed capacity
Network	The volume of traffic exceeds capacity and causes collisions

Because these components interact with each other, improving one may have a positive affect on the others. For example, when you add needed memory, you can expect excessive CPU paging and swapping to stop. On the other hand, making an improvement in one area may cause a bottleneck to develop in another area. For example, adding buffers improves disk I/O but decreases available memory. Ideally, tuning should result in a high degree of resource utilization without overtaxing any one of the individual components. The key is to get all of these interdependent components to work together. You may find it beneficial to consult other groups that have similar configurations to yours about their use of CPU, memory, disk, and network.

## Optimize CPU Usage

CPU problems normally occur when too many processes are requesting service at the same time. The first sign of CPU problems is when processes begin to take a lot longer than usual to complete. In extreme cases, thrashing occurs. Thrashing occurs when the CPU becomes so consumed with moving memory back and forth from disk that no time is left for processing user requests. Then, programs are placed in a wait state for CPU.

To correct minor CPU problems like slow processing and avoid severe CPU problems like thrashing, consider the following tips:

- Enable parallel processing, if possible, by spreading concurrent processing across multiple CPUs/servers. The more processes sharing a single CPU, the higher the risk of a CPU shortage.
- Direct processing of the largest jobs to the fastest CPU. Process elapsed time tends to decrease as CPU speed increases.
- Adjust priority settings to ensure time-critical processes are run first.
- Schedule long running encyclopedia processes to execute during non-peak hours, where practical. For example, a process, which causes a sequential search algorithm against a large model, is better suited to execute during off-hours.
- To the extent practical, spread the processing workload evenly across time. Establish a site schedule to coordinate what encyclopedia processes are run at different times.
- Add memory to reduce swapping and paging demands on the CPU.
- Optimize memory usage and disk usage.
- Get a faster CPU.

## Optimize Memory Usage

Memory problems can occur when there is not enough free memory available to accommodate peak processing without paging and swapping. When free memory is exhausted, the entire process is paged or swapped to disk. Since disk access is much slower than memory access, performance suffers when this occurs.

The amount of memory needed depends on the program running, the language of that program, the DBMS accessed, the operating system, and the number of users logged on. The more memory available, the faster the system will run.

Memory usage is considered optimized when the hit ratio is 90 percent or better. The hit ratio is the ratio of finding rollback, table, index, and cluster data in memory rather than having to perform I/O from disk. The goal is to achieve as close to 100 percent utilization as possible during peak processing times.

The key point here is that more available memory translates into reduced requirements for swapping, thus improving overall system throughput.

These benefits are especially notable when processing a large model.

To avoid memory problems, especially when processing a large model:

- Equip your system so that free memory will not drop below 5 percent during peak processing times. (Leaving too much free memory, say 10 percent, is inefficient.)
- Adjust the CSE object cache, that is, the number of unreferenced objects kept in memory.
- Add memory when shortages begin to impact performance and it looks like a pattern is developing.

## Optimize Disk Usage

Disk problems can occur when disk I/O activity is high. Common problems include disk fragmentation and chaining.

To avoid disk problems, especially when processing a large model:

- Spread the workload as evenly as possible across multiple disk drives to reduce physical I/O contention by allowing parallel I/O operations. For example, if you have four physical disk drives, you could separate the operating system, database log files, CSE tables, and CSE indexes. See the table Recommended Configuration for Optimal Disk Performance.
- If you have only a single drive, consider obtaining additional drives. The more disk drives you have, the less time processes spend waiting for disk I/O operations.
- Strive to ensure that all disks are operating within their recommended maximum I/O rates.
- If only certain disks exceed recommended maximum I/Os, move tablespace from the overworked disks to other disks.
- Ensure that tables and indexes are assigned storage locations of adequate size. If sized correctly, all data will be in one contiguous extent rather than fragmented. Unfragmented data provide the most efficient access by eliminating the need for extra read/write head movement.
- Plan for future expansion. If you need additional disks, consider adding several medium size disks rather than a few large disks to allow for better sharing of the workload.
- Add memory for I/O buffering to reduce disk activity.
- Adjust buffer caches as needed, either by adding more or reducing the current number.

- Provide ample amounts of embedded free space.
- Ensure that an adequate amount of system sort workspace is provided.
- If available for your hardware platform, take advantage of Array disk technology.

**Note:** While it is possible to configure the CSE environment such that the database is located on a different physical machine, it is not a recommended configuration for performance reasons. The architecture of the normal execution of the CSE process is such that there are a very large number of individual messages passed between the CSE process and the database process. This leads to performance issues since any network latency between the two machines will be multiplied by the number of requests made. If the current configuration is absolutely necessary, then care should be taken to ensure that the communications infrastructure between the two machines is as quick as possible.

## Optimize Network Usage

Network problems, such as collisions, can occur when the volume of data to be transferred exceeds network capacity.

**Follow these steps:**

- Ensure your network is fast enough to meet your needs.
- Position your equipment strategically. Moving the client and server machines closer together physically will reduce network delays.
- Schedule long-running processes, such as large checkin or checkout processes, during non-peak times to improve overall response time.
- Use your operating system's data compression utilities to compress data before transferring it across the network.
- Use disk drives directly mounted on your computer. Avoid use of drives attached over a network.

## Anticipate and Avoid Problems

Certain problems can be avoided, after you know when they occur. The following table lists the problems and their solutions.

Problem	Occurs when
Rollback extension	A transaction has to extend its size because the current rollback is not large enough to contain all the rollback data

Problem	Occurs when
Rollback contention	There are not enough rollback segments; therefore some transactions requesting rollbacks must wait for a rollback segment
Chaining and dynamic extension	The current extents assigned to tables, indexes, and rollback segments are insufficient to handle growth
Excessive I/O on hard drives	Data files are not spread among disks, resulting in degraded performance
Memory paging	Memory is insufficient to accommodate processing requirements; contributes to excessive I/O on hard drives



# Chapter 2: Collecting Performance Statistics

---

This chapter does the following:

- Provides queries you can use to collect statistics on the encyclopedia tables
- Provides queries you can use to collect statistics on the encyclopedia profile
- Suggests strategies for monitoring operating system resources, encyclopedia usage, and encyclopedia contention.

## Monitor the Encyclopedia Tables

Queries for monitoring the system tables are provided for the following DBMSs:

- Oracle UNIX and Windows
- SQL Server

### Oracle

Use the following SQL queries to gather statistics from system tables in an Oracle server.

#### Oracle Windows

For the query to display the number and size of database files and the number of extents for CSE tables, set the environment variable `ORACLE_SID` to the CSE database; for example `DBCSE`. Enter the following at the command prompt:

```
SET ORACLE_SID=DBCSE
```

`sqlplus` must be executed from this command prompt.

Alternatively, in your workspace, click **Start, Setting, Control Panel**. In the **Control Panel**, select **System**. In the **System Properties** panel, add the environment variable `ORACLE_SID` and its value, the database name. Click **Set** to set the value. `sqlplus` can be run from any prompt or the icon.

For the query to display next extent for each table and index in the `DBCSE` database, run the SQL from within `sqlplus` after logging in as owner of the `DBCSE` tables.

## Oracle UNIX

For the query to display the number and size of database files and the query to display the number of extents for CSE tables, set the environment variable `ORACLE_SID` to the CSE database; for example *DBCSE*.

If you are using the Bourne shell:

```
ORACLE_SID=DBCSE
export DBCSE
```

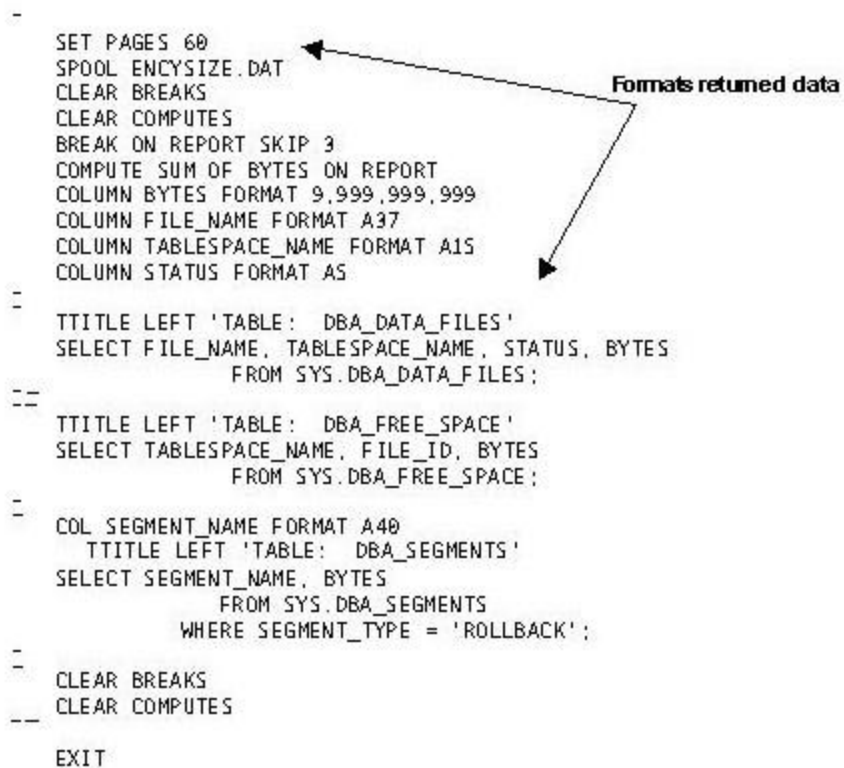
If you are using the Korn shell:

```
export ORACLE_SID=DBCSE
```

Or if you are using c-shell

```
setenv ORACLE_SID DBCSE
```

Then log in to the system account as owner of the DBCSE tables to access the DBA tables. For the query to display next extent for each table and index in the DBCSE database, run the SQL from within sqlplus.



The screenshot shows a SQL script being executed in a terminal. The script is enclosed in a box. Two arrows point from the text "Formats returned data" to the column format specifications in the script. The first arrow points to the format for the 'BYTES' column in the first query, and the second arrow points to the format for the 'BYTES' column in the second query.

```
-
SET PAGES 60
SPOOL ENCYSIZE.DAT
CLEAR BREAKS
CLEAR COMPUTES
BREAK ON REPORT SKIP 3
COMPUTE SUM OF BYTES ON REPORT
COLUMN BYTES FORMAT 9,999,999,999
COLUMN FILE_NAME FORMAT A37
COLUMN TABLESPACE_NAME FORMAT A15
COLUMN STATUS FORMAT A5
-
TTITLE LEFT 'TABLE: DBA_DATA_FILES'
SELECT FILE_NAME, TABLESPACE_NAME, STATUS, BYTES
FROM SYS.DBA_DATA_FILES;
--
TTITLE LEFT 'TABLE: DBA_FREE_SPACE'
SELECT TABLESPACE_NAME, FILE_ID, BYTES
FROM SYS.DBA_FREE_SPACE;
-
COL SEGMENT_NAME FORMAT A40
TTITLE LEFT 'TABLE: DBA_SEGMENTS'
SELECT SEGMENT_NAME, BYTES
FROM SYS.DBA_SEGMENTS
WHERE SEGMENT_TYPE = 'ROLLBACK';
-
CLEAR BREAKS
--
CLEAR COMPUTES

EXIT
```



The number of Extents on CSE tables and indexes:

```

SET PAGES 58
SET FEEDBACK OFF
CLEAR BREAKS
CLEAR COMPUTES
BREAK ON REPORT SKIP 3
COMPUTE SUM OF BYTES ON REPORT
COLUMN TABLESPACE_NAME FORMAT A15
COLUMN SEGMENT_NAME FORMAT A15
COLUMN BYTES FORMAT 9,999,999,999
--
SPOOL ENCY_EXTENTS.DAT
--
TTITLE LEFT 'TABLES OR INDEXES WITH 1 OR MORE EXTENTS.'
SELECT SEGMENT_NAME, COUNT (*) "NUM EXTENTS" FROM SYS.DBA_EXTENTS
        WHERE OWNER='ENCY' AND EXTENT_ID > 0
GROUP BY SEGMENT_NAME
ORDER BY SEGMENT_NAME;
--
EXIT

```

Next Extent for CSE tables and indexes:

```

SET PAGES 58
SET FEEDBACK OFF
CLEAR BREAKS
CLEAR COMPUTES
BREAK ON REPORT SKIP 3
COMPUTE SUM OF BYTES ON REPORT
COLUMN TABLESPACE_NAME FORMAT A15
COLUMN SEGMENT_NAME FORMAT A15
COLUMN BYTES FORMAT 9,999,999,999
--
SPOOL ENCY_EXTENTS.DAT
--
TTITLE LEFT 'TABLES OR INDEXES WITH 1 OR MORE EXTENTS.'
SELECT SEGMENT_NAME, COUNT (*) "NUM EXTENTS" FROM SYS.DBA_EXTENTS
        WHERE OWNER='ENCY' AND EXTENT_ID > 0
GROUP BY SEGMENT_NAME
ORDER BY SEGMENT_NAME;
--
EXIT

```

## SQL Server

Use the Microsoft SQL Server Management Studio and its associated documentation to access information about the CSE tables and indexes. You can use the Management Studio or the available command line tools to gather some basic statistics.

Number of rows in common encyclopedia tables:

```
SELECT COUNT (*) FROM DOBJ
SELECT COUNT (*) FROM DASC
SELECT COUNT (*) FROM DTXT
SELECT COUNT (*) FROM DSUBEX
```

## Monitor the Encyclopedia Profile

Use the following SQL queries to gather statistics about the encyclopedia profile.

Number of models in encyclopedia:

```
SELECT COUNT (*)
FROM DMDL
```

Number of models updated during last month:

```
SELECT COUNT (*)
FROM DMDL
MODEL_DATE > CURRENT DATE - 1 MONTH
```

Size of models and number of objects:

```
SELECT MODEL_NAME, COUNT (*)
FROM DMDL,
DOBJ
WHERE OBJ_MODEL_ID = MODEL_ID
GROUP BY MODEL_NAME
ORDER BY 2 DESC
```

Number of objects in model and subset checkouts:

```
SELECT MODEL_NAME, S_SUBSET_NAME, CKO_USER_ID, COUNT (*)
FROM DMDL, DCKOID, DSUBID, DSUBEX
WHERE CKO_MODEL_ID = MODEL_ID
AND CKO_STATUS = 'C'
AND CKO_SUBSET_ALL = 'N'
AND CKO_SUBSET_ID = S_SUBSET_ID
AND SE_CKO_ID = CKO_ID
GROUP BY MODEL_NAME, S_SUBSET_NAME, CKO_USER_ID
UNION
SELECT MODEL_NAME, 'ALL', CKO_USER_ID, COUNT (*)
FROM DMDL, DCKOID, DSUBEX
WHERE CKO_MODEL_ID = MODEL_ID
AND CKO_STATUS = 'C'
AND CKO_SUBSET_ALL = 'Y'
AND SE_CKO_ID = CKO_ID
GROUP BY MODEL_NAME, CKO_USER_ID
```

User IDs authorized to encyclopedia:

```
SELECT U_USER_TYPE, COUNT (*)
FROM DUSR
GROUP BY U_USER_TYPE
```

## Monitor Operating System Resources

Statistics can help you monitor and optimize encyclopedia performance. Use DBMS-specific monitoring utilities to monitor operating system memory and CPU use, since these are environment specific.

## Monitor Encyclopedia Usage

To determine elapsed time and related details, examine utility and server log files.

Log files display start time, end time, user ID, and, when appropriate, model name and subset name. The Message Dispatcher .ini file contains the prefix of log file names for each server and utility. The number is appended before and after the type separator.

The CSE log file base names are specified in the Message Dispatcher INI file, %IEF\_CONFIGDIR%\iefmd.ini on Windows or \$IEF\_CONFIGDIR/iefmd.ini on UNIX, as the value of the LogLocation parameter. These names are constructed from the value specified for the working directory, IEF\_WORKINGDIR, also specified during CSE configuration. For instance, for the upload utility APPNAME=UTLUP, the base name is <ief\_workingdir>\up; where <ief\_workingdir> is the value of IEF\_WORKINGDIR. The first instance of the upload utility writes its log information into "<ief\_workingdir>\up1.01".

## Monitor Encyclopedia Contention

The monitoring of contention caused by timeouts and deadlocks is to overall encyclopedia use. Often, as deadlines approach for a particular project, timeouts and deadlocks increase as the work related to a model increases. Many users attempt to funnel work to one place. Since all DBMSs protect data through locking mechanisms, some users experience timeouts or deadlocks.

# Chapter 3: Capacity Planning

---

Each CSE contains two types of tables: schema tables and data tables. In this chapter, all tables are referred to by their short names.

## Become Familiar with Table Sizes

Tables that are dependent on what DBMS or operating system you use include:

- Schema tables
- User data tables that are small
- User data tables that vary in size
- User data tables that are large

## Host Versus CSE Table Comparison

Most tables composing the Host encyclopedia have counterparts on the CSE. Most CSE tables are about the same size as the corresponding tables on the Host. The significant exception is the DOBJ table, or Object table, which is much larger on the CSE.

### DOBJ Table

The CSE DOBJ row size is 350% longer than the size of a Host encyclopedia row, since properties are stored on this table rather than a separate DPRP table, as they are on the Host.

### DSUBEX Table

The CSE DSUBEX table, or Subset Expansion table, stores details only on objects in checked out subsets, not on objects in checked out models. The Host encyclopedia DSUBEX table stores details on objects in both checked out subsets and checked out models.

## Schema Tables

The following schema tables are never resized:

- SASC
- SDIV

- SOBJ
- SPRP
- STRG

## Small Tables for User Data

The following user data tables are small and relatively static in size:

Size	Depends on the number of
DGRPUS	groups and users defined for the encyclopedia
DMAX	N/A; size is stable
DMDLUS	users authorized to access or update each model
DSETID	aggregate sets created; stores details including set name, owner creation date and time, administrator ID
DSETDF	original object IDs in each aggregate set; stores original encyclopedia and object IDs for each aggregate set
DUSR	users defined for the encyclopedia
DMDL	models defined for the encyclopedia
DSUBID	subsets created; stores details including ID, owner, type
DSUBDF	subsets created; stores scoping objects of subset definitions
DSUBUS	users authorized to access or update each subset
DXCPID	available code page maps; this two column table has an average of 50 rows

## Variable-Sized Tables for User Data

The sizes of the following data tables vary with usage:

Size	Depends on the number of
DCKOHIST	Checked out models and subsets; stores history of checkout information for determining why updates cannot be processed
DCKOID	Checked out models and subsets; stores checkout information down to object-level
DTXT	of descriptions added to model and their length

## Large Tables for User Data

You can expect the following tables to grow in size over time. DASC, the object association table and DOBJ, the object table, are the two largest tables in the encyclopedia.

Size	Depends on the number of
DASC	object associations across the encyclopedia
DNAME	objects with load name properties; about 25% of model objects
DOBJ	objects in the encyclopedia; stores object property, and model checkout information
DSUBEX	Subsets checked out, their types (system test subsets are larger than unit test and design subsets), and the number of objects they contain or have associations with (stores scoping objects and other selected objects in checked out subset)

## Relative Size of Indexes to Tables

The percentages provided in the following table are based on relative row lengths. For example, the row length of Index 1 for the DASC table, is 35 percent of the row length of the DASC table, or Object Association table.

Table Name	Index 1	Index 2	Index 3	Index 4
DASC	35%	25%	35%	N/A
DCKOHIST	47%	N/A	N/A	N/A
DCKOID	10%	20%	20%	N/A
DGRPUS	100%	100%	N/A	N/A
DMAX	3%	N/A	N/A	N/A
DMDL	3%	28%	N/A	N/A
DMDLUS	40%	40%	N/A	N/A
DNAME	2%	20%	N/A	N/A
DOBJ	2%	20%	6%	4%
DSUBDF	50%	50%	N/A	N/A
DSUBEX	42%	42%	42%	N/A
DSUBID	6%	56%	N/A	N/A
DSUBUS	39%	39%	51%	N/A
DTXT	1%	N/A	N/A	N/A
DUSER	14%	N/A	N/A	N/A

## Estimate CSE Size

A medium encyclopedia contains about 250,000 objects and a large encyclopedia contains about 500,000 objects.

The actual amount of space required for the CSE depends on the operating system and the DBMS. However, you can base an estimate on the size of the DOBJ table.

To calculate a rough estimate of the size to allow for the CSE, use a formula within the following range:

Low: CSE size (in Kbytes) = 1.75 \* DOBJ

High: CSE size (in Kbytes) = 1.9 \* DOBJ



The previous formula is based on the following assumptions:

Table Name	Size Relative to DOBJ	Description
DASC	.5	DASC is about half the size of DOBJ
DNAME	.2	Approximately 25 percent of DOBJ rows have a corresponding row in this table. The DNAME row is shorter than the DOBJ row.
DSUBEX	0 - .1	Varies depending on the extent subsetting is used. If there is no subsetting, do not include this factor in estimate.
All others	> .1	The total size of all remaining tables is less than 10 percent the size of DOBJ.

## Split Files Across Multiple Physical Drives

If physical disk resources on the encyclopedia server permit, splitting files across multiple physical drives will improve disk performance. DOBJ and DASC are the largest and most frequently accessed tables in the encyclopedia. Placing them on separate drives, as well as their indexes, can significantly improve performance. DSUBEX is heavily accessed during subset download; performance will also benefit by having that table and its indexes on different drives.

The following configuration will provide the best possible disk performance:

Drive	Contents
A	DBMS log files
B	DBMS rollback segments (if applicable)
C	All tables except DOBJ, DSUBEX, and DASC
D	All indexes except those for DOBJ, DSUBEX, and DASC
E	DOBJ table
F	DOBJ indexes
G	DSUBEX table

Drive	Contents
H	DSUBEX indexes
I	DASC table
J	DASC indexes

You need not distribute data manually if CSE database files reside on RAID disks that already use striping (for example, RAID 0, RAID 1, RAID 3, and RAID 5). It is suggested by most DBMS vendors that log files not be striped, as log records are inserted and retrieved sequentially.

## Capacity Planning Checklist

Use the following list as a guide for capacity planning:

1. Estimate the number and type of models and subsets that will be used. Remember that system test subsets will be larger than design or unit test subsets. Also, if you choose to create development models from your production models, they may be smaller than the original production models. This impacts:
  - DMDL
  - DSUBDF
  - DSUBID
  - DSUBEX
2. Estimate the number of users and groups that will be accessing the models and subsets. Identify the functions they will perform. This impacts:
  - DUSR
  - DSUBUS
  - DGRPUS
3. Estimate the average number of subsets that will be checked out at one time. This impacts:
  - DSUBEX
  - DCKOID
  - DCKOHIST
4. Estimate how much descriptive information will be stored. This impacts DTXT.

5. Review the technical requirements for the CSE and obtain the minimum disk space and memory required.
6. If you plan to spread the CSE tables across multiple disks, see [Split Files Across Multiple Physical Drives](#) (see page 25).



# Chapter 4: Tuning the Client Server Encyclopedia

---

This chapter deals with tuning the ClientServer Encyclopedia.

## Client Server Encyclopedia Processes Affecting Performance

Most requests for the ClientServer Encyclopedia (CSE) server that originate from a client are completed immediately. However, there are several requests that start resource intensive processes on the server. The time to complete these requests depends on server configuration, server activity, and size of the model/subset included in the request.

### Resource Intensive Processes

The resource intensive processes on the CSE include those initiated by the following commands:

- Adoption
- Checkout/Download/Extract
- Consistency Check
- Copy Model
- Create Model from Subset
- Delete Model
- Encyclopedia Validate Report
- Expansion Conflict Report
- Generate Code
- Generate DDL
- Migration
- Override Model Checkout
- Override Subset Checkout
- Update/Upload/Apply

## Enable Parallel Processing

Each of these commands has a corresponding utility process started on demand by the Message Dispatcher. Such requests are initiated immediately if that utility is not currently processing a previous request. Any subsequent requests are queued and are executed one-at-a-time until all are complete. These utilities work exclusively on each request until it is complete. To enable multiple copies of these utility processes to run simultaneously, modify the Message Dispatcher initialization file, usually `iefmd.ini`, to increase maximum occurrences.

## Backup and Recovery of the CSE Tables

The type of backup taken, and the recovery level of database tables are determined by the volatility of the data in the tables and the recovery level required by the application.

### Backup Options

In general, there are three backup scenarios for the CSE which are defined by their level of recoverability:

- Disk device backup, which includes all files
- Table backup without archived update logs
- Table backup with archived update logs

### Table Backup/Recovery with Disk Device Contents

Disk device backup, which is produced at the operating system level, is appropriate for disaster recovery operations since the backup consists of the contents of an entire disk device. Recovery of tables can be accomplished only in conjunction with restoration of all files on the disk device. Therefore, it is probably not a preferred method for preparing for normal, non-disaster recovery. During a disk device backup, encyclopedia databases cannot be updated, thus read consistency is assured.

### Table Backup/Recovery Without Update Logs

Encyclopedia tables can be backed up in such a way that read consistency is guaranteed, but application of logs is not possible or not intended. For example, Oracle export can accomplish this. Recovery from this type of backup can be made only to the point at which the encyclopedia table backups were taken. Such a backup strategy is ideal for sites where the databases are updated only during say an 8 a.m. to 8 p.m. workday and backups are scheduled for nights. In such a case, no update logs would be expected after a certain hour, so risk of possible data loss during recovery would be low.

## Table Backup/Recovery with Update Logs

Encyclopedia tables can be backed up with update logs to allow recovery to the point of failure. Here, a read consistent backup is taken in accordance with DBMS requirements and the update logs are saved, or archived. In the event of an encyclopedia failure, the tables can be recovered to the backup and the update logs applied as close to the failure point as possible. This backup method supports true 24 by 7 availability of the CSE.

## Guidelines for Scheduling Backups

Scheduling frequent backups is recommended when:

- Tables are heavily updated
- The number of update logs created per day is large
- Recovery must begin immediately when needed
- Recovery time must be short

To ensure the continual availability of the table data over time, maintain backups and update logs on disk. When lead-time to availability is acceptable, update logs and backups can be archived to free disk space. To minimize the risk of loss of data during a disaster, maintain disaster recovery copies of backups and update logs in an off-site location. Schedule the transfer to occur on a regular basis.

## Performing CSE Cleanup

The cleanup utility removes objects, properties, and associations in unusable or incomplete models. This is indicated by a value of U in the model status field of the DMDL table. This situation occurs when GEN NEW MODEL, COPY MODEL, CREATE MODEL FROM SUBSET, UPLOAD, or LOAD does not complete successfully and have taken intermediate checkpoints. Objects and associations are saved during these checkpoints, but are not valid until the respective CSE process successfully completes. The cleanup utility removes unusable objects, properties, and associations from the encyclopedia. The cleanup utility does intermediate checkpoints, so it can be stopped and restarted.

## Cleaning Up the CSE

### Follow these steps:

1. To determine whether unusable objects or associations exist, execute this query from the encyclopedia DBMS query manager:

```
select model_id, model_name, model_status from dmdl
```

If any returned rows have a model\_status of U, the model is unusable; proceed.

2. Ensure that none of the resource intensive CSE server processes are active.

**Note:** For more information, see [Resource Intensive Processes](#) (see page 29).

3. Run the cleanup command line utility with the following parameters. If you do not specify optional parameters, the values default to the indicated IEFMD environment variables.

```
cleanup -u userid
```

```
[-p dispatcher name][-v environment][-g encyclopedia group]
```

Where:

*userid* is the logon ID of the encyclopedia administrator

Optional Parameter	Description	Environment Variable (default)
-p <i>dispatcher name</i>	Name of the Message Dispatcher	IEF_MDNAME (iefmd)
-v <i>environment</i>		CSE_ENV
-g <i>group</i>	Encyclopedia group name	IEF_ENCYGROUP (ency)

## Maintaining CSE Tables and Indexes

Regular maintenance of the CSE tables and indexes, both after cleanup of unusable models and as a function of normal operations, is critical in ensuring that efficient accesses to CSE data are maintained. The frequency at which this maintenance is performed depends on factors such as the volume of use of the CSE and the volatility of the models in the CSE.

These factors also impact the growth rate and fragmentation of CSE tables and indexes. As objects are added to the CSE, table and index freespace will be reduced and space extents may be taken. This activity should be monitored and corrected, by reorganization and space reallocation, before performance is impaired.



## Cache and Threshold Settings

The following values are set in the CSE Support Client:

- Object cache
- Override checkout threshold

### Object Cache

The object cache is the number of unreferenced objects kept in memory during any process or active utility. This value applies to the entire CSE. In general, the smaller the object cache, the slower the processing speed, but the lower the memory requirements for the CSE process. With higher object caches, processing speed is increased and so is memory use. If the object cache is set too high, paging of memory can occur, degrading performance.

The paging of objects out of memory to disk is a function of a Least Recently Used (LRU) algorithm. The object cache is sometimes known as the LRU cache because the object cache value indicates the number of unreferenced objects to be held in memory before invoking the LRU algorithm to release objects. The default value for the object cache is 100,000.

**Note:** Only the Encyclopedia Administrator is authorized to change the object cache value.

### Modifying the Object Cache Value

**Follow these steps:**

1. Start the Support Client. Select Utilities then Object Cache.
2. Evaluate the displayed value. To increase processing speed, determine how much to increase this value; to lower the memory requirements, determine how much to decrease it.
3. To estimate the maximum value for your configuration, use the following formula:

$$\text{available memory} * 1500$$

Where:

available memory is the total system memory in megabytes minus the operating system requirements.

1500 represents the number of objects per megabyte of memory utilized during resource intensive processing on the CSE.

4. Type the preferred object cache value over the displayed value and select the Modify command.

## Override Checkout Threshold

The override checkout threshold is the number of objects that causes the override of a checkout of a model or subset to be completed in one transaction with no incremental commits. If the number of objects in the model or subset exceeds this value, incremental commits are made every 200 objects. This value applies to the entire CSE.

### Modifying the Override Checkout Threshold

**Follow these steps:**

1. Start the Support Client. Select Utilities then Object Thresholds.
2. Evaluate the value displayed in the Override Checkout entry field. The default value is 100,000. To get quicker results, determine how much to increase this value. To conserve machine space and to allow more concurrency, determine how much to decrease this value.
3. Type the preferred override checkout threshold value over the displayed value and select the Modify command.

# Chapter 5: Managing the Client Server Encyclopedia Workload

---

You can improve the performance of the CSE by implementing processes that help you use the encyclopedia more efficiently, reduce contention among requests, and decrease the delay associated with failures and poor planning.

## Ways to Improve Processes

The process improvements span a variety of areas from scheduling to communications, and aid in the general coordination of development projects.

In many cases, you can use general techniques that do not address specific areas of concern. These techniques are good practices and help increase overall performance by decreasing the time associated with non-successful attempts. The following topics fall under this category of general techniques:

- Understanding what causes contention
- Avoiding contention and model corruption
- Making the most of peak and non-peak processing
- Creating subject matter experts
- Using reports to facilitate planning
- Managing and coordinating projects

In other cases, you must identify and understand the specific encyclopedia functions that incur the most processing time, usage, and contention. For more information, see *Specific Encyclopedia Functions*.

## Causes of Contention

The first step in reducing contention is to understand what causes it. Contention can be caused by:

- DBMS timeouts and deadlocks
- Encyclopedia-enforced enqueue waits

## DBMS Timeouts and Deadlocks

DBMS timeouts or deadlocks occur when two users attempt to access a database record or table that is locked. Some of the earlier supported DBMS's lock records at page level. Data for different models were stored on the same DBMS page. Because of this, any two users, regardless of the models being accessed or updated, can experience timeouts and deadlocks. Even though it is not easily predictable, certain encyclopedia functions have a tendency to cause timeouts and deadlocks. Model deletion is one of these functions.

These functions, especially when run on large models, perform thousands of DBMS inserts, updates, and deletes to key encyclopedia tables such as DOBJ and DASC. Functions like copy model and upload of a new model may perform thousands of inserts and updates but tend to access new pages rather than existing ones. Model delete accesses existing pages that can be spread across the entire encyclopedia. This amount of activity increases the chance of another user attempting to access the same page of a table or index.

Another group of encyclopedia functions have a tendency to cause timeouts and deadlocks because they are not capable of taking interim commits. Because these functions must force a complete rollback if they fail, they cannot take an interim commit, which causes the DBMS page locks to escalate to tablespace locks when run on a large amount of data. These functions are:

- Migration
- Adoption
- Apply of an existing model (Apply of a new model does incremental commits.)

## Encyclopedia Enqueue Waits

The second kind of contention is caused by encyclopedia constraints known as enqueue waits. Enqueue waits are constraints built into the encyclopedia to prohibit two functions that are known to be incompatible from running at the same time. Because enqueue waits can be predicted, it is worthwhile to learn the rules that govern them.

## Encyclopedia Concurrency Matrix

The encyclopedia concurrency matrix is the table of rules that determine which functions are compatible to run at the same time. An enqueue wait occurs when a second process is started that violates the rules of the concurrency matrix. The second process is put into wait mode and retried every minute to see if the enqueue still exist. After a specified number of retries per function, the second process terminates (it never really got started) with a resource conflict message. An enqueue wait occurs at the encyclopedia, model, and subset levels. However, encyclopedia enqueue are used only for updating generation options not stored in the model.

Some of the more common restrictions include:

- A download and an upload cannot run in the same model at the same time.
- No uploads, downloads or code generations are allowed in a model while a migration into that model is running.

Encyclopedia users should be aware of the concurrency matrix and know how to use it.

**Note:** For more information, see the appendix “Concurrency Matrix” in the *CA Gen Client Server Encyclopedia User Guide*. The matrix shows only the concurrency enforced by the CSE. Other types of contention, such as that caused by the DBMS, cannot be determined from the matrix.

## Avoiding Contention and Model Corruption

You can start to avoid contention the following ways:

- Know what is currently running
- Know what is scheduled to run
- Communicate what is going to be run

### What Is Currently Running?

Before you perform an encyclopedia function that may cause contention, see which functions are currently running. You can determine what is running with operating system utilities or by viewing the log files created by the servers and utilities.

Each server and utility registers with the Message Dispatcher using the Message Dispatcher initialization file (usually named `iefmd.ini`). One of the parameters in the initialization file indicates the log location for the server or utility. The path and file name is defined by the value in the log location parameter concatenated with instance numbers. For example, the first instance of the upload utility would normally be named `up1.01`.

After determining what functions are currently running, view the appropriate log files to determine what user, model, and subset is being processed by the function of interest.

### Review Lock and ID Allocation Server Configuration

Review the configuration of the lock and ID allocation servers in the Message Dispatcher initialization file (usually `iefmd.ini`) using any text editor. Only one lock and ID allocation server can exist on a database. However, you can have multiple servers running different software versions against the same database.

In this case, in the Message Dispatcher initialization file you must define only one of these servers (usually the one with the latest software) to have its lock/ID servers running. The other server(s) will use the first server's lock/ID servers to lock models and allocate IDs.

If you do not configure the servers in this way, multiple processes can perform write locks against the same model, which can result in a corrupted model. In addition, two ID servers can fetch IDs. This can also result in contention problems.

If no encyclopedia functions are defined for a database, you can have a Message Dispatcher initialization file with no lock/ID servers defined. However, if you have an encyclopedia defined, you must specify a lock/ID server.

## Examine Outstanding Locks

You can use the Support Client to see what model and subset locks are currently outstanding. There are three types of model locks: Read, Write and Upload. If model locks are outstanding, then certain locks can occur on the same model but others are not allowed.

The following table shows concurrent model locks:

Model Lock	Allowed	Not Allowed
Read	Read	Write, Upload
Write		Read, Write, Upload
Upload	Upload (if for different subset)	Read, Write

**Note:** Upload locks allow simultaneous uploads of two subsets. When a subset lock is outstanding, no other locks are allowed on the same subset definition.

## Functions Performing Read Locks

The following functions perform Read locks on the model indicated. When a model is Read locked by one function, other functions also can lock that model for Read but no function can lock it for Write or Upload. The following table shows Functions that can hold concurrent read locks on a model:

Function	Which Model
Add Subset	selected model
Adoption	source model
Compare	source and destination models

Function	Which Model
Consistency Check	selected model
Copy Model	source model
Display Generation Components	selected model
Display Packaging	selected model
Encyclopedia Validate	selected model
Expansion Conflict Report	selected model
Generate Delta	selected model
Migration	source model
Walkency	selected model

### Functions Performing Write Locks

The following functions perform Write locks on the model indicated. When a model is Write locked, no function can lock it for Read, Write, or Upload. Functions that can hold write locks on a model follow:

Function	Which Model
Add/Update/Delete Packaging	Selected model
Add/Update/Delete Paths	Selected model
Adoption	destination model
Change Model Administrator	selected model
Change Model Checkout Userid	selected model
Cleanup	all unusable models
Convert Model	selected model
Delete Model	selected model
Download	selected model
Generate Code	selected model
Generate DDL	selected model
Migration	destination model
Override Model	selected model
Rename Model	selected model
Verify Delta	model specified in translation file

## Functions Performing Upload Locks

The following function performs Upload locks on the model indicated. When a model is Upload locked, another function can Upload lock it for a different subset but no function can lock it for Read or Write. This table shows the function that can hold upload locks on a model:

Function	Which Model
Upload	Selected model

## Functions Performing Locks on Subsets

The following functions perform locks on the subset indicated. When a subset is locked, no other functions can simultaneously lock it. Functions that can hold write locks on a subset:

- Add Subset
- Change Subset Administrator
- Change Subset Checkout Userid
- Create Model from Subset
- Copy Subset
- Delete Subset
- Download Subset
- Modify Subset
- Override Subset
- Rename Subset
- Upload Subset

## What Is Scheduled to Run?

By scheduling activities, organizations can give critical tasks higher priority and reduce resource contention. Jobs that affect the entire encyclopedia, such as image copy and reorganization, should be scheduled to run at a consistent time that is communicated to all encyclopedia users. Users should understand that the encyclopedia is unavailable during this time and that all encyclopedia jobs must finish before the scheduled start time.



Schedule encyclopedia jobs that you believe may cause contention with other encyclopedia activity, such as large model copies, deletes, and migrates, during a period when other ongoing activity is curtailed. These time periods are best scheduled during non-peak hours and must be communicated and agreed upon by all encyclopedia users. All users should have access to the schedule.

## Communicate What Is Going to Run

Activity that has a likelihood of causing contention within a model should be communicated to everyone working in that model. Advanced warning allows others to prepare and schedule their activities to avoid contention. It also allows users to respond if the activity being scheduled causes hardship on other priority activities.

For example, if 10 users are actively working in a model and one user needs to run a download, no other user can perform an upload during the download. The user should send a group message to the other users in that model notifying them of the time of the download. The other users can then plan their activities around the download and avoid the cost and frustration of a failed upload.

## Database Object ID Partitioning

The CSE installation and configuration no longer provides the user the option to create multiple Object ID partitions on the Encyclopedia Database. The option was originally added for page level locking, which was present in older versions of supported databases. For example, SQL Server. Since SQL Server now provides row level locking, multiple Object ID partitions are no longer necessary for any CSE database platform. The multiple Object ID partition option is not available at installation or configuration time however the feature is supported for the existing Encyclopedia database.

If your CSE does not contain many models you may still experience contention, but the contention should improve as more models are loaded to the CSE. You may also choose to define physical database partitions to help avoid contention (if your database supports this feature).

If you experience contention during the initial uploads, try single threading uploads until the database is sufficiently populated and then you can increase utilities.

If you are upgrading, your existing data remains the first partition, even if you increase the number of partitions.

**Note:** For additional information on Object ID partitions, see the Support Client online help.

## Peak and Non-Peak Processing Times

Some encyclopedia functions can run during peak processing hours and other functions should run during non-peak hours. You get better at deciding the appropriate times by understanding the concurrency matrix, reviewing performance reports, and continuing to work with the encyclopedia (which increases your experience).

### Peak Processing

Encyclopedia functions that can run together within the same model are easily run during peak processing hours. A few illustrations of these types of processes are code generation, reporting, and defining subsets.

Other encyclopedia functions that have a tendency to cause contention should be broken into small jobs (where applicable). Illustrations of these are migration and adoption. Both can cause contention when they do not complete in a short time and both can be broken into small pieces. Running 10 migrates that each complete in 10 minutes causes much less contention than one migrate that runs for 100 minutes.

### Non-Peak Processing

Use software available on the operating system you are using as your server to submit requests to initiate standalone executables at a specific date and time during off-peak hours. Execute the function without any parameters to display the parameter options.

Any utility that services requests for long running jobs can be controlled by permitting only one instance of that utility to be active at a time. The Message Dispatcher .ini file includes the maxinstance parameter for each utility. Setting the maxinstance parameter to 1 is the way to ensure that requests to the corresponding utility will be single-threaded. For example, to ensure that only one upload request can be processed at a time, set the maxinstance parameter to 1 for the Upload utility, UTLUP.

The following table shows standalone utilities:

Utility Name	Description
ACBLKUSE	Action block use report utility
ADPTMODL	Adopt model utility
CLEANUP	Clean up (delete) encyclopedia utility
COMPMODL	Compare model report utility
CONSISCK	Consistency check report utility
CONVALL	Convert all models utility

Utility Name	Description
CONVMODL	Convert model utility
COPYMODL	Copy model utility
DELMODEL	Delete model utility
DIAG	Diag utility
DOWNLOAD	Download utility
ENCYVAL	Encyclopedia validate utility
EXPCNRPT	Expansion conflict report utility
GENDELTA	Generate delta utility
GENVER	Generate verify utility
MIGRMODL	Migrate model utility
OVERRIDE	Override utility
TRNRPT	Create transaction report utility
UPLOAD	Upload utility
VERDELTA	Verify delta utility
VERUP	Verify upload utility

The following table shows server executables:

Utility Name	Description
SRVCONS	Construction server
SRVCOORD	Coordination server
SRVID	Id allocation server
SRVLOCK	Lock server
SRVMS	Model/Subset server
SRVUGA	User/Group/Authorization server
SRVVC	Version Control server

The following table shows utility executables:

Utility Name	Description
UTLACBLKU	Action block use report utility

Utility Name	Description
UTLADPT	Adoption utility
UTLCONCK	Consistency check report utility
UTLCPYM	Copy model utility
UTLCRMFS	Create model from subset utility
UTLDELM	Delete model utility
UTLDIAG	Diagnostic utility
UTLDOWN	Download utility
UTLEVAL	Encyclopedia validate utility
UTLEXPRP	Expansion conflict report utility
UTLGENCD	Generate code utility
UTLGEND	Generate DDL utility
UTLMIGR	Migrate utility
UTLOVER	Override checkout status utility
UTLUP	Upload utility
UTLVERD	Verify delta utility
UTLVERUP	Verify upload utility

## Subject Matter Experts

Training is a very effective way to improve performance. The more information a developer has about his or her requirements and how all parts of the system interact, the greater his or her success. Although it is not always feasible to train all team members in every area, try to have a few individual team members trained in specialized areas. These individuals can share their knowledge with other team members and improve the performance of the entire team.

Two areas in which training can be very effective are subsetting and migration.

### Subsetting Expert

An improperly defined subset results in too many or too few objects to complete a task. Users who do not have all objects needed to complete a task have a tendency to over define a subset and get more than is needed. Over defining a subset has a ripple effect of causing other users to get downgraded. The net effect is that subsets must be overridden, re-defined and downloaded. This incurs unproductive time.

You can usually improve subsetting practices by training one project member for the role of subsetting expert. This person should review each subset for the correct scoping objects, expansion, and protection needed for the task. Team members should be taught to use the subsetting documentation to scope their individual subsets. These subsets should then be reviewed with the subsetting expert. The subsetting expert should explain any requested changes so that each developer can gain a better understanding of the subsetting rules.

As team members become better at subsetting, the number of subsets that are incomplete or too large decreases. Well-scoped subsets reduce processing time and improve performance. The number of unnecessary objects included in each subset decreases, as does the number of multiple subsets being downloaded for a single task.

## Migration Expert

A migration may fail for several reasons. Failed migrations take more than twice as long as successful migrations because they result in a rollback. Trial migrations are equivalent to a failed migration because they also result in a rollback. You want to reduce the failure rate and number of trial migrations because migrations may cause contention on the entire encyclopedia.

You can do the following to eliminate or reduce the number of unsuccessful migrations:

- Understand the rules of migration. For each object, know its enabling and companion object.
- Keep selected objects for migration small to avoid repeat migration attempts of many objects if one object fails.
- Use Public Interface (PI) and available reports to understand the model and objects to be migrated.

Each project should have a migration expert who is thoroughly trained in the rules and best practices of migration. This prevents a large number of failed migrations and reduces or eliminates the need for trial migrations. The migration expert should implement a request process to aid in the scheduling of migrations. The process should prevent the need for large migrations by allowing for incremental movement of changed objects.

## Reports

Use of the following can help you manage a heavy CSE workload:

- Compare Aggregate Objects Report
- Public Interface views

- Expansion Conflict Report
- Action Diagram listings

## Determine Objects Needing Migrating

The Compare Report can provide a complete list of objects to evaluate for possible migration. You can then ensure that all changes are tracked and included in the migration plans.

**Note:** For more information, see the chapter “Compare Reports” in *CA Gen Client Server Encyclopedia Version Control User Guide*. You may also use PI to determine what aggregate objects have changed during a given time period.

A common cause of failed migrations is attempting to migrate a function or process to a model, which does not contain the parent objects in the function hierarchy. You can use the PI to identify all parent functions and processes. Those, which are not contained in the destination model, may be included in the first migration attempt to prevent the need for a second attempt or for a trial migrate.

**Note:** For more information, see *CA Gen Client Server Encyclopedia Public Interface Reference Guide*.

## Determine Potential Checkout Downgrades

Use the Expansion Conflict report to display shared objects that will be downgraded when the subset is checked out. This report does not update the encyclopedia and is less time-consuming than download.

## Examine Listings

Developers often use action diagram listings to understand and diagnose existing code. The action diagram listing is at the top of the generated source code. Instead of running a report that reads the encyclopedia tables, users can browse the top of the source listing and eliminate the need for any encyclopedia access.

## Project Planning and Coordination

Good project planning is perhaps the most essential element to improving encyclopedia processes. Understanding and implementing good project planning practices is crucial to reduce overlaps when subsetting, migrating, creating project models, and so forth.

## Naming Standards

Naming standards can help identify objects that are obsolete or temporary and reduce object redundancy. Without good naming standards, encyclopedia users create redundant objects or use obsolete ones. When this happens, the users often must re-subset to delete the object usage and include the correct object.

### Standards for Shared Objects

Standard naming conventions can also be used to communicate the purpose or status of a shared object. By having a naming convention, an object list search is quicker and more likely to succeed. A common reason for object searches is to locate an action diagram or exit states to include in new code. In addition to a naming convention for shared objects, a good description further indicates whether the object is suitable for the developer. Good in-line comments communicate how the action block should be implemented.

### Tagging Obsolete Objects

Another use for naming conventions is to tag obsolete objects. If an obsolete object cannot be deleted (because it is referenced in some way), the object can be renamed so that its obsolete status is communicated. One possibility is to rename it so that it begins with ZZ\_OBS\_ and then appends as much of its original name as possible.

This naming scheme groups all obsolete objects at the end of an object list. You can easily find them and the name clearly indicates to the users to remove them from their code and subsets. This in turn makes it easier to eventually delete the objects and reduce the size of the model.

### Tagging Temporary Objects

A naming convention for temporary objects is also helpful in locating and removing unnecessary objects. One possibility is to start the name with ZZ\_TEMP\_ followed by the developer's initials and then whatever descriptive information the developer chooses. This naming scheme also groups temporary objects at the bottom of the list. It makes them easy to find and identifies the developer. The objects can be deleted if the developer forgets.

### Tagging Subsets

The last illustration of naming conventions is to have the developers name their subsets starting with their initials. It reminds the developers of all subsets they have so that they can remove any that are obsolete. It also identifies to all other developers the owner of the subset.

## Project Models

Dividing the development work among project models can greatly aid in the reduction of processing time and conflicts. Developers working in a smaller project model have reduced processing time during downloads since the subsets pull in fewer neighborhood objects. Potential subsetting downgrades and resource conflicts are also reduced.

When feasible, contention can be reduced and developer productivity increased by separating work into one or more project models. Separate models increases the development coordination necessary for a project, but reduces the contention incurred by spreading the work across models.

## Coordinating Shared Objects

Careful management and coordination of shared objects is also essential to the success of a development team. By carefully monitoring the modification of shared objects between models, you can reduce the number of migrations necessary to get the most current copy containing all changes into each model (for development or testing). This coordination also prevents the need for multiple coding efforts that become necessary when a shared object is changed in different models and a migration cannot preserve both sets of changes.

## Avoiding Mass Upload

Calling for all subsets to be checked in to facilitate migration can be very costly and nonproductive. Migration can take place while subsets are checked out.

Consider checking out a subset with the root subject area for Modify and the databases for Delete before any other subset is checked out. You can then migrate the data-related objects (entity types, attributes, relationships, and so forth) by overriding the subset, and then re-checking out the subset.

You can use this same technique for other shared objects. If shared objects are being changed in one model, you should check them out for Modify or Delete in all other models but not download them. This protects them from being changed or ending up in another subset for modify or delete. When it is time for them to be migrated, you can override the protective subsets and the migration can proceed without conflict.

When a mass check in is required, spread the uploads over time so everyone does not try to upload at the same time (such as at the end of the business day).

Following a mass upload, a mass download is usually needed for the next day.



## Regeneration and Installation

Another way to improve the performance of the encyclopedia is by developing procedures for the regeneration and installation of code.

An organization should develop procedures for informing the group whenever they modify an object that has a large impact on regeneration. This depends on how objects are shared in an organization. In any case, set up procedures that limit when these changes are made and that indicate to the entire organization when the changes are done.

## Specific Encyclopedia Functions

Improving processes that involve specific encyclopedia functions begins with understanding how these encyclopedia functions work; this is key to making sound decisions on coordinating their usage. The following encyclopedia functions are discussed individually along with tips for improving their performance:

- Download
- Upload
- Override checkout status
- Migrate
- Adopt
- Add a new model
- Model delete

## Download

A download consists of three basic phases: expansion, extraction, and output. During the expansion phase, the scoping objects that are selected during subset definition are expanded to include related (neighborhood) objects. The extraction phase ensures that the subset is complete (that all necessary objects are included in the subset) and downgrades the user's protection to objects if necessary. Downgrades occur because an object is already checked out to another user or because the object was requested with delete and associated objects on the same model were not included in the subset expansion. In the output phase, the model is written to the transaction file.

Although download is primarily a read-only process, the encyclopedia does not allow other functions to occur while a download is in progress.

As objects are extracted from the encyclopedia, the encyclopedia updates the DOBJ or the DSUBEX table to indicate that the objects are checked out with a certain level of protection. Updates to the model for this purpose are the source of locks. Periodic commits are executed to alleviate the scope and duration of the locks. If a download does not complete successfully, the subset is not marked as being checked out. However, because commits were taken along the way, the next download must perform a cleanup routine before executing the download to remove all checkout updates for the failed download.

## Techniques

The following techniques can improve the process involving downloads:

- Plan downloads in advance and execute them during non-peak hours.
- To avoid downloading during prime time, have more than one subset checked out at all times. If work on one subset is finished or needs to be redefined, there is always a second subset to work on.
- If downloads are allowed during the day, require that all users of that model be notified.
- Create a subsetting expert for each project or model. This expert can help users define subsets appropriately, which reduces the number of incomplete subsets and subsets that are too large.
- Users that have read-only access to a model can do a read-only download that does not check out anything. This is different from having update access and scoping objects for read protection. Users can also explicitly request a read-only download.

## Upload

An upload updates the model in the CSE. If the upload is without check in, the upload usually runs faster because the object status is not being changed. For an upload with check in, the following occurs:

- All of the transactions in the transaction file are processed.
- A commit occurs.
- The DCKOID row is deleted.
- All objects in the subset are checked in or deleted from DSUBEX with periodic commits taken.

If the upload abends during the process of checking in the objects, the next download of the model or any subset of that model cleans up the partial checkout.

Since upload is an update process, the encyclopedia only allows limited access to the model while the model is being updated. An upload cannot run while a download or code generation is occurring in that model. The only activities that can run simultaneously with an upload in the same model are other uploads or overriding, renaming, deleting or copying another subset. Locks occur as objects are inserted and deleted, properties are modified, and associations are added and deleted.

If the upload fails, you cannot continue to work on the subset until the upload is completed. This may be important when you plan for an upload.

## Techniques

The following techniques can improve the process involving uploads:

- Upload without check in frequently. This serves as a primary method of back up in case you need to recover work on the workstation.
- Understand how to perform an upload manually. This can be valuable if you have to restart a failed upload.
- Keep uploads small during peak processing hours. Frequent uploads without check in accomplishes this. Most uploads that take less than 10 minutes will not fail because of an enqueue wait.
- Perform large uploads (more than one new procedure) during non-peak processing hours.
- Perform uploads with check in during non-peak processing hours, when possible. Uploads with check in take substantially longer than uploads without check in.
- Teach users to select Cancel or Ignore when *not* making a change with the toolset. Selecting OK, even if there were no changes, can trigger an update.

## Override Checkout Status

A table of checked out objects is kept in the CSE. For checkout of a model, objects are flagged as checked out in the DOBJ table. When the override function is requested, the encyclopedia updates the DOBJ table or deletes the rows in the DSUBEX table for the model or subset.

Most functions are allowed to run at the same time as an override of a checkout status, but because of the amount of updating that is done, this can cause contention.

Periodic commits are taken during the delete or update. If a failure occurs during override, the next download of the model or a subset of it cleans up the partial checkout.

## Techniques

To improve the process involving override checkout status for large subsets, consider uploading with check in during non-peak processing hours even if there have been no updates.

## Migration

Migration copies objects from a source model to a destination model. For the source model, migration is a read process. For the destination model, migration is an update process. Periodic commits are not performed because of the difficulty of detecting and correcting the results of a partial migration. Because there are no periodic commits, migration can quickly cause contention across the entire encyclopedia. Trial migration executes the same logic as migration except a rollback is done at the end of processing, which approximately doubles the processing time.

## Techniques

The following techniques can improve the process involving migration:

- Migrate at the lowest aggregate level. For example, if you change an attribute, migrate the attribute, not the entity type.
- Keep migrations small (less than 10 minutes) if you must do them during peak processing hours.
- Know what is running before migration. Do not perform the migration if other large encyclopedia jobs are running; such as an upload, download, model copy or another migrate.
- Whenever possible, schedule migrations to run during a time period for single stream processing.
- If you must perform migrations during peak processing hours, notify all encyclopedia users in advance.
- Analyze the objects to be migrated. Know the rules of migration for each type of object. Identify the enabling and companion objects for each object to be migrated. You can use the Compare Report to compare the results between the source and destination models.
- Avoid large migrations. If one object fails, the entire migration fails.

## Adoption

Adoption is used to assign common ancestry to an object for the purpose of migration. The original encyclopedia ID and the original object ID of the selected destination model object are updated to match the original encyclopedia ID and the original object ID of the equivalent object in the source model. Adoption establishes a correspondence between objects that are alike so they can be migrated or compared.

The adopt function automatically unadopts component objects. Component objects that do not get adopted are given a new original encyclopedia ID and a new original object ID to sever any common ancestry they may have. Incremental commits are not taken during the adopt function. Adoption of large models can cause contention.

To avoid contention, you can selectively adopt large models rather than using the All option.

The following table lists the suggested order of adoption for specific object types and the prerequisites that are required for the adoption of the objects:

Object Type	Prerequisite
Relationships	Entity types
Data records	Entity types, Relationship
Tablespace	Databases
Commands	Business system
Templates	Business system
Functions	Entity types and work attribute sets defined in views
Processes	Entity types and work attribute sets defined in views
BAA common action blocks	Entity types and work attribute sets defined in views
BSD common action blocks	Entity types and work attribute sets defined in views
Procedure	Business system, entity types and work attribute sets defined in views
Dialog flow	Procedures, Exit states
Online load module	Business system
Window load module	Business system
User defined matrix	User defined object class

All other selectable objects do not have prerequisites and can be adopted in any order.

## Techniques

The following techniques can improve the process involving adoption:

- Identify the purpose and scope of the adoption. Avoid an adoption on all objects if you require only a limited adoption.
- Refrain from deleting and recreating objects (thereby losing common ancestry) which forces an adopt before a migrate.
- Put procedures in place for tracking name changes that are not picked up by an adopt.
- Avoid large adoptions during peak processing hours. Keep adoptions small (less than 10 minutes) if you must perform them during peak time.
- Know what is running before an adoption. Do not adopt if other large encyclopedia jobs are running.
- Notify all encyclopedia users in advance if you must perform an adoption during peak processing hours.
- Whenever possible, schedule large adoptions to run during a time period for single stream processing.
- Use trial adoption only when necessary to identify and exclude objects that should not be adopted, such as those with the same name, or to ensure that no unintended name changes occur.

## Adding a New Model

The following activities can be considered as adding a new model to the encyclopedia:

- Model copy
- Generate new model
- Initial model upload
- Cross copy from another encyclopedia
- Create model from subset

Each of the activities creates objects, associations, properties, and text. The activities place a read lock on the source model to prevent changes to the model while the activity occurs. Periodic commits are executed to alleviate the scope and duration of the database locks.

When you add a new model, the model is initially put into an unusable state. The row in DMDL is inserted first, then all objects, properties, associations and text are added. Finally, the DMDL row is put into a modifiable state. Unless the add completes normally and the final step of updating DMDL occurs, orphan objects are left in the database. The orphan objects do not affect other operations, but the incomplete model is unusable. You can clean up the orphan objects with the cleanup utility.

For large models, the overall effect of a failed activity, such as a failed copy, can be substantial. Because a copy does periodic commits, orphan objects are left in the encyclopedia following a failure. For a large model in an encyclopedia where disk space is in short supply, you may need to clean up the orphan objects and REORG the database before being able to perform the copy again.

## Techniques

The following techniques can improve the process involving the addition of a new model:

- Check current disk space usage and estimate required disk space for a model before adding the new model. Increase disk space if necessary before performing the activity.
- Schedule model copies and generate new models for overnight processing.

## Model Delete

Model delete deletes objects, associations, properties, text, and subset definitions. It does periodic commits to alleviate the duration of the database locks. Because of the heavy amount of database activity, it has a tendency to cause contention across the entire encyclopedia. The first task done by a model delete is to flag the model as unusable from the DMDL table. Periodic commits occur during processing. If a model delete does not complete successfully, you can clean up orphan objects with the cleanup utility.

## Techniques

The following techniques can improve the process involving model delete:

- Schedule model deletes to run during non-peak processing hours.
- Schedule model deletes before a REORG.
- Rename models intended for delete and schedule the delete during a time period for single stream processing.





# Chapter 6: Monitoring and Tuning for Specific Platforms

---

This chapter discusses many installation and configuration options that affect application performance, which are most important to the Client Server Encyclopedia (CSE) architecture.

## Oracle

Consider the following recommendations for ways to tune the CSE for better performance.

### Routine Database Cleanup

Run the Cleanup utility often to delete obsolete objects and associations. For information to determine whether cleanup is necessary, see [Performing CSE Cleanup](#) (see page 31).

To determine whether cleanup is necessary, set ORACLE\_SID to the appropriate value for the encyclopedia database name and use sqlplus to execute the query. Typical value for ORACLE\_SID is DBCSE (for older Windows encyclopedias, older UNIX encyclopedias or for newer encyclopedias on either platform).

**More information:**

[Performing CSE Cleanup](#) (see page 31)

## NEXT and PCTINCREASE Reduction

To reduce PCTINCREASE and NEXT, alter the tables and indexes by issuing the following from within sqlplus. For very heavy encyclopedia usage, consider replacing 4M for DOBJ and DASC with higher values. The value, 4M, should be adequate if you export and import weekly. MAXEXTENTS is 99 by default, where the highest valid value is 121. Query to Reduce PCTINCREASE and NEXT:

```
alter table dasc storage (next 4M pctincrease 0) ;
alter index dasci1 storage (next 4M pctincrease 0) ;
alter index dasci2 storage (next 4M pctincrease 0) ;
alter index dasci3 storage (next 4M pctincrease 0) ;
alter table dobj storage (next 4M pctincrease 0) ;
alter index dobj1 storage (next 4M pctincrease 0) ;
alter index dobj2 storage (next 4M pctincrease 0) ;
alter index dobj3 storage (next 4M pctincrease 0) ;
alter index dobj4 storage (next 4M pctincrease 0) ;
alter table dname storage (next 4M pctincrease 0) ;
alter index dnamei1 storage (next 4M pctincrease 0) ;
alter index dnamei2 storage (next 4M pctincrease 0) ;
alter table dtxt storage (next 4M pctincrease 0) ;
alter index dtxti1 storage (next 4M pctincrease 0) ;
alter table dsubex storage (next 4M pctincrease 0) ;
alter index dsubexi1 storage (next 4M pctincrease 0) ;
alter index dsubexi2 storage (next 4M pctincrease 0) ;
alter index dsubexi3 storage (next 4M pctincrease 0) ;
```

← Increase 4M values for very heavy encyclopedia usage.

## Database Defragmentation

Performing an export followed by an import effectively defragments the encyclopedia database.

### Steps for an Export/Import-Oracle Windows

**Follow these steps:**

1. Run the export facility:

```
SET ORACLE_SID DBCSE
exp <userid> <password>
```

2. Export the user tables with the defaults. An SID of DBCSE is a default substitute for the ENCY database name. Dropping the user tables from the CSE will smooth the import process. To drop the user tables, change to the directory where CSE Oracle SQL is installed (%IEFCSEGEN%\..\cse\_oracle) and create the file composed of only DROP statements.

```
find "drop table" dbiefd.sql > drop.dbcse
```

3. Using sqlplus, run DROP.DBCSE.
4. To drop the PIVIEWS defined in the database, change to the directory where the CSE OracleSQL is installed (%IEFCSGEN%\..\cse\_oracle) and create the drop.view file:

```
find "CREATE OR REPLACE VIEW" piviews.sql > drop.views
```

Using a text editor, edit the drop.views file as follows:

1. Replace CREATE OR REPLACE VIEW with DROP VIEW.
  2. Append a semicolon (;) to the end of every line that starts with DROP.
  3. Save the file and exit from the editor.
5. Using sqlplus, run DROP.VIEW.
  6. If your database is fairly large, add a 100 MB file to the temporary tablespace:

```
SQLPLUS /NOLOG

CONNECT SYS AS SYSDBA
ALTER TABLE IEFENCY_TEMP ADD DATAFILE
'\DBCSE\DBCSETEMP.DAT'
SIZE 100M ;
```

7. Run the import facility:
- ```
imp <userid> <password>
```

## Steps for an Export/Import-Oracle UNIX

### Follow these steps:

1. Run the export facility:
- ```
setenv ORACLE_SID DBCSE
exp <userid> <password>
```
2. Export the user tables with the defaults. An SID of DBCSE is a default substitute for the ENCY database name. Dropping the user tables from the CSE will smooth the import process. To drop the user tables, change to the directory where CSE Oracle SQL is installed (\$IEFCSGEN/../../cse\_oracle) and create the file composed of only DROP statements.
- ```
grep "DROP TABLE" dbcse.ora > drop.dbcse
```
3. Using sqlplus, run drop.dbcse.

4. If you installed PIVIEWS, drop these. To drop the PIVIEWS defined in the database, change to the directory where CSE OracleSQL installed (\$IEFCSEGEN/./cse\_oracle) and create the drop.view file:

```
find "CREATE OR REPLACE VIEW" piviews.sql > drop.views
```

Using a text editor, edit the drop.views file as follows:

- a. Replace CREATE OR REPLACE VIEW with DROP VIEW.
- b. Append a semicolon (;) to the end of every line that starts with DROP.
- c. Save the file and exit from the editor.

5. Using sqlplus, run drop.view.

6. If your database is fairly large, add a 100 MB file to the temporary tablespace:

```
sqlplus /nolog
connect sys as sysdba
alter table iefency_temp add datafile
'/usr/iefcse/dbs/dbcsetemp.dat'
size 100M ;
```

7. Run the import facility:

```
imp <userid> <password>
```

## CSE Database Partitioned Across Multiple Drives

When you install the CSE, consider dividing the CSE database (DBCSE for both Windows and UNIX) across multiple physical drives. Place the low-volume system tablespace on any drive. Recommended configuration for four drive systems:

| Drive | Contents          |
|-------|-------------------|
| 1     | Log files         |
| 2     | Rollback segments |
| 3     | Table tablespace  |
| 4     | Index tablespace  |

## DDL Customization

If you have plenty of disks you can customize the DDL. The tables of greatest impact are DOBJ and DASC, the largest and most accessed, and DBSUBEX, which is heavily accessed during subset download. For a description of the results of customizing, see the table, Recommended Configuration for Optimal Disk Performance.

To create a system with this configuration, add the required tablespaces in the \$IEFCSGEN/./cse\_oracle/dbiefd.sql file. Then use sqlplus to create those tablespaces and run the cse\_config using the initialize option instead of the create option.

## SQL Server

The following section deals with optional tuning for SQL Server.

### Procedure Cache

The procedure cache variable defines the percentage of memory allocated to the procedure cache after the server's memory needs are satisfied. The SQL Server Installation Default for Procedure Cache (30) is valid with optional tuning as needed.



# Chapter 7: Database Administration for Specific Platforms

---

Database administration includes procedures run on a routine basis and less frequently used procedures that help you to customize your database as your resource needs grow.

## SQL Server/Windows Routine Procedures

Several procedures must be performed frequently to keep the CSE databases running efficiently. Unless the system administrator ID/password has been changed, use sa as the login ID and a blank password to start the SQL Server Management Studio:

- Cleaning up your database
- Updating database statistics
- Backing up your database
- Backing up transaction logs
- Adjusting database size
- Truncating transaction logs
- Stopping SQL Server
- Starting SQL Server
- Common SQL Server system procedures

## Cleaning Up Your Database

To avoid wasting space in the encyclopedia, remove unusable models from your database by running a routine cleanup procedure. It may be a good idea to schedule a nightly (or off-hours) cleanup.

Unusable models may be left in the encyclopedia from unsuccessful procedures, including:

- Unsuccessful copy model operations
- Unsuccessful create model from subset operations
- Unsuccessful upload new model operations

For example, if you run out of space while creating a model, data associated with the model remains in the tables until you perform a cleanup.

## Updating Database Statistics

SQL Server provides the ability to update statistics in a database. Update statistics will cause SQL Server to evaluate the current state of the database and store statistics about how many rows are stored in each table. This helps SQL Server determine which indexes to use when retrieving data from the tables.

Updating statistics frequently (you may choose to update statistics at the same time as your database backups), can help improve performance in the CSE. You should not take down the database or CSE software to run statistics for SQL Server. You may want to update statistics daily.

### **Follow these steps:**

1. Start up the SQL Server Management Studio from the SQL Server program group. Unless the system administrator ID/password has been changed, use sa as the login ID and a password to start the Management Studio.
2. From the DB drop down list box, select <DBCSE> (or your encyclopedia database name).



3. Select File Open, to open the file `estats.sql` in `%IEFCSGEN%\..\cse_msqsls`.

If you do not have `estats.sql`, then type in the following commands manually:

```
update statistics DASC
update statistics DCKOHIST
update statistics DCKOID
update statistics DENCY
update statistics DGRPUS
update statistics DMAX
update statistics DMDL
update statistics DMDLUS
update statistics DNAME
update statistics DOBJ
update statistics DSUBDF
update statistics DSUBEX
update statistics DSUBID
update statistics DSUBUS
update statistics DTX
update statistics DUSR
update statistics SASC
update statistics SDIV
update statistics SOBJ
update statistics SPRP
update statistics STRG
update statistics DXCPID
```

4. Select the Execute Query (F5) icon.

This executes all the previous commands against your <DBCSE> database. Output can be viewed in the Results window. The update statistics commands have completed successfully when the following message is displayed on the Results window:

This command did not return data, and it did not return any rows.

5. You may save the data to a file for use the next time.

**Follow these steps:**

1. Start up the SQL Server Management Studio from the SQL Server program group. Unless the system administrator ID/password has been changed; use **sa** as the login ID and a password to start the Management Studio.
2. From the DB drop down list box, select <DBDIR> (or your Coordination server database name).

3. Select File Open to open the file called dstats.sql. If you do not have dstats.sql, then type in the following commands manually:

```
update statistics DIRLOGON
update statistics DIRUSER
update statistics DIRENCY
update statistics DIRXCPID
```

4. Select the Execute Query (F5) icon.

This executes all the previous commands against your <DBDIR> database. Output can be viewed in the Results window. The update statistics commands have completed successfully when the following message is displayed on the Results window:

This command did not return data, and it did not return any rows.

5. You may save the data in a file for use the next time.

## Backing Up Your Database

To back up your SQL Server system data, follow the guidelines found in your SQL Server System Administrator documentation. A separate procedure is recommended to back up the CSE databases (default <DBCSE> and <DBDIR>) regularly.

### Create a Database Dump Device for Each CSE Database

See the following illustrations and the SQL Server documentation for the exact procedure:

- Create a database dump device for the <DBCSE> database using Management Studio, for example:

```
sp_addumpdevice 'disk', 'DBCSE_DB_DUMP', 'D:\DBCSE.DMP', 2
```

- Create a database dump device for the <DBDIR> database using Management Studio, for example:

```
sp_addumpdevice 'disk', 'DBDIR_DB_DUMP', 'D:\DBDIR.DMP', 2
```

### Create a Database Backup to Disk or Tape

These commands create a location to back up the database to disk. You can dump your data to a tape by issuing a similar command.

## Weekly Database Backups and Update Statistics Are Typical

Database backups should occur often enough for your particular work environment. Typical schedules include weekly database backup and update statistics. This may not be sufficient for your environment. Assign a database administrator (DBA) for your work area to schedule database backups.

1. Back up the Encyclopedia database <DBCSE> using the database dump device and the following Management Studio command:

```
dump database DBCSE to DBCSE_DB_DUMP
```

2. Back up the Coordination Server database <DBDIR> using the database dump device and the following Management Studio command:

```
dump database DBDIR to DBDIR_DB_DUMP
```

## Restoring Databases from a Database Dump Device

Back up stored information in files created when you create your dump device. The files, as described earlier, have the .DMP extension. For information on LOAD DATABASE command or restoring database in event of failure, see the SQL Server documentation for the LOAD DATABASE command. This gives instructions on restoring from a database dump device.

## Backing Up Transaction Logs

The CSE server installation creates .LOG files for both server databases. These transaction files record a log of updates and deletes against the databases. In the event of a failure, databases can be recovered using database backups and forward recovery of the transaction logs.

## Typical Log Backups Occur Daily

Back up the transaction logs using a transaction log dump device. The procedure is similar to the database backup. However, transaction logs should be backed up more frequently than databases. Typical schedules include daily backups.

## Log and Database Dump Devices Are Separate

Log dump devices should be separate from the database dump devices so that you can work with them separately.

## Create a Transaction Log Dump Device for Each CSE Database

### Follow these steps:

1. Create a transaction log dump device for the <DBCSE> database using Management Studio, for example:

```
sp_addumpdevice 'disk', 'DBCSE_LOG_DUMP', 'D:\ENCYLOG.DMP', 2
```

2. Create a transaction log dump device for the <DBDIR> database using Management Studio, for example:

```
sp_addumpdevice 'disk', 'DBDIR_LOG_DUMP', 'D:\DIRLOG.DMP', 2
```

## Create a Log Backup to Disk or Tape

These commands create a location to back up the transaction log information to disk. You can dump your logs to a tape by issuing a similar command.

1. Back up the Encyclopedia transaction log <DBCSE> using the transaction log dump device and the following Management Studio command:

```
dump transaction DBCSE to DBCSE_LOG_DUMP
```

2. Back up the Coordination transaction log <DBDIR> using the transaction log dump device and the following Management Studio command:

```
dump transaction DBDIR to DBDIR_LOG_DUMP
```

## Adjusting Database Size

As you increase the number of models in your encyclopedia, you may need to increase the size of the Encyclopedia database.

The simplest way to increase the size of a database is to expand it using the Microsoft SQL Management Studio.

**Note:** The default installation procedure creates a device size equal to your database size. After the installation, you may need to add a device and expand your database onto that device.

### Follow these steps:

1. Start the Microsoft SQL Management Studio icon from the SQL Server folder.
2. Highlight the database to which you want to add space. You may need to select plus(+) boxes to expand the Server Manager listing to show your database.
3. Select database properties.
4. If it is not already selected, select the General tab.

5. In the file properties section, verify that Automatically grow file is checked. Enter values for File Growth and Maximum file size.
6. Click OK.

If the transaction log for your database fills up, use this same procedure to add space to the log device, using the Transaction Log tab.

## Truncating Transaction Logs

SQL Server transactions logs are never truncated unless you issue specific commands to truncate them. As a result, the transaction logs fill up quickly and frequently.

### Truncating Transaction Logs Is Critical

It is critical that you organize a strategy for preventing transaction or system logs from filling up during peak times. You should become very familiar with the SQL Server system administrator documentation regarding transaction logs. Read all the documentation before establishing a strategy that works for you.

## Truncating Transaction Log Guidelines

Use the following guidelines to help manage transaction logs and avoid space constraints. These options truncate transaction logs to keep them as small as possible:

- `DUMP TRANSACTION` statement and `WITH TRUNCATE ONLY` class
- `DUMP TRANSACTION` statement and `WITH NO_LOG` clause
- `sp_dboption` system procedure with `trunc. log on chkpt.` option

### Back Up Database Before Truncating Logs

Back up the entire database before truncating transaction logs. All three of these options truncate the log for the specified database, however the options do not make a backup copy.

**Note:** `DUMP DATABASE` backs up the database and the transaction log but does not remove the inactive portion of the transaction log.

### When to Use `DUMP TRANSACTION` Statement and `WITH TRUNCATE ONLY` Class

The `DUMP TRANSACTION` statement using the `WITH TRUNCATE_ONLY` dumps the inactive part of the transaction log without making a backup copy of it. After you use `WITH TRUNCATE_ONLY`, the changes recorded in the active part of the transaction log are erased. Follow the `DUMP TRANSACTION` with a `DUMP DATABASE` statement, or you will be unable to recover the database in the future by using the transaction logs.

## When to Use DUMP TRANSACTION Statement and WITH NO\_LOG Clause

Use the DUMP TRANSACTION statement using the WITH NO\_LOG clause only when you have run out of space in the database and cannot execute DUMP TRANSACTION using the WITH TRUNCATE\_ONLY option. If you use the WITH NO\_LOG option, follow the DUMP TRANSACTION with a DUMP DATABASE statement, or you will be unable to recover future transaction logs.

## When to Use sp\_dboption System Procedure with trunc. log on chkpt. Option

Setting the trunc. log on chkpt. option is a way to save space, however it means that you can recover the state of the database only as it was at the last database dump. Turning on the trunc. log on chkpt. option causes the transaction log to be automatically truncated whenever SQL Server initiates the checkpoint checking process (about every minute). The default is off. You can turn it on by issuing the following Management Studio dialog:

```
sp_dboption DBCSE, 'trunc. log on chkpt.', true
```

Bring the SQL Server down, using the SQL Server Manager icon in the SQL Server program group after issuing this statement, and then restart the server.

**Note:** Although Setting the trunc. log on chkpt. option saves space, it means that you can recover the database only as it was at the last database backup. It is usually better to have a background task regularly back up the transaction log.

If you want to restore only from the last DUMP DATABASE command issued against your database, then you may want to seriously consider use of the trunc. log on chkpt. db option. If so, you may want to back up using the DUMP DATABASE command more frequently than once a week.

## Stopping SQL Server

**Follow these steps:**

1. Find MSSQLSERVER in the Services panel.
2. Select the Stop Services operation.

## Starting SQL Server

**Follow these steps:**

1. Find MSSQLSERVER in the Services panel.
2. Select the Start Service operation.

## SQL Server/Windows Database Customization

You may want to enhance performance after installation. This section discusses several database configurations you can implement to enhance performance. The basic installation of the Encyclopedia database produces a generic database configuration. The adjustments you can make to enhance performance are:

- Distributing data files across disks
- Accessing Public Interface views

### Adjusting Storage Parameters

To avoid performance degradation from tablespace fragmentation, or if you anticipate storing more than 500,000 objects in your encyclopedia, adjust the storage parameters for tables and their corresponding indexes.

The following are the key encyclopedia tables:

- DOBJ
- DTX
- DASC
- DSUBEX
- DNAME

To increase storage parameters for these tables follow the SQL Server procedures for space management.

### Distributing Data Files Across Disks

If you have multiple physical disks available for your database, you can improve performance by distributing data files between disks or by using disk striping.

Placing log files on one disk and data tables on another disk can reduce I/O contention.

To distribute data files across disks, indicate different disks at installation time, or follow the re-installation procedures.

**Note:** For more information, see *Configure a CSE Server Environment* in the *CA Gen Client Server Encyclopedia User Guide*.

During installation the log files are created. The log file directory and application database location are specified during installation.

If you will be performing uploads and checkouts with large models (more than 125,000 objects), your system will require additional virtual memory.

You also will need to ensure that you have adequate swap space. Use the online help in the Client Server Encyclopedia (CSE) Support Client to assist in determining an appropriate amount of swap space for Object Cache settings.

## Accessing Public Interface Views

Use Management Studio to query Public Interface views.

**Note:** To populate or update Public Interface views, use the add or update models functions within the CSE. If you add or update outside of the CSE you can corrupt the models, encyclopedia, or both. For more information, see the *Client Server Encyclopedia Public Interface User Guide*.

## Setting Options to Improve Performance

Use the `sp_configure` system procedure to set options for your CSE environment. For more information on system procedures, see the SQL Server online documentation.

The CSE installation does not modify default SQL Server system configuration parameters. There are currently no recommended settings other than those configured as the defaults when you install SQL Server. You may use different settings to improve performance.

## Common SQL Server System Procedures

Several system procedures provide information to database administrators.

Run these procedures from Management Studio:

| Procedures                 | Information                                                                                               |
|----------------------------|-----------------------------------------------------------------------------------------------------------|
| <code>sp_helpdevice</code> | Displays information about database devices and dump devices                                              |
| <code>sp_helpdb</code>     | Displays information about the relationship between database devices and segments in a specific database. |
| <code>sp_dboption</code>   | Allows you to set some specific options for your database, for example, trunc. Log on chkpt.              |



| Procedures      | Information                                              |
|-----------------|----------------------------------------------------------|
| sp_configure    | Allows you to configure your SQL Server system.          |
| sp_addumpdevice | Creates a dump device for a database or transaction log. |
| sp_dropdevice   | Drops a device previously created.                       |
| sp_who          | Information about processes currently running.           |
| sp_password     | Changes a user password.                                 |

## Oracle/Windows Routine Procedures

Database Administration includes procedures run on a routine basis and less frequently used procedures that help you to customize your database as your resource needs grow.

Several procedures will need to be performed frequently to keep the C/S databases running efficiently:

- Cleaning up your database
- Backing up your database
- Adjusting database size
- Stopping the database
- Starting the database

### Cleaning Up Your Database

To avoid wasting space in the encyclopedia, remove unusable models from your database by running a routine cleanup procedure. It may be a good idea to schedule a nightly (or off-hours) cleanup.

Unusable models may be left in the encyclopedia from unsuccessful procedures, including:

- Unsuccessful copy model operations
- Unsuccessful create model from subset operations
- Unsuccessful upload new model operations

For example, if you run out of space while creating a model, data associated with the model remains in the tables until you perform a cleanup.

## Models Remaining in Database Until Cleanup Is Run

Unusable models also remain from any delete model operations. Your database can become fragmented over time from several delete model operations, or from performing delete model operations on large models. Deleted models remain in the database until a cleanup procedure is run.

The cleanup program provided with CSE deletes rows associated with unusable models from the encyclopedia.

**Note:** The servers must be running when cleanup is performed. As with all command line commands, the optional parameters can be determined by typing only the command and pressing enter. Required and optional parameters used with that command will be displayed. For information on setting parameters, see *Configure a CSE Server Environment* in the *CA Gen Client Server Encyclopedia User Guide*.

### Follow these steps:

1. Back up your database. (Optional. See *Backing Up Your Database*.)
2. From the server software directory, enter the following command at a full screen prompt:

```
cleanup -u <userid>
```

where:

<userid> is the CSE ID of an Encyclopedia Administrator. For example:

```
cleanup -u user1
```

## Backing Up Your Database

Use the Oracle Export utility to back up the database regularly. The Oracle Import utility may be used to recover your database from Export backups. For more information, see your Oracle documentation.

## Adjusting Database Size

As you increase the number of models in your encyclopedia you may need to increase the size of the tablespaces for the Encyclopedia database. The following are the tablespaces you will need to increase:

- **iefsency\_idx** - Index Tablespace
- **iefsency\_tab** - Application Table Tablespace

The simplest way to increase the size of a tablespace is to add a data file to the tablespace. Use the Oracle procedures for adding a data file.

**Follow these steps:**

1. Shut down the servers.
2. Back up the database:  
Use the Oracle Export utility. Follow the instructions in your Oracle Documentation.
3. Delete the existing data files (since you are changing sizes):  
For example: `rm <datafile>`
4. Recreate the tablespaces using the new values:  
Run the CSE server re-installation.
5. Initialize the database:  
Run the CSE server re-installation.
6. Restore your data:  
Use the Oracle Import utility.

**Note:** For more information, see the Configure a CSE Server Environment in the *CA Gen Client Server Encyclopedia User Guide*.

## Stopping the Database

You will need to stop each database before performing a system shutdown. Stop the Message Dispatcher.

**Note:** For more information, see the *Client Server Encyclopedia User Guide*.

## Starting the Database

If you encounter a system shutdown, you may need to start the databases. To start the database, you need to do the following:

1. Double-click the CSE program group folder.
2. Double-click the Message Dispatcher icon.

## Oracle/Windows Database Customization Procedures

You may want to enhance performance after installation. This section discusses several database configurations you can implement to enhance performance. The adjustments you can make to enhance performance are:

- Adjusting the number of rollback segments and rollback tablespace size
- Adjusting the init<Encyclopedia database name>.ora
- Adjusting the number and size of log files
- Adjusting storage parameters
- Distributing data files across disks
- Accessing Public Interface views

### Adjusting the Rollback Segments and Rollback Tablespace

If you have 20-25 users or more for the encyclopedia you may want to add rollback segments and increase the size of the rollback tablespace. Also, if you are working with large models (more than 125,000 objects) you may want to increase the size of the rollback tablespace.

Multiple concurrent uploads create heavy activity. This can cause contention, which can degrade performance. If you plan to run multiple concurrent uploads you need to increase the number of rollback segments and increase the size of the rollback tablespace. The rollback segments for the Encyclopedia database are:

- ROLL1
- ROLL2
- ROLL3
- ROLL4

To add rollback segments, use the Oracle procedures for creating rollback segments.

Current rollback segments are placed in the tablespace:

iefsency\_rol1

To increase the size of the rollback segments tablespace, follow the procedures in Adjusting Database Size.

## Adjusting the init Encyclopedia database name.ora Oracle Windows

If performance degrades, for example, checkouts take longer than usual, you may need to adjust parameters in the init<Encyclopedia database name>.ora in the directory \$ORACLE\_HOME/dbs.

Alter the values for the following parameters:

- DB\_BLOCK\_BUFFERS
- LOG\_BUFFER

The system guidelines for memory with 192MB and greater than 192MB are to set the DB\_BLOCK\_BUFFERS to 8000 and set the LOG\_BUFFER to 1048576.

If you increase the LOG\_BUFFER you may also need to increase the size and number of database log files.

## Adjusting the Number and Size of Log Files

Default configuration establishes four log files. To increase the number and size of log files, use the Oracle procedure for adding log files.

## Adjusting Storage Parameters

To avoid performance degradation from tablespace fragmentation, or if you anticipate storing more than 500,000 objects in your encyclopedia, adjust the storage parameters for tables and their corresponding indexes. The key encyclopedia tables are:

- DOBJ
- DTX
- DASC
- DSUBEX
- DNAME

To increase storage parameters for these tables follow the Oracle procedures for space management.

## Distributing Data Files Across Disks

If you have multiple physical disks available for your database you can improve performance by distributing data files between disks.

Placing log files on one disk, indexes on another disk, and data tables on a third disk, can reduce I/O contention.

To distribute data files across disks, indicate different disks at installation time, or follow the re-installation procedures.

**Note:** For more information, see Configure a CSE Server Environment in the *CA Gen Client Server Encyclopedia User Guide*.

If you will be performing uploads and checkouts with large models (more than 125,000 objects), your system will require additional virtual memory. Work with your system administrator to allow for this by setting the kernel configuration parameter `maxdsiz`.

You also need to ensure that you have adequate swap space. Use the online help in the CSE Support Client to assist in determining an appropriate amount of swap space for Object Cache settings.

## Accessing Public Interface Views

**Follow these steps:**

1. Set `ORACLE_SID=<Encyclopedia database name>`.  
where: `<Encyclopedia database name>` is the name of the Encyclopedia database.
2. Open Oracle `sqlplus`.
3. Connect using the Oracle `USERID` and password you used during the installation.
4. Use Public Interface views to access tables and views.

**Important!** To populate or update Public Interface views, use the add or update models functions within the CSE. If you add or update outside of the CSE, you can corrupt the models, encyclopedia, or both.

During configuration of the CSE, an Oracle user ID and password were specified for connection to the Encyclopedia database. This is the `userid` associated with all encyclopedia tables and views. The CSE software connects to the databases using this user ID and password.

To avoid forcing users to qualify each view name, you can create a public synonym for each view.

You will need DBA authority to create Oracle `userids` in the Encyclopedia database (with connect authority), to grant these user IDs select access to views, and to create public synonyms.

**Note:** For more information, see the *CA Gen Client Server Encyclopedia Public Interface Reference Guide*.

## UNIX/Oracle Routine Procedures

Database Administration includes procedures run on a routine basis and less frequently used procedures that help you to customize your database as your resource needs grow.

Several procedures will need to be performed frequently to keep the CSE databases running efficiently:

- Cleaning up your database
- Backing up your database
- Adjusting database size
- Stopping the database
- Starting the database

### Cleaning Up Your Database

To avoid wasting space in the encyclopedia, remove unusable models from your database by running a routine cleanup procedure. It may be a good idea to schedule a nightly (or off-hours) cleanup.

Unusable models may be left in the encyclopedia from unsuccessful procedures, including:

- Unsuccessful copy model operations
- Unsuccessful create model from subset operations
- Unsuccessful upload new model operations

For example, if you run out of space while creating a model, data associated with the model remains in the tables until you perform a cleanup.

### Models Remaining in Database Until Cleanup Is Run

Unusable models also remain from any delete model operations. Your database can become fragmented over time from several delete model operations, or from performing delete model operations on large models. Deleted models remain in the database until a cleanup procedure is run.

The cleanup program provided with CSE deletes rows associated with unusable models from the encyclopedia. Use the following steps to clean up your database.

1. Back up your database.

2. `cleanup -u <userid>`

where: <userid> is the CSE ID of an Encyclopedia Administrator.

For example:

```
cleanup -u daacmo
```

**Note:** The servers must be running when cleanup is performed. As with all command line commands, the optional parameters can be determined by typing only the command and pressing enter. Required and optional parameters used with that command will be displayed. For information on setting parameters, see Configure a CSE Server Environment in the *CA Gen Client Server Encyclopedia User Guide*.

## Backing Up Your Database

Use the Oracle Export utility to back up the database regularly. The Oracle Import utility may be used to recover your database from Export backups.

## Adjusting Database Size

As you increase the number of models in your encyclopedia you may need to increase the size of the tablespaces for the Encyclopedia database. The tablespaces you will need to increase are:

- **iefsency\_idx**- Index Tablespace
- **iefsency\_tab**- Application Table Tablespace

The simplest way to increase the size of a tablespace is to add a data file to the tablespace. Use the Oracle procedures for adding a data file.

If you prefer to maintain a single data file for a tablespace you will need to do the following:

**Follow these steps:**

1. Shut down the servers.
2. Back up the database. Use the Oracle Export utility.
3. Delete the existing data files (since you are changing sizes):

Use the UNIX Remove command.

For example: `rm <datafile>`

4. Recreate the tablespaces using the new values:  
Run the CSE server re-installation.



5. Initialize the database:  
Run the CSE server re-installation.
6. Restore your data:  
Use the Oracle Import utility.

**Note:** For more information, see the Configure a CSE Server Environment in the *CA Gen Server Encyclopedia User Guide*.

## Stopping the Database

**Follow these steps:**

1. Stop the Message Dispatcher, following the instructions for stopping the servers.
2. Set the SID for the required database:
  - a. If you are using Bourne Shell or Korn Shell, set and export the SID. For example:  

```
ORACLE_SID=<database name>
export ORACLE_SID
```
  - b. If you are using C Shell, use the setenv command to set the SID. For example:  

```
setenv ORACLE_SID
```
3. Type the Oracle command:  

```
sqlplus /nolog
connect sys as sysdba
shutdown
exit
```
4. Run Steps 2 and 3 for the Coordination and the Encyclopedia database.

**Note:** If there are processes pending against the database, the system may not respond to a normal shutdown. If this occurs, type the following:  

```
sqlplus shutdown immediate
```

For more information, see the *Client Server Encyclopedia User Guide*.

## Starting the Database

**Follow these steps:**

1. Login as the CSE Administrator using the account set up previously by the UNIX administrator.
2. Set the SID for the required database.

- If you are using Bourne Shell or Korn Shell set and export the SID. For example:

```
ORACLE_SID=<database name>
```

```
export ORACLE_SID
```

- If you are using C Shell use the setenv command to set the SID. For example:

```
setenv ORACLE_SID <database name>
```

3. Type the Oracle command:

```
sqlplus /nolog
```

```
connect sys as sysdba
```

```
startup pfile=<location of initdb.ora file>;
```

```
exit
```

4. Run Steps 2 and 3 for the Coordination and the Encyclopedia databases.
5. Start the servers. Follow the instructions for starting the CSE software provided previously in this guide.

## UNIX/Oracle Database Customization Procedures

You may want to enhance performance after installation. This section discusses several database configurations you can implement to enhance performance. The basic installation of the Encyclopedia database produces a generic Oracle database configuration. The adjustments you can make to enhance performance are:

- Adjusting the number of rollback segments and rollback tablespace size
- Adjusting the init<Encyclopedia database name>.ora
- Adjusting the number and size of log files
- Adjusting storage parameters
- Distributing data files across disks
- Accessing public interface views

### Adjusting the Rollback Segments and Rollback Tablespace

If you have 20 - 25 users or more for the encyclopedia, you may want to add rollback segments and increase the size of the rollback tablespace. Also, if you are working with large models (more than 125,000 objects), you may want to increase the size of the rollback tablespace.

Multiple concurrent uploads create heavy activity. This can cause contention, which can degrade performance. If you plan to run multiple concurrent uploads, you need to increase the number of rollback segments and increase the size of the rollback tablespace. The following are the existing rollback segments for the Encyclopedia Database:

- ROLL1
- ROLL2
- ROLL3
- ROLL4

To add rollback segments, use the Oracle procedures for creating rollback segments.

Current rollback segments are placed in the tablespace:

`iefsency_rol`

To increase the size of the rollback segments tablespace, follow the procedures in Adjusting Database Size.

## Adjusting the `init<Encyclopedia database name>.ora`

If performance degrades, for example, checkouts take longer than usual, you may need to adjust parameters in the `init<Encyclopedia database name>.ora` in the directory `$ORACLE_HOME/dbs`.

Alter the values for the following parameters:

- `DB_BLOCK_BUFFERS`
- `LOG_BUFFER`

The system guidelines for memory with 192MB and greater than 192MB are to set the `DB_BLOCK_BUFFERS` to 8000 and set the `LOG_BUFFER` to 1048576.

If you increase the `LOG_BUFFER`, you may also need to increase the size and number of database log files.

## Adjusting the Number and Size of Log Files

Default configuration establishes four log files. To increase the number and size of log files, use the Oracle procedure for adding log files.

## Adjusting Storage Parameters

To avoid performance degradation from tablespace fragmentation, or if you anticipate storing more than 500,000 objects in your encyclopedia, adjust the storage parameters for tables and their corresponding indexes. The following are the key encyclopedia tables:

- DOBJ
- DTX
- DASC
- DSUBEX
- DNAME

To increase storage parameters for these tables, follow the Oracle procedures for space management.

## Distributing Data Files Across Disks

If you have multiple physical disks available for your database you can improve performance by distributing data files between disks.

Placing log files on one disk, indexes on another disk, and data tables on a third disk, can reduce I/O contention.

To distribute data files across disks, indicate different disks at installation time, or follow the re-installation procedures.

**Note:** For more information, see Configure a CSE Server Environment in the *CA Gen Client Server Encyclopedia User Guide*.

During installation the log files are created in \$ORACLE\_HOME/dbs. The log file directory, the index tablespace and application table tablespace location are specified during installation.

If you will be performing uploads and checkouts with large models (more than 125,000 objects), your system will require additional virtual memory. Work with your UNIX Administrator to allow for this by setting the kernel configuration parameter maxdsiz.

You also will need to ensure that you have adequate swap space. Use the online help in the CSE Support Client to assist in determining an appropriate amount of swap space for Object Cache settings.

## Accessing Public Interface Views

If you will use the Oracle SQLPLUS utility to perform queries against the CSE databases, you will need to access the database with the following statement:

```
sqlplus <userid>/<password>
```

where:

<userid> is the Oracle userid for the Encyclopedia Database Administrator.

<password> is the Oracle password for the Encyclopedia Database Administrator.

**Important!** To populate or update Public Interface views, use the add or update models functions within the CSE. If you add or update outside of the CSE, you can corrupt the models, encyclopedia, or both.

During configuration of the CSE, an Oracle user ID and password were specified for connection to the Encyclopedia database. This is the user ID associated with all encyclopedia tables and views. The CSE software connects to the databases using this user ID and password.

To avoid forcing users to qualify each view name, you can create a public synonym for each view.

You will need DBA authority to create Oracle user IDs in the Encyclopedia database (with connect authority), to grant these userids select access to views, and to create public synonyms.

**Note:** For more information, see the *CA Gen Client Server Encyclopedia Public Interface Reference Guide*.



# Chapter 8: Server Administration

---

You can perform several server administration tasks to maintain your system and to enhance performance. This section provides suggestions for:

- What to do if a server stops
- Working with the <iefmd>.ini File
- Starting multiple instances of Encyclopedia utilities
- Capturing server utility errors during unexpected shutdowns
- Determining status of a server

## What Happens If a Server Stops?

The Client Server Encyclopedia (CSE) contains restartable and non-restartable servers:

| Restartable Servers    | Non-restartable Servers |
|------------------------|-------------------------|
| Coordination Server    | ID Server               |
| Construction Server    | Lock Server             |
| Model/Subset Server    |                         |
| User/Group Server      |                         |
| Version Control Server |                         |

If an unexpected condition causes a restartable server or server process to stop, another instance of the server will be started on demand with the next request. If there is not an automatic restart, use the following steps to restart the servers.

For non-restartable server processes, you must shutdown and then restart the servers using the following steps. Manually restarting a non-restartable server after an unexpected termination may affect database integrity.

If the ID server process or the Lock server process stops:

**Follow these steps:**

1. If possible, wait for any upload or checkout processing to finish.
2. Stop the servers.

3. Restart the servers

**Note:** For more information, see Configure a CSE Server Environment in the *CA Gen Client Server Encyclopedia User Guide*.

## Working with the <iefmd>.ini File

Adjust or work with the <iefmd>.ini file to perform several tasks, which help maintain your system and enhance performance.

### What Is the <iefmd>.ini File?

The <iefmd>.ini file is the Message Dispatcher configuration file, which contains information about the way the servers and their utilities are started. You configure the file during the server installation. The default file name is iefmd.ini. If you entered a different name during the installation, the file name will be what you entered.

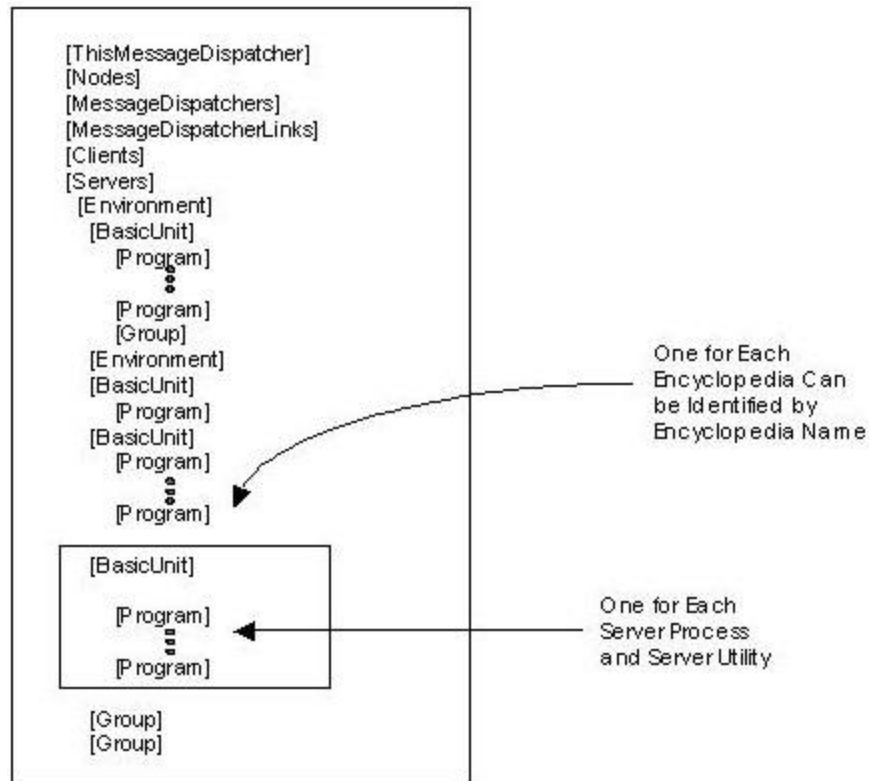
The following figure outlines the <iefmd>.ini file. It focuses on the Server Environment section of the file that contains all references to the server utilities. You can adjust this section after the installation.

Other sections of the file are set during the installation and can be changed by reinstalling the software and selecting different parameters.

**Note:** For more information, see Configure a CSE Server Environment in the *CA Gen Client Server Encyclopedia User Guide*.



The following is an outline of the <iefmd>.ini file:



After the installation and as you use your CSE, adjust the <iefmd>.ini file to accommodate your changing needs. Work with the <iefmd>.ini file for the following tasks:

- Starting multiple instances of Encyclopedia utilities
- Capturing server utility errors during unexpected shutdowns
- Determining status of a server
- Defining an exclusive message dispatcher for the coordination server
- Assigning port numbers to Message Dispatchers
- Reviewing Directory and Encyclopedia group name configurations

## Defining an Exclusive Message Dispatcher for the Coordination Server

In a multi-Message Dispatcher network, which implies multiple Encyclopedias, you may want to define a Message Dispatcher that is dedicated to the coordination server for network control purposes. Having this dedicated Message Dispatcher allows you to start and stop multiple Encyclopedias and to maintain independent access to the coordination server.

**Note:** In a single Encyclopedia network, having a dedicated message dispatcher is a waste of CPU resources.

You can edit the Message Dispatcher <iefmd.ini> file using any text editor.

## Changing the DIRGROUP and ENCYGROUP Parameters

Use a text editor to view the current setting for the DIRGROUP in the Message Dispatcher initialization file. Although the software allows you to use another name, you should always use the default setting (DIR) for DIRGROUP because there is usually no need to have more than one directory for a collection of encyclopedias.

**Note:** If you do not use a setting other than the default for DIRGROUP, the client process can override the DIRGROUP name by resetting the parameter through the IEF\_DIRGROUP environment variable.

You can also use a text editor to review the settings for ENCYGROUP parameters in the <iefmd.ini> file. You can have more than one encyclopedia referenced in the <iefmd.ini> same file. If your file contains multiple encyclopedias, the names you define for ENCYGROUP must be unique.

## Assigning Port Numbers to Message Dispatchers

If only one Message Dispatcher exists on each machine, you can assign the same port number for all MDs. However, if more than one MD exists on the same server, you must assign separate port numbers for each MD because the combination of IP address and port number must be unique.

The IP name or address can be in Internet Packet version 4 (IPv4) or version 6 (IPv6) format. The following Universal Resource Locator (URL) format is also supported in the Hostname field:

[host-name-or-IP-address] port-number

You can use the same port number for the RDS node names in all scenarios (one MD or multiple MDs on one server). However, configuring one RDS for each Message Dispatcher or encyclopedia is recommended. When you have different logs for each Message Dispatcher or encyclopedia, tracking the origin of messages is much easier.

## Starting Multiple Instances of Encyclopedia Utilities

You may improve the throughput of your CSE environment by allowing more than one instance of Encyclopedia utilities.

### Encyclopedia Utility Program Names

Encyclopedia utilities are the part of the Encyclopedia server that provide on-demand tasks for the following functions:

| Function                           | Program Name |
|------------------------------------|--------------|
| Upload                             | UTLUP        |
| Diagnostic                         | UTLDIAG      |
| Download                           | UTLDOWN      |
| Verify last upload                 | UTLVERUP     |
| Delete model                       | UTLDELM      |
| Copy model                         | UTLCPYM      |
| Override checkout status           | UTLOVER      |
| Create model from subset           | UTLCRMFS     |
| Expansion conflict report          | UTLEXP RP    |
| Model and subset statistics report | UTLEVAL      |
| Verify delta                       | UTLVERD      |
| Generate delta                     | UTLGEND      |
| Generate code                      | UTLGENCD     |
| Adopt/trial adopt object           | UTLADPT      |
| Migrate/trial migrate object       | UTLMIGR      |
| Compare aggregate object           | UTLCOMP      |
| Consistency check report           | UTLCONCK     |
| Action block use report            | UTLACBLKU    |

### Allowing the Server to Process Multiple Requests

Starting another instance of an Encyclopedia Server utility allows you to take advantage of running another image of the server software. This enables your system to handle multiple server requests concurrently.

To allow the CSE software to start another instance of an Encyclopedia utility automatically, you can change the default values for the min/max instance and the queue depth in your <iefmd>.ini file. The following defines each of these values:

| Parameter   | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Mininstance | When the MessageDispatcher (MD) is started, the MinInstance statement instructs the MD on how many instances of the server utility to start initially. The MinInstance for Encyclopedia utilities should always be 1.                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Maxinstance | The MaxInstance statement instructs the MD on the maximum number of instances of the server utility allowed to run at any given time. Maxinstance=3 is recommended.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| QueueDepth  | <p>The QueueDepth statement tells the MD when to start additional instances of a server utility. It bases this on the backlog of client requests.</p> <p>A QueueDepth value of one or more tells the MD to examine the backlog. When the backlog hits this QueueDepth value, the MD starts the next server instance. (As long as your MaxInstance is greater than one.) Use one when MaxInstance is set to one and use a value of one or greater when the MaxInstance is greater than the MinInstance.</p> <p>A QueueDepth value of zero instructs the MD not to examine the request backlog. In this case it will never start additional instances on demand.</p> |

## Calculating the MinInstance and MaxInstance Values

The following are the installation default values for the MinInstance, MaxInstance, and the QueueDepth for the Encyclopedia Server programs and utilities. These values mean that requests are funneled to one server process and subsequent requests are placed in the pending queue until the previous request is complete. The installation defaults are:

| Program         | MinInstance | MaxInstance | Queue Depth |
|-----------------|-------------|-------------|-------------|
| Server programs | 1           | 1           | 1           |

| Program          | MinInstance | MaxInstance | Queue Depth |
|------------------|-------------|-------------|-------------|
| Utility programs | 1           | 3           | 1           |

## General Guidelines for MaxInstance

In general, you can set the MaxInstance value to the number of requests that you want to process concurrently. However, the number of concurrent requests should be kept at a minimum to avoid excessive CPU, I/O, and memory use.

## Coordinating Resource Intensive Processes

In addition to allowing more than one instance of Encyclopedia Utilities, it is a good idea to establish a site practice of planning and scheduling resource intensive processes (preferably during off hours).

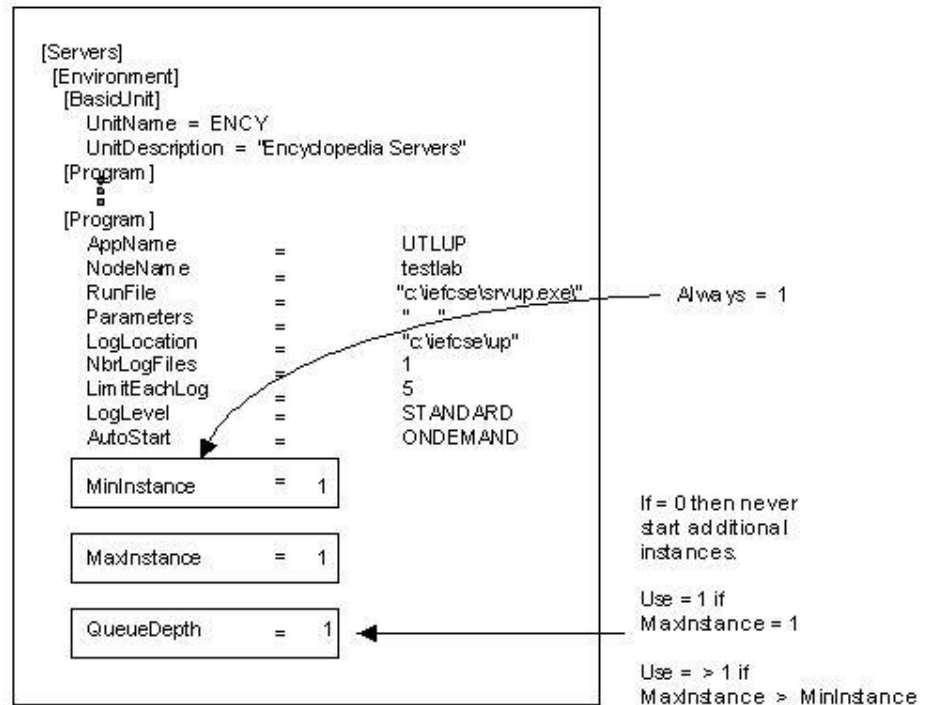
The following values for MinInstance, MaxInstance, and QueueDepth can be applied along with coordinating your resource intensive processes:

| MinInstance | MaxInstance | QueueDepth | Should support |
|-------------|-------------|------------|----------------|
| 1           | 1           | 1          | 4 - 8 users    |
| 1           | 2           | 1          | 6 - 12 users   |
| 1           | 3           | 1          | 10 - 20 users  |

## Changes to a Sample Utility Section in the <iefmd>.ini File

An Encyclopedia Servers Basic Unit exists for every Encyclopedia defined in your <iefmd>.ini file. You can set the MinInstance, MaxInstance and QueueDepth for each of the utilities within a Basic Unit.

The following figure identifies where changes can be made in a utility section of an <iefmd>.ini file:



**Note:** The <iefmd>.ini file is in text format. It must contain a new line as the last character in the file. Also, be sure that any text editor used to modify the file does not insert a CTRL-Z character at the end of the file.

## Adjusting the <iefmd>.ini File

Use the following steps and the previous information on calculating the MinInstance and MaxInstance values to adjust the <iefmd>.ini file to allow more than one instance of an Encyclopedia Utility:

### Follow these steps:

1. Open the <iefmd>.ini file using any text file editor.
2. Locate the utility by searching for the utility program name:  
= <utility program name>
3. Change the MinInstance value, the MaxInstance value, and the QueueDepth values.

### More Information:

[Encyclopedia Utility Program Names](#) (see page 91)

## Capturing Server Utility Errors During Unexpected Shutdowns

On UNIX and Windows CSE Servers, the error messages are directed to the Message dispatcher's standard error output.

## Determining Status of a Server

If a resource intensive process has been started and you are not sure if the process is still active, you can determine the status of the server by using the Message Dispatcher Monitor tool.

This tool provides a snapshot of the server status at the time that you run the MD Monitor. It places the snapshot information in a log file called MON1.log in your current directory. You can then review the log file to see if the server is busy, idle, or unavailable.

## What Are Resource Intensive Server Processes?

Most requests from clients to servers are completed immediately, however there are several requests that start resource intensive processes. The time to complete these processes will vary depending upon your system configuration and system load. With experience, you will have a good idea for the amount of time that each process takes.

Resource intensive processes using Encyclopedia Utilities are:

- Apply to Parent
- Checkout/Download
- Copy Model
- Checkout Report
- Create Model From Subset
- Delete Model
- Generate Code
- Generate DDL
- Generate Delta
- Load to Child
- Model Cross Copy Apply
- Model Cross Copy Extract
- Override Model Checkout
- Override Subset Checkout

- Resend Last Update to Parent
- Update/Upload

Each of these has its own utility process started on demand by the Message Dispatcher. Requests to the Encyclopedia Utilities are processed immediately if that utility is not currently acting on a previous request. Any subsequent requests that occur while the utility is engaged are placed in the pending queue. A resource intensive process will totally engage the utility for as long as it takes to complete the request.

It is a good practice to plan and schedule these activities not to run during peak hours. You can also configure your system to allow additional instances of an Encyclopedia Utility to run when needed. For more information, see [Starting Multiple Instances of Encyclopedia Utilities](#).

## Running the Monitor Report

### Follow these steps:

1. Change to the server install directory.
2. Run the monitor status report using the command:

```
monrpt
```

The MD Monitor tool starts. The report is created and overwrites MON1.log file.  
The MD Monitor tool closes.

## Finding the Utility Session

The MD Monitor facility records extensive information about your system that it gets from the Message Dispatcher. All of the information is recorded in sections within the log file.

The section that describes the utility session is the one that you will use to determine the utility's status. This section begins with the application name of the utility.

### Follow these steps:

1. From your current directory open MON1.log using any editor.
2. Locate the Encyclopedia Utility by searching for the utility program name:  
= <utility program name>
3. The second occurrence of the utility program name contains the status information.
4. Review the status information.

### More information:

[Encyclopedia Utility Program Names](#) (see page 91)



## What to Locate in the MON1.log File

To determine what the server status is and what it is actively processing, you will need to look for several lines within the section defining the server session. These lines are defined and then highlighted in the following illustration of an upload utility (UTLUP) session:

| Status              | Description                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Shared Queue List   | Contains messages waiting to be sent to any IDLE instance of the server. This number (combined with the number from the Individual Queue List) represents the total number of pending requests to the server.                                                                                                                                                                                                                |
| Sequence Queue List | Represents the number of concurrent processes that have run since the servers were started. It keeps a chain of messages together for processing by the same server instance. (This is typically used during file transfer operations.)<br><br>The number of entries in the Sequence Queue List gives you an indication of how many instances of the server to establish and use as the MinInstance in the <iefmd>.ini file. |
| Queue Length        | Waiting messages targeted specifically to a server instance.                                                                                                                                                                                                                                                                                                                                                                 |
| Sequence Q Handle   | Lists the index into the Sequence Queue List for a file transfer in process. If the value is <none>, then the server is not processing a file transfer or it is currently processing the last message in the sequence.                                                                                                                                                                                                       |

## Interpreting the MON1.log Information

After you locate the Encyclopedia Server session definition, you can look for the Status line within that section.

Status of the server utility is indicated as one of the following:

| Status      | Description                                                                                             |
|-------------|---------------------------------------------------------------------------------------------------------|
| BUSY        | The server utility is working on a request. It will respond to the client when the request is complete. |
| IDLE        | The server utility is waiting for a request. It cannot find any messages to process.                    |
| UNAVAILABLE | Startup not completed yet. Not available for service.                                                   |

### If the Server Utility is BUSY

If the server utility is BUSY, it is processing a request. You can determine what process is running by looking for the DestRPC line. This line lists the Remote Procedure Call (RPC) code for the process that is active. The following codes are for the resource intensive processes of the Encyclopedia Server:

| RPC Code | Resource Intensive Encyclopedia Server Processes |
|----------|--------------------------------------------------|
| 3110     | Apply to Parent                                  |
| 3106     | Checkout/Download                                |
| 3115     | Checkout Report                                  |
| 3102     | Copy Model                                       |
| 3103     | Create Model From Subset                         |
| 3101     | Delete Model                                     |
| 5092     | Generate All Code                                |
| 5094     | Generate All Cooperative Code                    |
| 5068     | Generate All DDL                                 |
| 5091     | Generate Code                                    |
| 5093     | Generate Cooperative Code                        |
| 3118     | Generate Delta                                   |
| 3111     | Load to Child                                    |
| 3120     | Model Cross Copy Apply                           |
| 3116     | Model Cross Copy Extract                         |
| 3104     | Override Model Checkout                          |

| RPC Code | Resource Intensive Encyclopedia Server Processes |
|----------|--------------------------------------------------|
| 3105     | Override Subset Checkout                         |
| 3112     | Resend Last Update to Parent                     |
| 3109     | Update/Upload                                    |

If the RPC code is one that is not listed here, take another snapshot with the MD Monitor tool. If the same RPC code continues to display, the server or DBMS may be hung. Contact your Product Support representative.

### If the Server Utility is IDLE

If the server utility is IDLE it may not have any requests to process, however it may be processing a file transfer. You can determine whether the server is processing a file transfer by looking for the Sequence Q Handle line.

If the Sequence Q Handle line is not equal to <none>, the server is engaged with a file transfer. Once a server instance starts a file transfer, it remains engaged until the file transfer is complete. A file transfer should take between 0 - 5 minutes, depending on your system load, network traffic, and the file size. You may want to take several snapshots with the MD Monitor tool over the next several minutes to see if the status changes. If the status does not change, contact your Product Support representative.

If the server utility Status is IDLE and Sequence Q Handle = <none> and there are no messages in the Shared Q, and a client is hung, contact your Product Support representative.

### If a Server Utility is UNAVAILABLE

If a server utility is UNAVAILABLE, check to see if it is running by selecting the server utility window. If the server utility is not running look in the MD window and the server utility log file, note any error messages and contact your Product Support representative with the information. If the server utility is running, contact your Product Support representative.

The following is an illustration of upload utility session status:

```
[2] Application Name           = UTLUP
    Session Type              = SERVER
    Environment Index         = 1
    BasicUnit Index          = 2
    Restartable               = ONDEMAND
    Recv Obituaries           = NONE
    MD Handle                 = 1
    Run Location              = LOCAL MD
    Program Index             = 1
    SHARED QUEUE LIST:        = (0 entries)
    SEQUENCE QUEUE LIST:      = (1 entries)
    [1] Entry NOT currently in use:
    SESSION INSTANCE LIST:    = (1 entries)
    [1] Session Instance Entry:
    Option List:
        HTI Handle            = 2
        [1] Description       = Upload Utility Server
        [2] Version           = 5.3
        [3] Begin Interface   = 1
        [4] Category          = 1
        [5] Range From RPC    = 1001
        [6] Range To RPC      = 1167
        [7] End Interface     = <blank>
    Sequence Q Handle         = <none>
    Status                    = BUSY
    Processing Request
        PacketLength          = 342
        PacketType            = TASK_DATA
        PacketCommand          = XMDS_CLIENT_FWD_REQ
        PacketSeq              = 2
        PacketSeqContinued     = FALSE
        DestEnvironment        = 1
        DestGroup              = 2
        DestInterface          = 1
        DestRPC                = 1067
        DestMD                 = 1
        DestSession            = 2
        DestInstance           = 0
        ReplyMD                = 1
        ReplySession           = 8
        ReplyInstance          = 1
    Queue Length              = 0
```

## Setting the CSE LogLevel

The Client Server Encyclopedia (CSE) has a communications logging facility that may be useful in diagnosing difficult problems. Setting the LogLevel parameter for each server utility will write errors and warnings, along with the time at which they occur, to a log file.

Log files are identified in the <iefmd>.ini file through the following entries:

- LogLocation
- NbrLogFiles

- LimitEachLog
- LogLevel

See the sample <iefmd>.ini file in this guide to refer to log file entries.

## LogLocation

By default, Log files are located in the directory where your executable files reside. This directory is indicated next to the LogLocation entries in the <iefmd>.ini.

## NbrLogFiles

The number of log files field, NbrLogFiles, enables a history of logging information. The information will be retained in file names appended with 01 to 99. If the <iefmd> value supplied is one (1), only one logfile for the application will be created and no rollover will occur. This field works in conjunction with the LimitEachLog field.

## LimitEachLog

Log file limits can be set with the LimitEachLog field. A number from one (1) to N can be specified, or the word UNLIMITED can be specified. UNLIMITED allows the first log file created to grow to the capacity of the disk.

A numeric entry indicates the number of kilobytes to retain in the log file. Once the limit is reached, logging switches (rolls over) to the next log file. Log files continue to roll over, giving you the ability to calculate the maximum disk space that will ever be used, regardless of how much you use the CSE or how long it stays running.

For example, the following defines your log files in your <iefmd>.ini file:

```
LogLocation = "<your-string>"
NbrLogFiles = 1
LimitEachLog = 5
LogLevel = STANDARD
```

This illustration will continue to use a single log file, but will grow to a maximum of 5 kilobytes. It will then close and reopen the log file and continue logging.

## LogLevel

The amount of information written to this file depends on the LogLevel set in the <iefmd>.ini file. The default setting is STANDARD. This is automatically set when the message dispatcher is installed. LogLevel parameters available are:

| LogLevel Parameters | Description                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| STANDARD            | This logging level is the default setting when the Message Dispatcher is installed. Errors and warnings are recorded along with the time at which they occurred.                                                                                                                                                                                                                                                  |
| SUPPORT             | This logging level provides additional information beyond that of STANDARD logging and should be used during the initial stages of problem diagnosis. SUPPORT logging provides internal descriptions of program operation during execution as well as a record of all communications message traffic. Use this level of logging to diagnose the symptoms of the problem with your Product Support representative. |
| DEBUG               | This logging level provides you with information regarding the internal program data flow need to analyze a problem symptom. This should not be used except with the assistance of Product SUPPORT.<br><b>Important!</b> <i>DEBUG produces large log files. If used routinely, the DEBUG logging level will degrade performance and consume large amounts of disk space for log records.</i>                      |

The following table identifies the extent of each LogLevel setting:

| Information Provided | STANDARD | SUPPORT | DEBUG |
|----------------------|----------|---------|-------|
| Time                 | x        | x       | x     |
| Error                | x        | x       | x     |
| Warning              | x        | x       | x     |
| Note                 | x        | x       | x     |

| Information Provided | STANDARD | SUPPORT | DEBUG |
|----------------------|----------|---------|-------|
| Comm                 |          | x       | x     |
| Info                 |          | x       | x     |
| Entry                |          |         | x     |
| Exit                 |          |         | x     |
| Input                |          |         | x     |
| Output               |          |         | x     |
| Return               |          |         | x     |
| Data                 |          |         | x     |
| Dump                 |          |         | x     |

Contact Technical Support for extent of LogLevel settings for the following:

- Heap
- Malloc
- Free
- Stats

The following list defines information provided in the communication log files. See the previous table to identify which LogLevel setting provides the information:

- **Time**-System date and time of the error.
- **Error**-Fatal or serious conditions.
- **Warning**-Non-fatal, but important conditions
- **Comm**-Communications traffic (send, received)
- **Info**- Internal program flow information (for example: all servers busy, message will be queued.)
- **Entry**-Name of subroutine entered.
- **Exit**-Name of subroutine exited.
- **Input**-Variable values passed into the subroutine.
- **Output**-Variable values passed out of the subroutine.
- **Return**-Return value of the subroutine, generally MDS\_OK or error code.
- **Data**-Key variable values internal to the subroutine.
- **Dump**-All variable values relevant to the subroutine.

- **Heap**-Heap checking during memory allocations and de-allocations.
- **Malloc-New** memory allocations (where: file,line number)
- **Free**-Memory de-allocations (where: file,line number)
- **Stats**-Memory utilization statistics upon program termination.



# Chapter 9: Troubleshooting

---

Use this section to identify common performance questions. For more information, see the referenced chapters, sections, and guides.

## Common Performance Questions

The following table lists the common performance questions along with their answers.

| Question                                                                                                    | Answer                                                                                                                                                                                                 |
|-------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Unusable models are in my Encyclopedia. How did they get there and how do I get rid of unusable models?     | Unusable models in the Encyclopedia are from unsuccessful operations (copy model, create model, or upload new model) or they are from delete model operations. Run the cleanup command to remove them. |
| Several delete model operations have been performed. Do I need to do anything to the Encyclopedia Database? | Deleted models are marked as unusable in the Encyclopedia Database. Run the cleanup command to remove them from the database.                                                                          |
| Checkouts are taking longer than usual. How can I improve performance on checkouts?                         | Reorganize the Encyclopedia database.                                                                                                                                                                  |
| What can I do to improve degraded performance?                                                              | Reorganize the Encyclopedia database. Also, schedule resource intensive processes during non-peak hours.                                                                                               |
| What do I need to do when database activity has been heavier than usual?                                    | Reorganize the Encyclopedia database.                                                                                                                                                                  |
| What do I do if one or more of my servers stop running?                                                     | If it is a restartable server process, the server will restart with the next request. If it is a non-restartable server process, you will need to shutdown and then restart the servers.               |
| There is an internal connect error on my client. What happened to the server?                               | The server went down and will need to be restarted.                                                                                                                                                    |
| How can I improve throughput of the CSE environment?                                                        | Allow more than one instance of the Encyclopedia Server utilities to run. Set the MinInstance/ MaxInstance and Queue Depth values in the <iefmd>.ini file.                                             |

| Question                                                                                                                                                                                                                                                         | Answer                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| How can I allow the system to handle multiple server requests concurrently?                                                                                                                                                                                      | Allow more than one instance of the Encyclopedia Server utilities to run. Set the MinInstance, MaxInstance, and Queue Depth values in the <iefmd>.ini file.                                                                                                                                                                                                                                                                                                                                                                                                  |
| How do I run two or more of the following requests at the same time: upload, download, delete model, copy model (for example: 2 uploads at the same time)?                                                                                                       | Allow more than one instance of the Encyclopedia Server utilities to run. Set the MinInstance, MaxInstance, and Queue Depth values in the <iefmd>.ini file.                                                                                                                                                                                                                                                                                                                                                                                                  |
| One of the server processes was started but I do not know if it is still running. How do I find out if a resource intensive process is still active?                                                                                                             | Use the monrpt command to take a snapshot of server activity.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| I get an MD connect error when a client tries to access the servers. My servers are running but all clients cannot connect. (One client may be able to access the servers.)<br>All clients could connect to the servers before. What happened to the connection? | Verify the TCP/IP IP addresses for all users in the system.<br>It is possible that a new user inadvertently set up their client with the IP address of the server instead of a unique IP address. This client may be able to connect to the servers, however all other clients on the network will try to connect to that client machine with the server's IP address.<br>Ensure that your server TCP/IP IP address is unique throughout the network. Continue to follow a management practice for all TCP/IP IP addresses and educate users on the network. |
| How can I better diagnose problems?                                                                                                                                                                                                                              | You can review the log files created by setting the LogLevel in the <iefmd>.ini file.                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Characters are not displaying correctly on the screen. I get unusual characters where there should be text values. What happened to my model?                                                                                                                    | Your model was created on platform other than the one you are currently working on. It was created in a previous release and has not yet been converted.<br>Look at the model on the same platform on which it was built. Alternatively, run the conversion command.                                                                                                                                                                                                                                                                                         |

**More information:**

[Cleaning Up Your Database](#) (see page 63)

[What Happens If a Server Stops?](#) (see page 87)

[Starting Multiple Instances of Encyclopedia Utilities](#) (see page 91)

[Determining Status of a Server](#) (see page 95)





# Index

---

## <

- <iefmd>.ini file
  - assigning port numbers to Message Dispatchers • 90
  - changing DIRGROUP and ENCYGROUP parameters • 90
  - defining exclusive Message Dispatcher for Coordination Server • 90
  - definition • 88
  - determining status of server • 95

## A

- adding a new model, encyclopedia function • 54
- adoption, encyclopedia function • 53
- anticipating and avoiding problems • 12
- avoid
  - common network problems • 12
  - contention and model corruption • 37
  - mass upload • 48

## B

- back up and recovery of CSE tables • 30
- backup options • 30
- backup, guidelines • 31
- bottlenecks among system resources, identifying • 9

## C

- cache and threshold settings • 33
- calculating MinInstance and MaxInstance values • 92
- capacity planning
  - checklist • 26
  - DOBJ table • 21
  - DSUBEX table • 21
  - estimating CSE size • 24
  - host vs. CSE table comparison • 21
  - large tables for user data • 23
  - relative size of indexes to tables • 23
  - schema tables • 21
  - small tables for user data • 22
  - splitting files across multiple physical drives • 25
  - table sizes • 21
  - variable-sized tables for user data • 23
- causes of contention • 35
- checklist, capacity planning • 26

- checkout status, overriding • 51
- cleanup
  - CSE • 32
  - utility • 31
- cleanup program, cleanup database • 79
- collecting performance statistics
  - monitoring encyclopedia contention • 20
  - monitoring encyclopedia profile • 18
  - monitoring encyclopedia tables • 15
  - monitoring Encyclopedia usage • 19
  - monitoring operating system resources • 19
  - Oracle • 15
  - SQL Server • 17
- collecting performance statistics - SQL Server • 17
- common performance questions • 105
- concurrency matrix, encyclopedia • 36
- contention
  - avoiding contention and model corruption • 37
  - causes • 35
  - communicating what is going to run • 41
  - DBMS timeouts and deadlocks • 36
  - determining what is running • 37
  - encyclopedia concurrency matrix • 36
  - encyclopedia enqueue waits • 36
- coordinating resource intensive processes • 93
- coordinating shared objects • 48
- CPU usage, optimizing • 10
- CSE
  - backup options • 30
  - cache and threshold settings • 33
  - cleaning up • 31
  - enabling parallel processing • 30
  - estimating size • 24
  - guidelines for taking backups • 31
  - maintaining tables and indexes • 32
  - modifying object cache value • 33
  - modifying override checkout threshold • 34
  - object cache • 33
  - override checkout threshold • 34
  - performing cleanup • 31
  - processes affecting performance • 29
  - resource intensive processes • 29
  - table backup recovery with disk device contents • 30
  - table backup recovery with update logs • 31

---

- table backup recovery without update logs • 30

- CSE performance, improving

- communicating what is going to run • 41

- database object ID partitioning • 41

- project planning and coordination • 46

- reports • 45

- what is scheduled to run • 40

- CSE tables

- fragmentation • 32

- growth rate • 32

## D

- database administration • 63

- database defragmentation • 58

- database object ID, partitioning • 41

- DBMS

- Oracle • 15

- SQL Server • 17

- DBMS timeouts and deadlocks • 36

- DDL customization • 60

- determining what is running

- examining outstanding locks • 38

- functions performing locks on subsets • 40

- functions performing upload locks • 40

- functions performing write locks • 39

- performing read locks • 38

- reviewing lock and ID allocation server

- configuration • 37

- determining, potential checkout downgrades • 46

- DIRGROUP and ENCYGROUP parameters, changing • 90

- disk usage, optimizing • 11

- DOBJ table size comparison • 21

- download, encyclopedia function • 49

- DSUBEX table size comparison • 21

## E

- encyclopedia

- concurrency matrix • 36

- enqueue waits • 36

- monitoring contention • 20

- monitoring profile • 18

- monitoring tables • 15

- monitoring usage • 19

- encyclopedia functions • 49

- adding a new model • 54

- adoption • 53

- download • 49

- migration • 52

- model delete • 55

- override checkout status • 51

- upload • 50

- encyclopedia utility

- adjusting <iefmd>.ini file • 94

- allowing server to process multiple requests • 91

- calculating MinInstance and MaxInstance • 92

- changes to sample utility section in <iefmd>.ini file • 93

- coordinating resource intensive processes • 93

- finding the session • 96

- MaxInstance guidelines • 93

- server utility unavailable • 99

- starting multiple instances • 91

- enqueue waits, encyclopedia • 36

- estimating CSE size • 24

- exporting/importing

- Oracle UNIX • 59

- Oracle Windows • 58

## F

- functions performing

- locks on subsets • 40

- read locks • 38

- upload locks • 40

- write locks • 39

## G

- guidelines, for taking backups • 31

## H

- hit ratio, memory usage • 10

- host vs. CSE table comparison • 21

## I

- I/O contention • 77, 84

- identifying bottlenecks among system resources • 9

- improving CSE performance

- database object ID partitioning • 41

- project planning and coordination • 46

- reports • 45

- what is scheduled to run • 40

## L

- least recently used object (LRU) algorithm, paging of object • 33

---

## locks

- examining outstanding • 38
- functions performing on subsets • 40
- functions performing upload • 40
- functions performing write • 39
- performing read • 38
- reviewing lock and ID allocation server configuration • 37

## M

- maintaining, tables and indexes • 32
- managing CSE workload • 35
- MaxInstance guidelines • 93
- memory usage, optimizing • 10
- Message Dispatcher for Coordination Server, defining exclusive • 90
- Message Dispatcher Monitor tool • 95
- migration encyclopedia function • 52
- migration expert • 45
- MinInstance and MaxInstance values, calculating • 92
- model delete, encyclopedia function • 55
- model\_status, U • 32
- modifying override checkout threshold • 34
- monitoring
  - encyclopedia contention • 20
  - encyclopedia profile • 18
  - encyclopedia tables • 15
  - encyclopedia usage • 19
  - operating system resources • 19
- multiple drives, partitioning CSE database across • 60
- multiple instances of encyclopedia utilities, starting • 91
- multiple physical drives, splitting files across • 25
- multiple requests, allowing server to process • 91

## N

- naming standards
  - shared objects • 47
  - tagging obsolete objects • 47
  - tagging subsets • 47
  - tagging temporary objects • 47
- network problems, avoid • 12
- network usage, optimizing • 12
- non-peak processing • 42

## O

- object cache, modifying object cache value • 33
- optimizing system resource usage
  - anticipating and avoiding problems • 12
  - CPU usage • 10
  - disk usage • 11
  - identifying bottlenecks • 9
  - memory usage • 10
  - network usage • 12
- Oracle Import utility • 74
- Oracle server platform
  - collecting performance statistics • 15
  - database defragmentation • 58
  - DDL customization • 60
  - NEXT and PCTINCREASE reduction • 58
  - partitioning CSE database across multiple drives • 60
  - performing export/import - UNIX • 59
  - performing export/import - Windows • 58
  - routine database cleanup • 57
- Oracle/Windows database customization
  - accessing Public Interface views • 78
  - adjusting logfiles • 77
  - adjusting rollback segments and tablespace • 76
  - adjusting storage parameters • 77
  - distributing data files across disks • 77
  - adjusting init<ency db name and gt • 77
- Oracle/Windows routine procedures
  - adjusting database size • 74
  - backing up database • 74
  - cleaning up database • 73
  - models remain in database until cleanup • 74
  - starting the database • 75
  - stopping the database • 75
- override checkout threshold, modifying • 34
- overriding checkout status, encyclopedia function • 51

## P

- parallel processing, enabling • 30
- partitioning CSE database across multiple drives • 60
- partitioning database object ID • 41
- peak processing • 42
- performance statistics, collecting
  - monitoring encyclopedia contention • 20
  - monitoring encyclopedia profile • 18
  - monitoring encyclopedia tables • 15
  - monitoring encyclopedia usage • 19

---

- monitoring operating system resources • 19
- Oracle • 15
- SQL Server • 17
- Port numbers, assigning to Message Dispatchers • 90
- problems, anticipating and avoiding • 12
- processes
  - affecting CSE performance • 29
  - resource intensive • 29
- processing times
  - non-peak processing • 42
  - peak processing • 42
- project models • 48
- project planning and coordination
  - avoid mass upload • 48
  - coordinating shared objects • 48
  - naming standards • 47
  - project model • 48
  - regeneration and installation of code • 49
  - standards for shared objects naming • 47
  - tagging obsolete objects • 47
  - tagging subsets • 47
  - tagging temporary objects • 47

## Q

- questions, common performance • 105

## R

- regeneration and installation of code • 49
- relative size of indexes to tables • 23
- reports, using to manage heavy CSE workload
  - determining objects needing migrating • 46
  - determining potential checkout downgrades • 46
  - examining listings • 46
- resource intensive processes • 29
- resource intensive processes, coordinating • 93
- reviewing lock and ID allocation server configuration • 37
- rollback segments and tablespace, Oracle Windows • 76
- routine database cleanup - Oracle • 57

## S

- schema tables • 21
- Sequence Queue list • 97
- server administration, if server stops • 87
- server process multiple requests, allowing • 91
- setting the CSE LogLevel

- LimitEachLog • 101
- LogLevel • 102
- LogLocation • 101
- NbrLogFiles • 101
- small tables for user data • 22
- splitting files across multiple physical drives • 25
- SQL Server - Windows database customization • 71
- SQL Server database customization
  - accessing Public Interface views • 72
  - adjusting storage parameters • 71
  - common procedures • 72
  - distributing datafiles across disks • 71
  - setting options to improve performance • 72
- SQL Server platform
  - collecting performance statistics • 17
  - procedure cache • 61
- SQL Server routine procedures
  - adjusting database size • 68
  - backing up database • 66
  - backing up transaction logs • 67
  - cleaning up database • 63
  - creating database backup to disk or tape • 66
  - creating database dump device • 66
  - creating log backup to disk or tape • 68
  - creating transaction log dump device • 68
  - log and database dump devices are separate • 67
  - restoring databases from database dump device • 67
  - starting SQL Server • 70
  - stopping SQL Server • 70
  - truncating transaction logs • 69
  - typical log backups occur daily • 67
  - updating database statistics • 64
  - using DUMP TRANSACTION statement • 69
  - using sp\_dboption system procedure • 70
  - weekly database backups and update statistics • 67
- standards for shared objects naming • 47
- status of server, determining • 95
- subject matter expert, managing CSE workload
  - migration expert • 45
  - subsetting expert • 44
- subsetting expert • 44
- support client, examining outstanding locks • 38
- system resources
  - anticipating and avoiding problems • 12
  - identifying bottlenecks among system resources • 9
  - monitoring • 19



---

- optimizing CPU usage • 10
- optimizing disk usage • 11
- optimizing memory usage • 10
- optimizing network usage • 12

## T

### table

- back up and recovery of CSE tables • 30
- backup options • 30
- DOBJ • 21
- DSUBEX • 21
- guidelines for taking backups • 31
- host vs. CSE table comparison • 21
- large tables for user data • 23
- relative size of indexes to tables • 23
- schema tables • 21
- sizes • 21
- Small tables for user data • 22
- variable-sized tables for user data • 23

### table backup recovery

- with disk device contents • 30
- with update logs • 31
- without update logs • 30

### tables and indexes, maintaining • 32

### tagging

- obsolete objects • 47
- subsets • 47
- temporary objects • 47

### troubleshooting, CSE performance and server administration

- common performance questions • 105

### truncating transaction logs, guidelines • 69

### tuning the Client Server Encyclopedia

- backup options • 30
- cache and threshold settings • 33
- cleaning up the CSE • 32
- CSE processes affecting performance • 29
- enable parallel processing • 30
- guidelines for taking backups • 31
- maintaining CSE tables and indexes • 32
- modifying object cache value • 33
- modifying override checkout threshold • 34
- object cache • 33
- override checkout threshold • 34
- performing CSE cleanup • 31
- resource intensive processes • 29
- table backup recovery with disk device contents • 30

- table backup recovery with update logs • 31
- table backup recovery without update logs • 30

## U

### UNIX/Oracle database customization

- accessing Public Interface views • 85
- adjusting init <ency db name>.ora • 83
- adjusting logfiles • 83
- adjusting rollback segments and tablespace • 82
- adjusting storage parameters • 84
- distributing data files across disks • 84

### UNIX/Oracle routine procedures

- adjusting database size • 80
- backing up database • 80
- cleaning up database • 79
- models remain in database until cleanup • 79
- starting the database • 81
- stopping the database • 81

### upload, encyclopedia function • 50

### user data

- large tables • 23
- small tables • 22
- variable-sized tables • 23

### utility, Compare Report • 46

### UTLUP, non-peak processing time • 42